THE UNIVERSITY OF ADELAIDE
SCHOOL OF COMPUTER SCIENCE

Providing Metacognitive Support Using Learning by Teaching Paradigm

Thesis of Master Project - 2017

Student: Ahoud Alhazmi [a1660266]

Supervisor: Dr. Amali Weerasinghe

3rd of November, 2017

Acknowledgement

First and above all, I praise God (Allah) always and forever in allowing me to complete this research. There are several people without whom this thesis would not have been at all possible. I would like to offer my sincere gratitude and thanks to my supervisor Dr. Amali Weerasinghe for her enthusiasm, motivation, patience and immense knowledge in supervising me. Especially, her guidance and support helped me to clarify my thinking for this research and move forward. Many thanks to my colleague (Refka Maaroufi) for helping to contact with participants during in the experiment days. Also, thanks to all participants for agreeing to take part in the experiment. I want also to express my gratitude and deepest appreciation to my husband, Abdulwahab, who always prayed for my success. I have very very special thanks to my daughter Addanah who shared with me these years in Australia and discovered with me the happiness and burdens of adapting to a new country and lifestyle. Finally, but by no means least, I would like to thank my parents, my sisters, and brothers for their continuous support and their phone calls.

Abstract

Learning by teaching technique is a powerful approach that enhances students to think deeply, orally and repeatedly. However, there are some obstacles to use this technique in school settings such as time-consuming, the anxiety of failing in front of the classmates and finding matching peers. In order to take advantage of this method for the student, there are several computer-based systems have been implemented to apply this approach where students teach the virtual agents to play the tutee role. All of these existing systems focus on various domains, and none of them have considered programming problem solving. In addition to that, the majority of the exiting systems did not provided meta-cognitive support. They only the focus on providing feedback about the content such as providing correct answers. This type of feedback called Knowledge of Correct Response: KCR). In our work, we build a computer-based learning environment that enables the novice programmers to teach problem solving to an animated agent. It combines learning by teaching technique and meta-cognitive support. That will help novice programmers to acquire deep learning on how to solve problems and prepare those programmers for future learning tasks. This project could provide a solution to novice programmers who usually tend to focus on writing the code rather than understanding the problem properly because that would lead them to be frustrated when they do not know how to deal with unfamiliar programming problems.

We conducted an experiment in order to compare the effect of providing guided meta-cognitive feedback and KCR feedback on the novice programmers' skills in learning by teaching paradigm. We implemented two versions of our system. The first version which provides meta-cognitive feedback and the other version which provides KCR feedback. We analysed data from novice programmers, 18-25 years old, who at least studied and passed at least one programming course. They are from College of Computer at Al-lieth in Umm Al-Qura University. The place of the conducted experiment was in the college's lab. We found that the meta-cognitive feedback effect positively on the novice programmers' skills comparing among the pre-test, post-test and delayed test. The performance of 82% of the participants in the experimental group (who received guided meta-cognitive feedback) has been improved after the post-test whereas the performance of only 30% of participants in the control group (who received KCR feedback) has been improved. Although the difficulty of the delayed test compared to the pre-test and the post-test, the performance of 70% of the participants in the experimental group has been improved whereas the performance of only 50% of the participants in the control group has been improved. We are not surprised about the improvement of the control group because learning by teaching technique can encourage (but not to induce) the practice of meta-cognitive skills implicitly whereas the experimental group use learning teaching technique with meta-cognitive support in an explicit way.

Contents

List of Figures

List of Tables

List of Abbreviation

TA	Teachable Agent
ITS	Intelligent tutoring system
LBT	Learning-by-Teaching
AI	Artificial Intelligence
SRL	Self-regulated learning
SRL-C	Self-regulated learning - Cognitive feedback
SRL-A	Self-regulated learning - Affective feedback
GoH	Guardian of history
CTA	Challenging Teachable Agent
DENISE	Development Environment for an Intelligent System in Economics
MCL	Math Concept Learning
KORI	KORea university Intelligent agent
GUI	Graphical User Interface
SWT	Standard Widget Toolkit
KCR	Knowledge of Correct Response
KMA	Knowledge Monitoring Assessment
MSLQ	The Motivated Strategies for Learning Questionnaire
MAI	Metacognitive Awareness Inventory
LPK	Low Prior Knowledge
APK	Average Prior Knowledge
HPK	High Prior Knowledge

Chapter 1

Introduction

1.1 Educational Software

Over the last few decades, the role of technology in education has evolved significantly. All traditional tools in classrooms such as black boards and white boards have increasingly been replaced by digital tools and platforms. Some of these tools and platforms usually include specific software for a particular field, and this software may typically differ according to the targeted educational field. However, it is not enough to provide students with hardware and software that support the acquisition of domain knowledge. More effort needs to be concentrated on how to foster students' skills to help them to be independent learners [1]. Especially, educational establishments are starting to focus on 'differentiated learning' nowadays, because they want to move the education approach from a one-to-many homogeneous experience to a one-to-one deeply immersive, personalized learning experience [2].

For that, many researchers from psychology, education and computer science have worked together for creating educational software, where students can follow a curriculum individually and enhance their learning capabilities [3]. On such support system is intelligent learning environments whose design is informed by cognitive theory and cognitive modelling [4]. These environments are capable of adapting the instruction to each individual student based on Artificial Intelligence (AI) techniques. Thus, they have the ability to support students to accomplish tasks which they cannot accomplish independently.

1.2 Pedagogical Agents

One of the educational software that has been developed. It includes virtual pedagogical agents that can play pedagogical roles such as teachers, students or mentors. This type of software can be categorized into two groups according to the roles of the agents. Firstly, virtual pedagogical agents which play a tutor role to teach a domain knowledge to students and provide materials and exercises. As these agents act mainly as teachers, they evaluate the level of students' learning. This type is called as Intelligent tutoring system (ITS). Teachable Agent (TA) environment is another type of pedagogical agents. It comes from an educational technique called Learning-by-Teaching (LBT), where students can play the teacher role and teach the animated agent using well-structured visual representations [5]. In this project, we will focus on the Teachable Agent Environment since the traditional ITS cannot promote students' motivation and skills.

A Teachable agent is an agent where it is taught by a student new information about a particular subject. This technique (learning by teaching) assists the student to understand the material because she will prepare herself effectively to deliver the information to somebody else. This approach has a positive impact on students' skill as will be discussed in chapter 2.

We will implement the learning by teaching strategy into a computer-based system where human students can teach the agent programming problem-solving. Due to the fact that students are not experts in solving programming problems correctly, and that will negatively impact the performance of the agent when facing a new problem, we will also build another agent to monitor the students' performance when teaching the agent. Therefore, our project will contain two agents. One agent (called Amy) plays the tutee role. The second agent (called Ms. Sarah) will monitor how the human student teaches the agent Amy.

1.2.1 Intelligent Tutor Environments for Programming Problem Domain

As mentioned previously, we are interested in programming problem solving to be the targeted domain in our system especially, due to the lack of teachable agent environments with this domain. Also, this domain need greater metacognitive skills in order to solve the problem successfully.

To give overview about this domain. It is one of the complex and ill-structured kinds of problems [7]. It is not a linear, straightforward process. Rather it is an iterative and cyclical process and involves ongoing monitoring and evaluation. Therefore, metacognition is critical for successful problem solving [8].

Such this domain is considered challenges to development in intelligent systems because it has two types of learning (acquiring declarative knowledge and problem-solving skills). It is very confused to decide the programming problem is well-defined domain or ill-defined domain. The authors of [6] discussed deeply how to distinction such these domains. Programming problem solving is considered as well-defined domain but the instructional tasks in this domain is ill-defined. Based on [6] we mean by domain is declarative domain knowledge or the domain theory while the instructional task is the task the student is learning in terms of problem-solving skills.

To illustrate, the concept of solving any programming problem is well defined. However, the task of developing solutions for a particular problem itself is ill-defined because it is usually underspecified and ambiguous, there is no algorithm to use to come up with the solution. Also, some researcher believes that ill-defined tasks are those domains have multi-correct solutions. However, in a teaching situation, the teacher often has a good pedagogical reason for preferring one particular solution over the others [8].

1.3 Project Motivation

Most novice programmers feel frustrated when they encounter unfamiliar programming problems. One of the main issues is that those novice programmers tend to focus on writing the code rather than understanding the problem properly. Most introductory programming courses often focus on the features of a programming language including syntax [9]. Thus, they may fail to understand the problem or use successful strategies for solving the problem in the first place. Thus, they usually use trial and error strategy. However, their approach could work in some cases but often take longer time and effort. In addition to that, trial and error strategy most likely produce unorganized and unreadable code.

Furthermore, such these courses do not help students how to think, evaluate and monitor about solving problems in general. Students are often concentrate on the textbook questions at the end of each chapter, solved using material discussed earlier in the chapter [10]. Therefore, students in these programming courses acquired the cognitive skills more that metacognitive skills because they are usually taught about the basic cognitive skills such as variables, selection statements and iterative statements. However, mastering only these component skills is not enough for beginner programmers to deal with non-routine problems because they need to know what and how to do [11] [12]. This aspect of the problem is called metacognition. It helps novice programmers about when and how to use specific strategies for problem solving.

Metacognition is a higher order thinking that allows students to understand, analyse and control their own thought processes [13]. In other words, it refers to think about one's own thinking process such as study skills, memory capabilities, and the ability to monitor learning [14]. Furthermore, it plays a significant role of the distinction between experts and beginners in a domain [15].

This project will provide metacognitive support that will enhance novice programmers to think, monitor and control their problem solving skills before generating program code. In order to achieve that, we only focus on the first three steps from six steps to solve any programming problem. Particularly, these first three steps are considered the toughest tasks for novice programmers [12]. The six s six steps to solve any programming problem are formulating the problem, planning the solution, designing the solution, implementation, testing and delivery [16].

1.4 Project Aims

There are several aims of this project:

- To explore how metacognitive scaffolding could be supported on the proposed system (teachable agents environment).
- To assist novice programmers to improve their problem-solving skills in terms of planning, knowledge construction, monitoring, evaluating and revising.
- To assist novice programmers to know their own strengths and weaknesses of programming problem skills.
- To support novice programmers to get the ideal solution.
- To understand novice programmer's progress and then provide feedback on their own metacognitive skills and the domain content.
- To evaluate the efficiency and usability of the proposed system.

1.5 Project Scope

In this project, we build a computer-based learning environment to support novice programmers in problem-solving. It combines learning-by-teaching technique with metacognitive support to enable novice programmers to acquire deep learning on how to solve programming problems and prepare them for future learning tasks. To the best of our knowledge, this project is the first project that provides support to the novice programmers to think about their thinking of solving the problem rather than writing the code.

As mentioned, previously, the proposed system have two animated pedagogical agents. The first agent is Amy which plays the tutee role. The second agent (called Ms. Sarah) will monitor how the human student teaches the agent Amy. Furthermore, it provides several actions for teaching the animated agent. As well, it provides metacognitive feedback in order to develop learner's abilities to monitor the agent knowledge and their own learning and understanding progress. Figure 1.1 shows the summary of the learning cycle in the project. The learning cycle has five components: (i) present the first problem, (ii) teach Amy, (iii) observe Amy's solution of another isomorphic problem, (iv) obtain metacognitive feedback from Ms. Sarah and (v) correct Amy's solution to get the ideal solution.

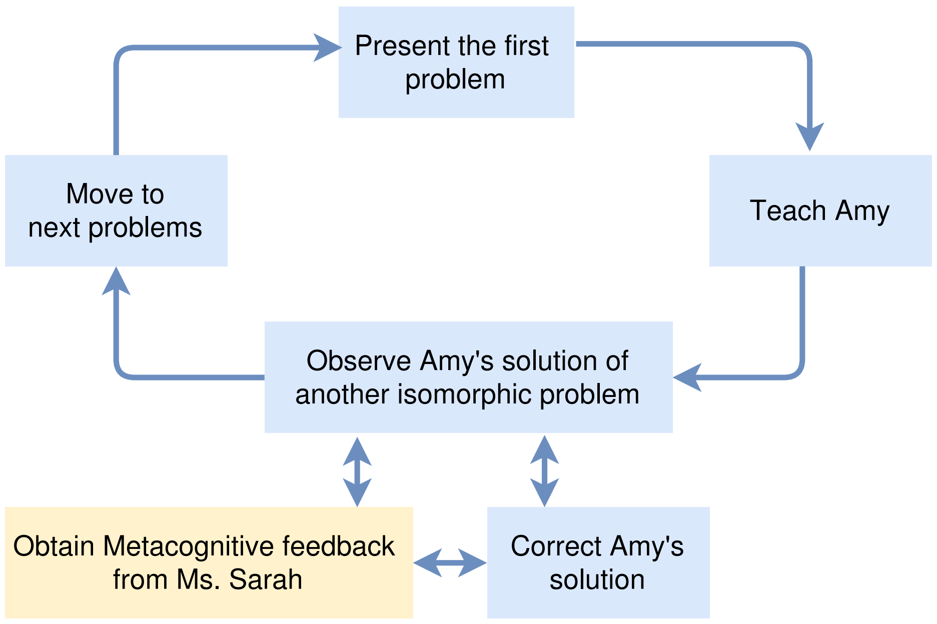

Present the first problem
Teach Amy
Observe Amy's solution of another isomorphic problem
Move to next problems
Obtain Metacognitive feedback from Ms. Sarah
Correct Amy's solution

Figure 1.1: Learning Cycle

1.6 Research Questions

The main goal of this project is to provides metacognitive support to the novice programmers in learning by teaching paradigm. In general, feedback improves students' performance in problem-solving. For that, it becomes important in computer-based learning environments. There are many types of feedback. Firstly, metacognitive feedback is like guided feedback that is to assist a student in solving the problem but without giving the correct solution. Secondly, another type of is called

Knowledge of Correct Response: KCR) it informs the learner of the correct answer to a specific problem with no additional information.

Our research questions are:

1. What is the effect of KCR and metacognitive feedback on novice programmer's own problem solving skills (planning, monitoring, evaluation, debugging skills)?
2. How can KCR and metacognitive feedback affect on novice programmers to adjust their approach for teaching the teachable agent (Amy)?

1.7 Project Significance

Learning by teaching helps learners to understand the domain knowledge in a profound way. The authors [17] report that people who teach others in the interest of preparing themselves for a quiz have performed better than those who prepare only themselves for the same quiz. The authors believe that because during the preparing stage of teaching others, people will be forced to organize their ideas to gain a deeper understanding of the materials in order to be able to explain the materials in a simple way for others [18]. Moreover, authors in [19] [20] found that students who teach others spent more time in learning than those who spent the time to learn themselves, and that leads to gain more knowledge in the targeted domain.

Therefore, this project's aim is to enhance novice programmers to prepare themselves and organize their knowledge of how to solve problems using metacognition approach. We aim to make the students improve their thinking of solving a programming problem before start coding any solution for that problem. In addition, this project provides support to novice programmers who seem to be frustrated about their learning and organizing the knowledge as well as monitoring their own learning progress for solving programming problems.

As mentioned previously, metacognition is very significant in the learning process and solving a problem. For that, lack of metacognitive skills makes analysis problem-solving task di cult for novice programmers. However, providing metacognitive support, explicitly or implicitly, will improve novice programmers to gain expertise and solve any problem successfully. [21] recommends that the metacognitive support should be explicit support because it makes the support for metacognition apparent and induces the learner to practice metacognition whereas implicit metacognitive support does not induce (only encourage) the learner to practice metacognition. Therefore, metacognitive feedback will assist novice programmers to prepare themselves in the future learning in effective way.

Previous studies have designed systems which have the characteristic of learning by teaching technique for programmers, where these systems allow those users to teach animated agents (which are as students or peers). Also, these systems provide directed feedback to the users regarding missing information. directed feedback such as KCR is useful only in early stage of learning process [22]. For that, metacognitive feedback would reinforce an exploratory and constructivist learning approaches for programmers.

1.8 Structure of the Thesis

This document includes seven chapters **Chapter 1** introduces the research topic and presents project motivation. It also presents the scope, aims of the project and an overview of the research questions with the significance of the project. **Chapter 2** is the literature review. Firstly, It presents the relationship between metacognition and problem-solving process in Programming with showing the difficulties of Novice Programmers. It also presents the role of metacognition to enhance the thinking of novice programmers. Secondly, it presents the benefits of learning by teaching approach for learners Thirdly, it describes some existing teachable agent environments and presents also the limitation of these existing systems. After that, Chapter 3 and Chapter 4 are dedicated to our system. **Chapter 3** describes the design of the system and **Chapter 4** describes the implementation of the system in details.

Chapter 5 presents the experimental study of this research. It was an empirical evaluation of novice programmers' interaction with our system focusing mainly on the observation of how metacognitive feedback in learning by teaching paradigm affects on performance changes. The experiment was undertaken with undergraduate students at College of Computer at Al-lieth in Umm Al-Qura University. The design of the experiment is presented with the materials that used in the experimental study. **Chapter 6** summarises and discusses the results Also, it presents and main contributions of this work. Finally, **chapter 7** presents the conclusion, the contribution of this work, and the limitations. Also, it includes a section to explain the future work.

Chapter 2

Literature Review

In general, this chapter includes three significant parts. The first part introduces the metacognition definition and relationship between it and problem-solving in programming. Also, it presents the difficulties of Novice Programmers and how the metacognitive support can enhance their thinking. The second part gives an overview about learning by teaching technique. It also presents the advantages of this technique and the obstacles of using it in the school setting. The third part presents the existing system that supports learning by teaching paradigm with presenting the limitation of these systems.

2.1 Metacognition Definition

Metacognition is a term in educational psychology filed. It comes from the root word "meta" - to refer "thinking about thinking" or "knowledge about knowledge". Metacognition is a higher order thinking that allows students to understand, analyze and control their own thought processes [10]. In other words, it refers to think about one's own thinking process such as study skills, memory capabilities, and the ability to monitor learning [11]. Metacognition is as an awareness about the student to learn or solve a problem because it allows learners to plan and monitor their learning process to improve their performance [12] [13]. The authors of [12] [14] [1] defined the components of metacognition as shown in 2.1. Firstly, it is knowledge about cognition which is called metacognitive knowledge that consists of (i) Person variables knowledge which is understanding one's own capabilities, (ii) Task variables knowledge which is how one perceives the difficulty of a task which is the content, length, and the type of assignment. (iii) Strategic variables knowledge which is one's own capability for using strategies to learn information. Secondly, it is regulation of cognition which is called self-regulation and consists of the following: planning, monitoring, evaluating, and revising.

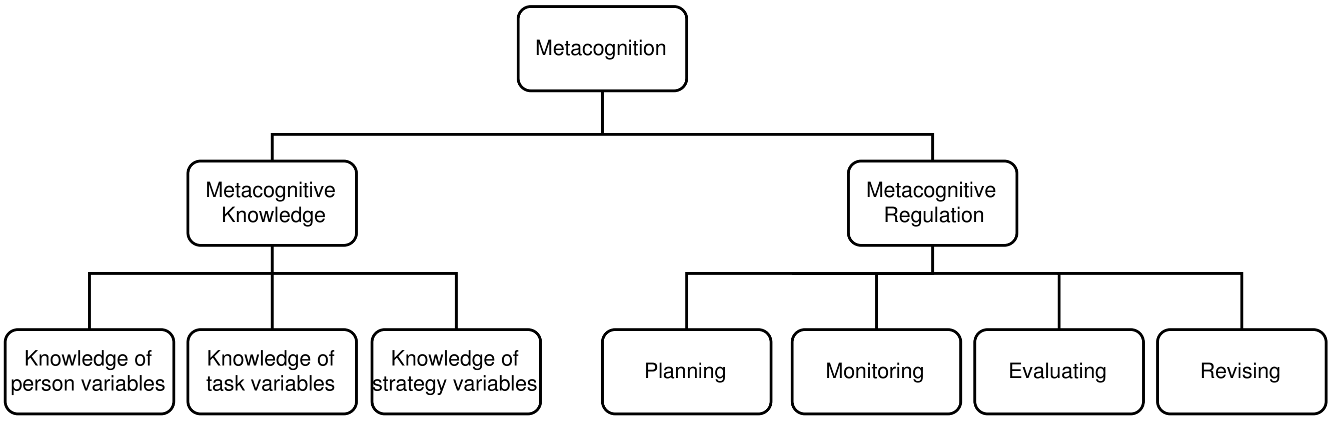


Figure 2.1: Components of Metacognition [1]

Metacognition plays a critical role for many of our daily activities such as reading comprehension, problem-solving, social cognition ... etc. In other words, it is a key factor for learners to be successful as well it is related to intelligence [15][16][17][18]. Furthermore, paper [19] displays how metacognition was included in the thinking objectives such as critical thinking, decision-making, problem-solving, creative thinking and searching for meaning. For that, metacognition is the most factor that distinguishes between poor learners and good learners. In [15] the authors argue that good learners are more flexible in their methods to deal with problems and they are more self-monitoring. As well, they have a larger repertoire of strategies and they are more organized about their strategies to ensure their activities works in the appropriate sequence. Furthermore, we will explain more about its benefits to novice programmer in order to solve a programming problem in section 2.2.

2.1.1 Difference Between Metacognition and Cognition

The difference between metacognition and cognition is complicated. For example, when we read a text, we need cognitive skills. This is different when we want to monitor our own understanding of text. That is considered metacognitive skill. Another example, novice programmers know about basic cognitive skills such as constants and variables; comparison and logical operators; selection statements; iterative statements; and arrays. When they have to know when and how to use them that is considered metacognitive skills. [20] defines that cognitive strategies help learners to achieve a goal whereas metacognitive strategies ensure the achievement of a goal. However, theses previous examples are not clear enough to distinct between metacognitive and cognitive skills. In [20] the authors show the example about the interchangeability of cognitive and metacognitive functions. For example, the activity is about looking for the main points in a text. That also can be seen as the strategy itself due to monitoring function and reflection of the knowledge.

In [12], Flavell's model of metacognition assumes that metacognitive and cognitive are distinct in their content and function, but that there is a resemblance in their quality and form. The similarity between them, they can be acquired, forgotten, corrected or not. In contract, they are different in terms of contents. The contents of metacognition come from the learner's internal mental representations (such as skills about cognition) but the contents of cognition come from the learner's external reality. The second difference is about function. The function of cognition is to solve the problem, but the problem solver needs to regulate their cognitive operation in order to execute a task that is considered the function of metacognition.

2.2 Metacognition and Problem Solving

Problem solving is complex mental process. It is an iterative and cyclical process, and it involves ongoing monitoring and evaluation. For finding solution, problem solver needs three factors. [11] explains these factors are domain-specific knowledge (content knowledge in a specific discipline) and structural knowledge (the knowledge of how concepts within a domain are interrelated). In addition, metacognition is the third factor that is very significant because it involves ongoing monitoring and evaluation.

There are three cyclic stages of problem solving process in any discipline: familiarization stage, production stage and judgement or evaluation stage. The first stage is about understanding the problem. It is very important stage in problem solving because a correct solution depends on this stage more than others [21]. The second stage about producing the solution paths. The third stage is about evaluating the solution which could help to select the best solution.

In our daily lives, when we solve a problem, we acquire something. However, the performance for solving a problem could be different from person to another. What is demanding and challenging to a person might be easy and simple to another thus is not considered as a problem solving situation. The interaction of person's experience and the demands of the task make the problem-solving are different from person to person [22][23]. The authors in [24][25] demonstrate who utilize their intellectual skills consciously and who show perseverance and flexibility in problem solving. They have well-developed their metacognitive abilities. Therefore, metacognition in general helps problem solver to recognize and figure out the problem to be solved and understand the idea of how to reach the solution. There are many metacognitive processes in problem solving domain involved in (i) determining and identifying what kind of problem it is, (ii) representing and developing mental model of problem, (iii) planning how to proceed especially when problem is novel and (iv) evaluating the performance [20].

2.2.1 The Relationship Between Metacognition and Problem-Solving Process in Programming

In general, the steps of problem solving include the identification and the definition of a problem, the formulation of a strategy, the organization of information, resources allocation, observartion and monitoring, and evaluation [26]. Solving programming problems requires similar steps. [27] presented six steps of problem solving that help a programmer to develop their model. Theses steps are formulating the problem, planning the solution, designing the solution, implementation, testing and delivery. In this project, we will focus on the first three steps because the programmer would need them before generating the code. These steps are considered the difficult tasks for beginners since the first three steps require the problem solving ability to solve any problem. Most likely, those novice programmers lack these three steps.

For that , the importance of metacognition is very obvious. [28] found students who use metacognitive strategies outperform in programming than who do not use it. Furthermore, [29] stated whenever a programming problem increases the complexity, a student will need more metacognitive control for their process because they need to understand the problem, devise a plan, correct errors, test program output and think deeply about their solutions. In [30] authors propose four metacognitive skills that a programmer needs: understanding and interpreting the problem correctly, determining the required steps sequentially, choosing the best solution with considering programmers' skills and evaluating the solution correctly.

2.2.2 The Difficulties of Novice Programmers and the Essential Metacognitive Skills

In [31] defined novice programmers is a person that lack the knowledge and programming skills. Their approach for solving the programming problem is that they read the problem statement and start directly writing the code. That means they are confused between problem solving and coding. Writing the code is the stage of expressing the solution to the problem in a way that it can be transferred to the computer [32]. Additionally, novice programmers usually use the trail-and-error strategy. In some cases, their approach works but often takes time and effort. As well, their code would be unreadable and hard to understand. Indeed, the limitations of their approach is that they use line-by-line of written code rather than understanding the big picture of program structures. In [32] they emphasize the problem solving is a heart of programming rather than coding. For that, we will focus on how to solve programming problem before starting coding.

When novice programmers resort to non-productive trail-and error strategy, they develop their own problem solving strategy. For that, they need about 10 years to become an expert programmer [33]. Researchers investigated what the difference between novices' and experts' methods for solving a programming problem are. Although, beginner programmers face grammatical problems in a programming language, they have a limitation to decompose problems. They are not skilful at problem decomposition, and they usually use the low level of abstraction. In contrast, experts have the ability to decompose problems into sub-problems and manage it easily. In addition, they can find alternative solutions and choose the best solution [34][35]. Thus, lack of strategies and tactical knowledge of problem solving is one of the challenges that the student should deal with.

As mentioned previously, novice programmers usually learn about the basic cognitive skills such as: constants and variables, comparison and logical operators, statements of selections, statements of iterations, and arrays. However, they face a problem about how and when to use the cognitive skills. [36] argued that most students are able to solve routine problems (familiar problems to the students and they know how to solve) because students can engage both cognitive and metacognitive skills easily. However, to solve non-routine problems (problems have been failed to solve in the past or are new), students need metacognition. Furthermore, in [30] the author confirms the importance of metacognitive skills that enhance the performance of the qualities of a successful problem solver.

To emphasize the critical role of metacognition to distinguish between novice and expert programmers, authors of [37] reported that expert programmers (i.e. who had at least two years experiences of the filed) were superior to the novice programmers (i.e. who were trained in the domain but did not have any work experience) only in the metacognitive knowledge of the programming task. Experts programmers have more abstract hierarchical organization based on the principles of program functions, whereas novice programmers have only syntax-based knowledge organization. Furthermore, they found novice programmers lack the mentoring goal-setting, plan for finding solution, self-monitoring skills and awareness of errors in a problem. Thus, providing metacognitive support will help novice programmers to become strategic learners and expert programmers. For that, one of the parts of our project will be responsible for providing metacognitive feedback to the user in order to develop their metacognitive skills and apply these skills during the programming problem solving in the future.

2.2.3 Polya's Problem Solving Techniques

A problem solver always uses step-by-step method to get closer to the end of solution. As mentioned previously, beginner programmer lacks three steps which are considered hardest tasks that leads to frustration. Firstly, it is formulating the problem that explaining preliminary problem description clearly and structuring the problem representation. Secondly, it is planning the solution that refers to strategy discovery, goal decomposition. Thirdly, it is designing the solution which refers to organization and refinement [27].

In [38] George Polya (Professor of Mathematics) displayed primary importance to successful problem solving skills. His method has four steps as shown in figure 2.2 understand the problem, devise a plan, carry out the plan and look back. The process of these steps is not linear, the problem solvers need to go back and forth to the steps until they reach the goal. For instance, problem solver tries to create a good plan. Then in this stage, she discovers a need to understand the problem better. So, she will go back to understand the stage and evolve their understanding of the problem.

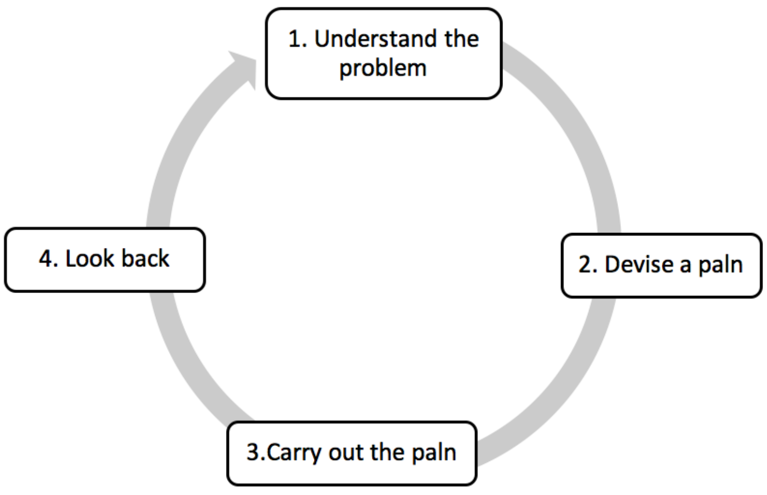


Figure 2.2: Polya's Approach to Problem Solving

Despite that Polya uses his process for mathematical problem solving, his approach to problem solving was used in other disciplines. In [32] used Polya's approach with the programming problem. They also suggest two extra processes after the look back step. They describe what the learnt lessons from the process and write summary about assumptions that have been made and conditions that must be met before the solution. However, we will not consider these both steps because we expected the users will notice by themselves after following the metacognitive feedback.

Furthermore, [32][38] suggest some actions for each of the previous steps. Firstly, it is about understanding the problem. The beginner programmers need to break the problem into sup-problems and determine the requirements of the problem such as input and output. They also need to examine the special cases of the problem. In some cases, they need to separate various parts of condition. Secondly, it is about obtaining a plan of the solution. They need to try to simplify the problem by solving the easiest sup-problem. As well, the users need to design what they are doing. Also, they need to check all the information in the problem statement have been considered. Thirdly, it is about carrying out the plan. It is writing the sequence of the solution for example as pseudo-code with checking the order of the actions correctly. Lastly, it is about verifying any defects in the solution in order to find some bugs.

2.2.4 Guided Metacognitive Feedback versus Other Types of Feedback

Before explaining what the difference between the metacognitive feedback and others, we will presents the feedback as generally. Feedback is one of the most powerful influences on learning and achievement. It is conceptualized as information provided by an agent (e.g., teacher, peer, parent, self, experience) regarding aspects of one's performance or understanding. [39] provided an excellent summary in their claim that "feedback is information with which a learner can confirm, add to, overwrite, tune, or restructure information in memory, whether that information is domain knowledge, meta-cognitive knowledge, beliefs about self and tasks, or cognitive tactics and strategies".

The main aim of providing feedback is to assist students to develop and increase their knowledge, skills, and understanding in some content area or general skill (e.g., problem solving). The temporary assistance from a teacher helps students to complete the tasks efficiently. Vygotsky [40] defined this area as the zone of proximal development. For that, teachers usually ask themselves how the feedback should be. That authos of [41] suggested that feedback should be specific, accurate, timely, clear and necessary to encourage a person in order to change their thinking and then improve their experience.

The benefits from the feedback is the students can reduce the cognitive load of learners, especially novice or struggling students [42]. These students can become cognitively overwhelmed during learning due to high performance demands, and thus they may benefit from supportive feedback designed to decrease the cognitive load.

The feedback factor consists of three main elements: (a) the content of the feedback (i.e., evaluative aspects, such as verification, as well as informative aspects, such as hints, cues, analogies, explanations, and worked out examples); (b) the function of the feedback (i.e., cognitive, metacognitive, and motivational); and (c) the presentation of the feedback components (i.e., timing, schedule, and perhaps adaptively considerations)[43].

There are many types of feedback depending on the content of the feedback [43]. [[44] explain the some of them. First, correct response feedback is known as knowledge of correct response (KCR), it informs the learner of the correct answer to a specific problem with no additional information. Secondly, it is knowledge of result as known as verification feedback and it informs the learner about the correctness of her response(s), such as right/wrong or overall percentage correct. Error-flagging is also known as location of mistakes (LM), error-flagging highlights errors in a solution, without giving correct answer. There are others.

There are many studies that examining which the good feedback type has to be provided the students. For instance, giving feedback response as only correct or incorrect report (Knowledge of Response: KOR) is less useful to learners than providing feedback with the correct found answer (Knowledge of Correct Response:KCR) [44]. [45] discovered that students require feedback that does not provide too many comments and also gives more guidance to be useful for their improvement. Since, they will think more about how to fix the mistakes and they will use their intellect skills. The defining characteristic of intelligent novices seemed to be an ability to control and monitor their own thought processes.

Intellectual skills are defined as the methods that an individual can use to evaluate or organize information and data [46]. The authors in [24] [25] demonstrate who utilize their intellectual skills consciously and who show perseverance and flexibility in problem solving. They have well-developed their metacognitive abilities. In general, metacognition helps problem solver to recognize and figure out the problem to be solved and understand the idea of how to reach the solution. [47] explain metacognition helps learners to plan, monitor, and regulate or change their learning strategies. Furthermore, [12] stated the contents of metacognition comes from the learner′s internal mental representations (such as skills about cognition) but the contents of cognition come from the learner′s external reality. In order to improve learner′s learning performance, supporting and guiding learners′ metacognitive control becomes necessary. One effective method facilitating learners′ self-regulation and learning may be to provide feedback on their metacognitive processes, such as decisions about which and how to use cognitive strategies [48].

Metacognitive feedback is the communication that makes a learner conscious of the learning strategies and styles being used and the degree of success. Feedback is an accepted strategy for increasing the learner′s awareness of what is unknown, what knowledge is needed, and what learning strategies work [49]. For that, providing feedback on metacognitive skills of novice programmers help them to aware their strengths and weaknesses. Then in future, Those who know their strengths and weaknesses in this areas will be more likely to "actively monitor their learning strategies and resources and assess their readiness for particular tasks and performances". Furthermore, the absence of metacognition connects to the research by the authors of [50]. They found that "people tend to be blissfully unaware of their incompetence," lacking "insight about deficiencies in their intellectual and social skills."

2.3 Learning by Teaching

Learning by teaching is a powerful method to learn. It is recommended by educators and cognitive science researchers because it increases in the use of techniques such as memorizing, organizing and reflecting [51]. It has three phases: preparing, teaching, and gaining feedback [2].

Firstly, when preparing to teach, tutors consider two questions about delivering the knowledge: what the knowledge is and how to deliver it. So, some studies found this approach is better than traditional learning approach which the learners only listen. For example, [52] found the performance students who prepared to teach others to take a quiz is better than those who prepared to take the quiz themselves. Furthermore, preparing to teach other people enhances the sense of responsibility. [53][54] found the students spent more time in order to teach computer agent than they spent time to teach themselves. Also, preparing to teach others can engage metacognition because teachers usually prepare themselves to anticipate the questions of their students. For that, they are more organized when they explain information. For example, [52] found that college students (who prepare to teach) understand the topic deeply more than students (who prepare to take a quiz). Similarly,[55] reports that students (who prepared to teach) gain deeper understanding of the materials.

Although learning by teaching approach forces the learner to prepare themselves better, it enhances three critical aspects of teaching interactions. Firstly, it is re-structuring of teacher's understanding. Tutors often answer the pupils' questions. These questions lead the teachers to recognize and repair gaps in their own understanding [55]. Secondly, it is taking responsibility. The teachers usually ensure that their students leave their classroom with all the tools they need to continue their learning on their own [56]. Thirdly, it is the reflection. Efficient teaching process demands the explicit monitoring of how the information is understood and used. For that, teachers often reflect on their interactions with students during and after the teaching process [57]. This reflection helps teachers to evaluate their own understanding of domain knowledge and evaluate their approach that was used to transfer this understanding to their students.

After teaching process, the teachers observe how their students apply what they have been taught. In [2] differs between direct and recursive feedback in learning by teaching paradigm as shown in Figure 2.3. In addition, they found the students-tutor (who receive recursive feedback when they observed their agent) use logic to solve novel problems compared to those who receive direct recursive feedback.

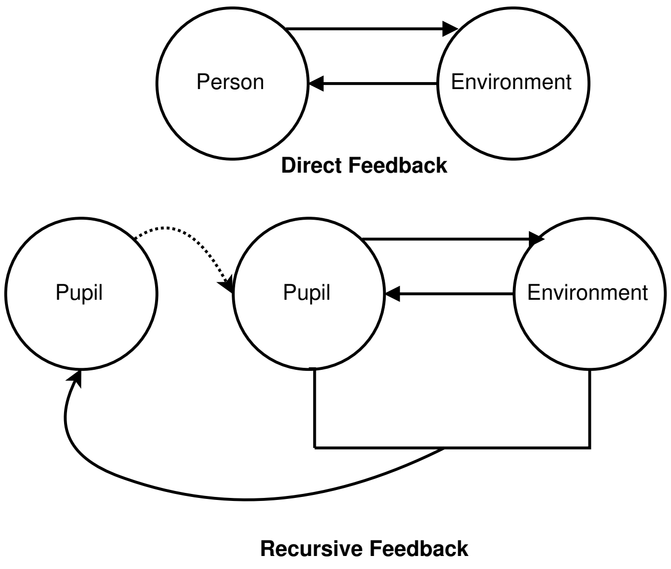


Figure 2.3: Direct and Recursive Feedback [2]

2.3.1 Drawbacks with Learning by Teaching in School Settings

However, this technique often is not used in the class because of some factors. It takes time and effort. Especially the numbers of students have been increased and time of scheduled class is limited. In addition, some students do not prefer this method because of their anxiety from failing in front of their classmates. Furthermore, there is another drawback which is finding matching peers. For example, student-teacher could be too low competence compared with his students who are too high competence. In this case will reduce the possibility of knowledge exchange [58]. In addition, [59] found a student with high competence prefers a teacher that exhibit similar characteristics. To overcome these limitation, many researches have created educational software.

2.4 Teachable Agents Environment Background

As mentioned previously, teachable agents environments is computer system that uses learning-by-teaching technique where students explicitly teach an computer agent. It helps to enable learning process to be an effective and significative manner by using a variety of computing technologies. The difference between teachable agents environment and the traditional Intelligent Tutor system is how to the student's evolving to interpret information. In traditional ITS, the virtual tutor tries to modify the student's knowledge to converges to the tutor. However, this approach is very static one because the students do not monitor their learning activities. Different from traditional ITS, when students identify the information and they monitor and evaluate their learning progress.

Thinking about learning process, it is more than learning content. Some researchers discuses how to design guidelines in order to create the teachable agents. According to [60], there are three main fields is considered the basis on Teachable agents environments as shown in Figure 2.4. Education filed helps the design of teaching strategy and detect what is the best pedagogical intervention for each individual to provide feedback. Psychology provides insights into how we learn and feel and computer Science assists how to design artificial intelligent systems.

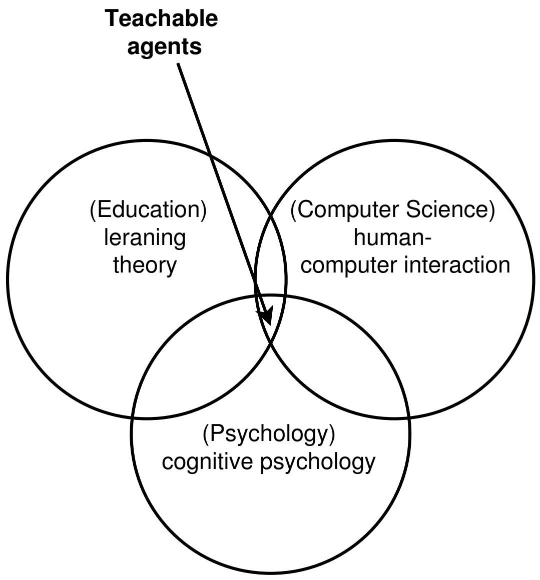

Teachable agents
(Education) leraning theory
(Computer Science) human-computer interaction
(Psychology) cognitive psychology

Figure 2.4: Disciplines Involved in Teachable Agent Environment based on [3]

In our project, we will need dealing with these three fields. From the educational perspective, our system will focus on learning by teaching approach where a user behaves as a human teacher in order to teach a computer agent about problem-solving skills. Furthermore, problem-solving is considered as one of teaching strategies that relies on motivating students to analyze, think, propose and test alternatives or hypotheses. As well, the effectiveness feedback is very important in the learning process. From the perspective of Psychological filed, it helps researchers to understand the learning process of students and assist in simulating the agent with behaving like a human. All these factors should be realized through a computer-based system. For that, Artificial Intelligence (AI) (which is subfield in computer science) becomes necessary at this stage in order to understand the student approach in the teaching process and monitor them during receive feedback.

2.4.1 Pedagogical agents

Educational software includes pedagogical agents which take the role such as the expert person, a student, a peer competitor or collaborator. In this project, we will focus on the teachable agents (TA). The first a computer agent is taught by the user. this agent is dependent on the student when it is taught. Furthermore, [53] state this type of agent has to behave independently. Also, there is another agent who mentor the problem solving approach and provide metacognitive feedback

2.4.2 Previous Work of Teachable Agent Systems

A number of systems have been designed where students can interact and teach a computer agent and try to learn from that interaction. These systems have shown effect on student's learning processes positively [61]. In addition to that, these systems motivate students to spend more time and effort to get their goal from teaching the agents [53] [54] [6].

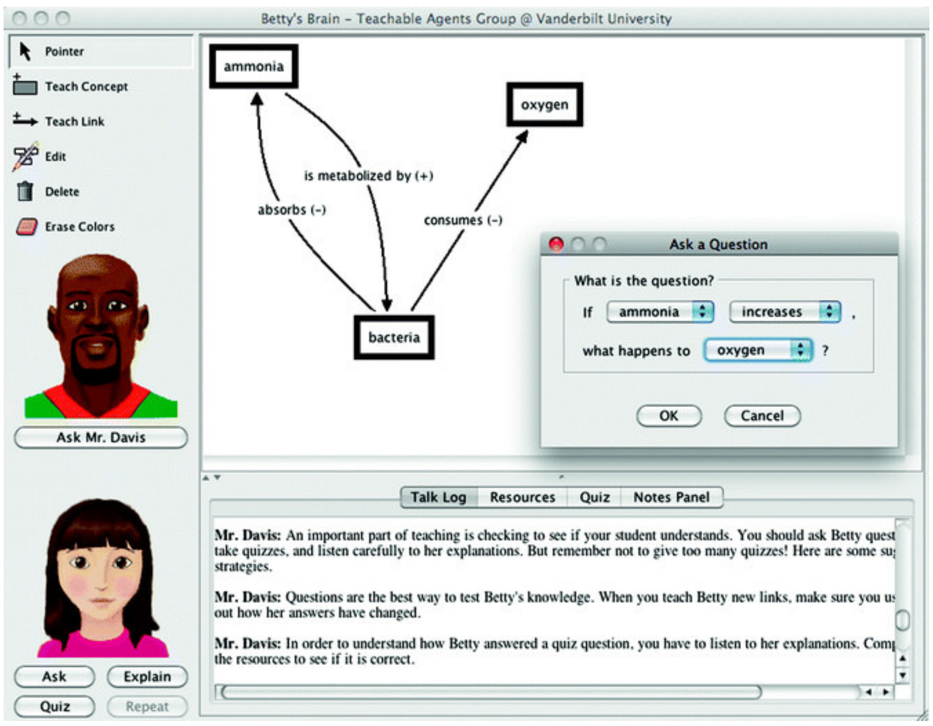
Different from traditional intelligent tutoring systems, human students can engage in identifying, planning, developing solutions and evaluating their own ideas and solutions. For that, they have to unfold, examine, and reflect on their own ideas.

2.4.2.1 Betty's Brain system

Betty's Brain has been improved for more than 10 years which is built by Vanderbilt University by the Teachable Agents Group. We will focus on Betty's Brain Version 2 because they apply metacognitive learning theories in order to aid learners in developing metacognition strategies in the future. In general ,Betty's Brain system combines learning by teaching technique with self-regulation mentoring to help learner to understand the topic deeply. However, in Version 1, they only focus on developing learner's cognitive strategies about the specific topic in science.

[62] [63] [4] [64] designed Betty's Brain system that is an open-ended learning environment where middle school students teach a virtual agent about river ecosystem. The interface of this system is shown in Figure 2.5. Middle school students are encouraged to learn by teaching computer agents(called Betty) through the use of three strategies. They can teach Betty using concept map representation. They can also query her with their own questions to see if she understands. She uses qualitative reasoning methods for answering questions. That operate through chains of links from the source concept to the target concept In addition, Betty can take a quiz to see how well it performs

in that quiz. The questions in quiz will be selected by Mr. Davies (who is another virtual agent). He will compare the user's concept map to the expert map and finding some incomplete ,missing or incorrect links or concept. Then he will provided some questions about that. In order to organize the student's learning and teaching process, there is other activates that user can use them: reading resource , editing the causal map and listening to Betty's explanation.



What is the question?

If ammonia increases ,

what happens to oxygen ?

Mr. Davis: An important part of teaching is checking to see if your student understands. You should ask Betty quest take quizzes, and listen carefully to her explanations. But remember not to give too many quizzes! Here are some su strategies.

Mr. Davis: Questions are the best way to test Betty's knowledge. When you teach Betty new links, make sure you us out how her answers have changed.

Mr. Davis: In order to understand how Betty answered a quiz question, you have to listen to her explanations. Com the resources to see if it is correct.

Figure 2.5: Betty's Brain System [4]

Interestingly, there are different types of feedback that can be provided by a mentor agent (called Mr. Davis). The first type is content feedback that includes correct and incorrect answers that Betty made on the quiz after every quiz. The second type is guided metacognitive feedback that will help human students to monitor their metacognitive strategies for future learning. The metacognitive feedback aids users in order to decompose the resources correctly, understand the resources to convert the information to map as well as monitor Betty's understanding continually. All theses feedback will help user to gauge their progress and find deficiencies in learning process. Furthermore, there is another feedback is provided from Betty but this is inquisitive rather than explicitly instructive. She focuses on the interactions between the user and herself, ineffective behaviors . For example, if the user has not been asking her to explain many answers. She will say "Could you listen to my explanations and make sure that they match what the resources say?".

There were a lot of experiments on this system. In [63] the authors wanted to compare among traditional learning , Learning-by-teaching and learning-by-teaching with metacognitive support for self-regulated learning, they divide students into thesed3 groups. The first group used ITS where Mr. Davies asked user to create concept map then it would correctly answer of test question. The second group used LBT where users teach Betty to help her pass test. The third group used self-regulated learning (SRL) where user teach Betty to pass test Betty behavior integrated metacognitive support.

In beginning, the results showed the users in ITS and LBT groups outperformed than SRL groups about the task of creating concept maps to answer specific questions. The reasons for the previous performance for SRL group is this group spent the first time in learning self-regulation strategies but after that, they understand these strategies. Thus, their performance improved extremely. Furthermore, SRL Students developed better learning and monitoring strategies and they also had the ability to continue the learning and teaching process than other groups. In addition, this group is able to learn and understand the new material better than other groups.

Due to the many of forms of feedback, in [4] they wanted to examine which the best feedback mechanisms in teachable agent systems in order to improve students' abilities to monitor their agent's knowledge as well as to improve their own process of understanding and learning. For that, they divide students into 3 groups similar in [63] and they focused on feedback mechanism in the three groups. The ITS and LBT groups received only corrective feedback, whereas, SRL group received guided metacognitive feedback. The results show the corrective feedback help the students only in the early stages of the quiz than who received guided feedback. Interestingly, metacognitive feedback assists the student for preparation for future learning task . Furthermore, [4] studied how the guided feedback may affect the student's learning behaviors. They created two version of SRL System. Firstly, SRL-cognitive feedback (SRL-C) is content directed feedback and hints that help the user applies metacognitive strategies to improve the learning, monitoring and debugging task.
Secondly, SRL-affective(SRL-A) feedback is similar to SRL-C feedback, but the responses are emotional rather than content-oriented. Table 2.1 show the examples about SRL-A and SRL-C response. The result demonstrate SRL-C response help students to extract information from the text recourse and creating concept better than SRL-A group . Also, SRL-C had better learning perform in new domain. However, the performance of students who recieve SRL-A feedback is not good to prepare the students in the future because after the metacognitive support was removed. They did not show any improve for their performance whereas SRL-A group show better performance even metacognitive support is removed.

Table 2.1: Feedback Example of SRL-C and SRL-A [4]

Pattern	Cognitive Response	Affective Response
If after four questions, Betty has not been queried on an unlinked concept	Excuse me. You taught me a concept, but didn't teach me any relationships between it and other concepts. Please teach me more, and ask me questions to make sure I understand	Hey, I'm confused and I don't under- stand what you taught me. Please teach me more, and ask me some questions.

Overall, [63][4] show that metacognitive support helps users to prepare better in future learning even metacognitive support is removed. However, getting metacognitive feedback could not help novice learner in the early stage of the learning process but after that assist the users to learn and understand new material better than others. Focusing only on corrective feedback will help the users only to pass quiz but without understanding the important correlation between the concepts. That appears from the attempted in resources, quires and quiz access, LBT group focus only on getting quiz right answer. They do understand the important correlation among them. In other hand, metacognitive feedback will assist the user to pass quiz with abilities to learn and understand better such as shown with SRL-C group.

2.4.2.2 Guardian of History System

Another system that uses learning by teaching paradigm is Guardian of history(GoH). This system includes learning activities that correspond to the Swedish national curriculum for history in 5-6th grade [5]. Users can teach the agent by using concept map and timeline as shown in Figure 2.6 and 2.7. The main idea from this system is to add challenges to the traditional teachable agent such as Betty's Brain in order to see how users can handle theses challenges. The agent in this system is perceived as the troublemaker peer agent whereas Betty agent is a very polite student. The added challenging behaviors are three challenges. Firstly, it is introduction of error. The agent will give an erroneous suggestion, then user will react and offer a correct solution. The second challenge is the rejection of correct fact. The agent will refuse the proposal answer (ex. I think that you picked the wrong period). The user will confirm the answer or withdraw his answer. The third challenge is proposal of higher level of difficult. The agent will encourage user to select highest level. For that, GoH system would help human students to be more self-confident on differentiating between correct and incorrect answers. However, the system does not provide any metacognitive feedback.

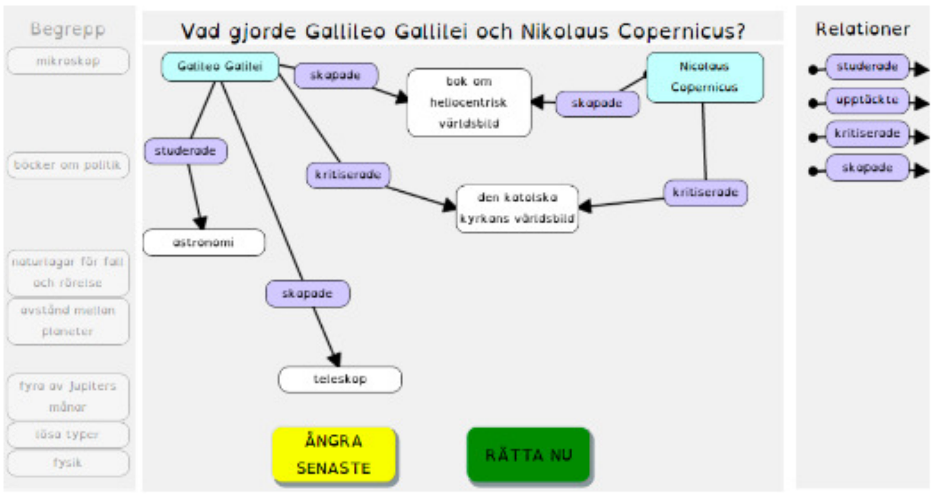


Figure 2.6: GoH Interface Using Concept Map [5]

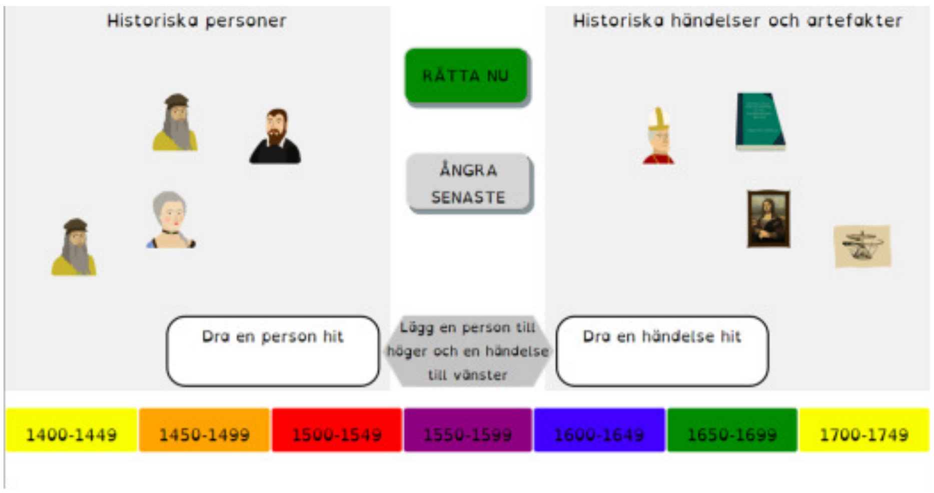


Figure 2.7: GoH Interface Using Timeline [5]

There is experiment on this system. Data is collected from 146 students, in 5-6th grade, from a Swedish school. Different agent conditions are used: traditional teachable agent (TA) or a challenging TA (CTA) in order to compare the results. The first and second challenging behaviors (introduction of error and rejection of correct fact) are used in two conditions in agent: traditional TA and CTA. When a student teaches the agent wrong concept in TA condition. Then, the agent will suggest the same concept to the student during the quiz time. However, the behavior frequency is higher in the CTA condition than Traditional TA. The third challenge is the proposal of a higher level of difficulty only occurs in CTA condition

The results show no difference between Traditional TA and CTA on the the learning gain. Both of them were positive. In addition, the results show students in the CTA condition show high level of self-efficacy. This finding explains the finding of Bandura (1994) that students people who have a high level of self-efficacy better respond to difficulties and attributes it as something positive.

Sometimes, we think if someone refuses our answers in class, it could make kind of confusing to students. However, the authors in this paper found students accept when the agent proposed an incorrect fact in TA condition. However, students in CTA condition have a higher appropriate response to incorrect fact proposals. That means, Students get better decisions about correct and incorrect answer in CTA condition. Also, they found the rejection of correct is better accepted than the introduction of errors. Then, it also conducted in a higher appropriate response rate.

2.4.2.3 A Teachable Agent Game

[65] [6] designed and implemented a teachable agent game to engage primary school children to learn basic arithmetic concepts and reasoning as shown in Figure 2.8. They combine two techniques learning-by-doing (playing the game) and learning-by-teaching (teaching the agent to play) in order to leverage engagement, reflection and learning. Additionally, they use an agent-driven question dialogue which offers three benefits to children. Firstly, it is to challenge students' mathematical thinking. Secondly, it is to explain implicitly how ideal learner behaves. Lastly, it is to transfer game knowledge to out-of- game mathematics. Also, this system focus only on corrective feedback. Interestingly, the agent in this system is designed to mimic the characteristics of an ideal learner who is active, creative, self motivated, and curious person. For example, the agent asks deep questions such as: why, why not, how, and what-if/not. It does not ask a shallow question such as: who, what, when, and where.

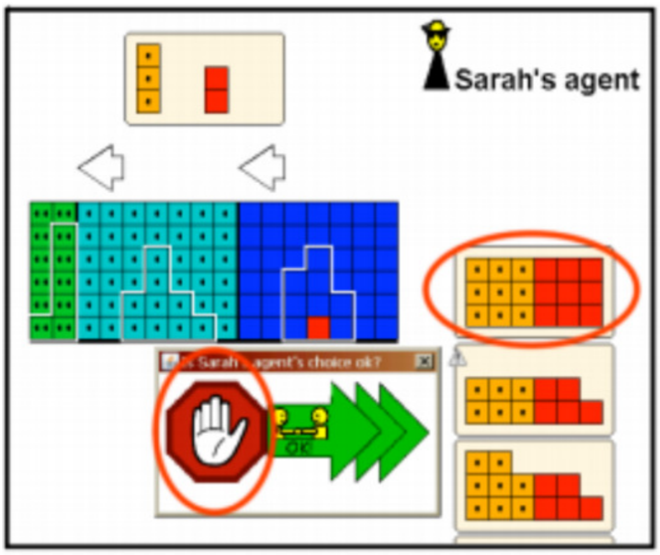
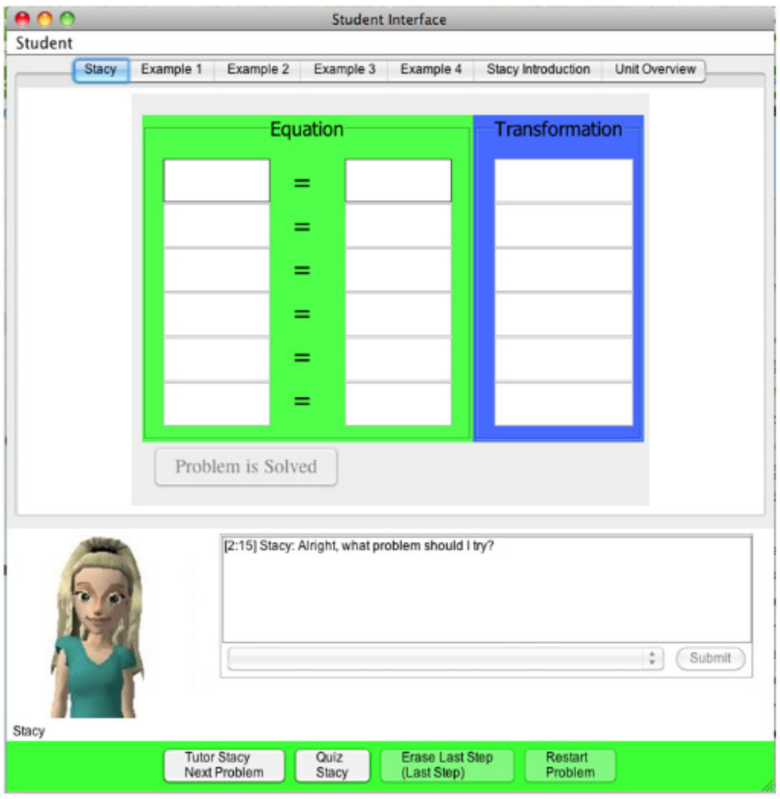


Figure 2.8: Teachable Agent Game Interface [6]

In [65] the authors reported findings from an evaluation study of the game. They divided participants into two groups. The first group is who use the game and teach the agent and the second group is who attend standard instruction and take the test. In terms of learning gain, the first group have better learning gain than the second group. They learned more about problems dealing with the conceptual understanding of the base-10 system when playing on their regular mathematics lessons, and they did not drop behind in other mathematical skills being tested but not practiced on in the game. That shows playing students group have better learning gain especially in the category of conceptual understand. Thus, this study supports a connection between learning in the game and learning outside the game. Furthermore, the results show young primary students can act as successful tutors. They can teach the agent, response the questions and explain. However, grade 2nd students had challenges during the teaching the agent due to their lack of reading. Thus, they were the only group who judged playing the game as more enjoyable than teaching the agent.

2.4.2.4 SimStudent system

SimStudent is another computer agent is taught linear equations as shown in Figure 2.9 . It is designed to be suitable for young students from 6th to 8th grade students [66]. Users can teach the agent by collaboratively solving a problem [67]. The user provides an equation to the agent. Then, the agent attempts to answer it depending on what has been taught. After that, it will ask the user whether a certain operation is correct or incorrect. If the answer is correct, the agent will move to the next equation, but if the answer is incorrect, the agent will ask a hint from the user in order to find the correct answer. In addition, the system allows the user to refer similar examples in the interface. During the teaching stage, the system will store the student's problem solving. Then during quizzing stage, the agent will use the similar pattern to solve the quiz problem. Thus, the incorrect procedure will avoid the agent to complete solving a problem whereas the correct procedure will assist the agent to complete solving a problem.



[2:15] Stacy: Alright, what problem should I try?

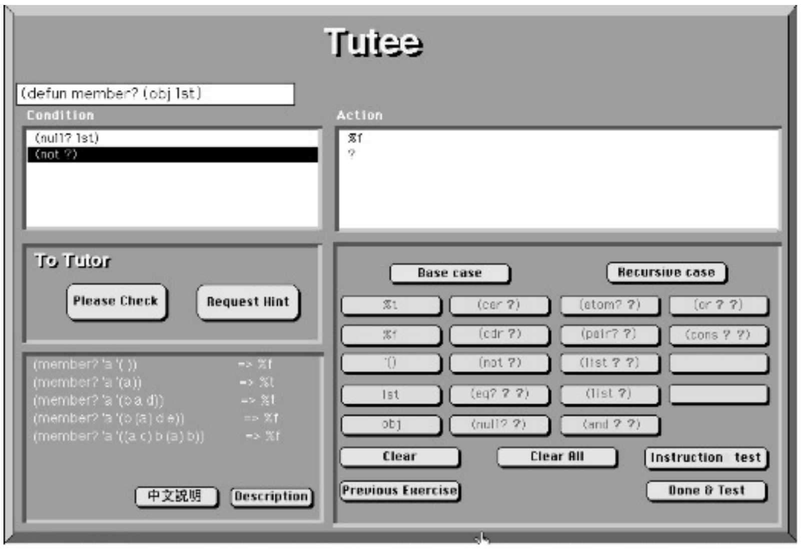
Figure 2.9: SimStudent Interface [7]

Their studies showed that users usually select problems are familiar with them[7]. For that, [68] is developed the second version of the system which offers metacognitive scaffolding on what problems users should teach. Mr. Williams is another agent in this system which takes the role of the teacher agent. He provides metacognitive help when asked. He suggests to students what problem should be tutored next and explains why (ex."Since Mandy was wrong on the quiz, you may want to give 4y-8=10 to Mandy") [68].

The results in [66] [67] show the students' skills improve after tutoring the agent for about 70 minutes in average. They improved their skills with solving equations and identifying errors in given solutions. Furthermore, the results show the students with low proficiency on equation solving often got stuck when providing a hint to the agent's request for what to do next. The system provides Curriculum Browser contents and example problems. However, these students still did not use these. Also, when they want to provide an example to the agent, they started with a number and some arithmetic to make another number. For example, the student may said, *"I'll start with 4, that is an x. If I multiply it with 3, I get 12. I'll add 5, which is 17. So, 3x+5 = 17"*.Thus, these students did not understand the conception related to equation solving. However, in[68] the authors reported the metacognitive help given by Mr. Williams allowed users to select appropriate problems that affected both the agent and their learning especially with who usually used only familiar examples.

2.4.2.5 Other Systems

There are many of system where students can teach a computer agent in this section we will provided only description about theses system. Firstly, Chan & Chou's system [8] showed different types of agents (a tutor and a peer) for teaching Lisp programs. Specifically, we found only this system which focuses on computer science domain. However, this system combines between the traditional intelligent tutoring system and the teachable agent environments. The agent and the user take reciprocal roles. For example, User is tutor and agent is tutee. As well, the user is tutee and agent is tutor. Figure 2.10 shows the interface for tutee role. However, this system focuses only on giving debugging tasks. However, it teaches only syntax, so such as system could not help users to recognize and detect that there is a problem to be solved.



```
(defun member? (obj lst)
```

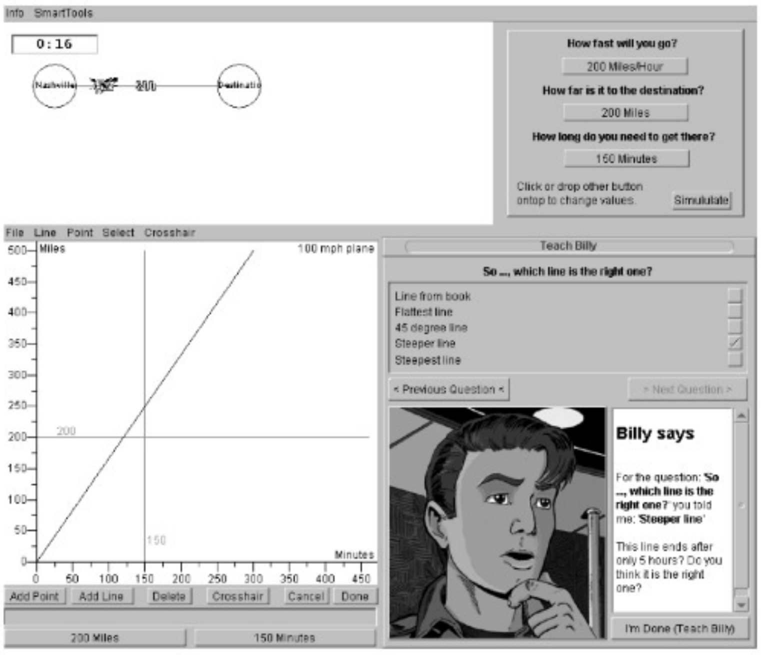
```
(null? lst)
(not ?)
```

```
%f
?
```

```
(member? 'a '( ))            => %f
(member? 'a '(a))            => %t
(member? 'a '(b a d))        => %t
(member? 'a '(b (a) d e))    => %f
(member? 'a '((a c) b (a) b)) => %f
```

Figure 2.10: Chan & Chou's Tutee Interface [8]

AdventurePlayer system is another system that helps users to teach computer agent (called Billy) about solving a sequence of problems involving the delivery of packages among cities as shown in Figure 2.11 [60]. The targeted users are undergraduate students in Cognitive Science. Firstly, users will read the introduction of the problem. Then, they teach Billy and they see and correct his behaviour. At the end, if the users have taught Billy successfully, they can move on to the next problem. In general, the problems require Billy to choose among airplanes that travel at different speeds (100, 200 and 300 mph), and decide on alternate routes to perform the deliveries in a way that minimizes the time for delivery.



How fast will you go?
200 Miles/Hour
How far is it to the destination?
200 Miles
How long do you need to get there?
150 Minutes
Click or drop other button ontop to change values.

So ..., which line is the right one?

Line from book
Flattest line
45 degree line
Steeper line ✓
Steepest line

Billy says

For the question: '**So ..., which line is the right one?**' you told me: '**Steeper line**'

This line ends after only 5 hours? Do you think it is the right one?

Figure 2.11: AdventurePlayer System [9]

Moby is another TA environments. It was designed to help high school students to learn biology through a process of hypothesis induction and testing [2]. These system has two agents: Good Moby and Evil Moby. Evil Moby is a competing agent that provides wrong and incorrect solution.

Furthermore, DENISE (Development Environment for an Intelligent System in Economics) is another system where the agent can be taught economics using a dialog template [69]. The agent probes the student-teacher for more information on material as it is taught, but the student can take control at any time, and specify new relationships instead of following the agent's directive probes. Students have access to a dictionary that contains all of the concepts they have taught the agent. They can also query the agent about relations between concepts. However, this system does not show any representation that help user to monitor their information that was taught to the agent.

Furthermore, MCL (Math Concept Learning) system is very similar to SimStudent system in terms of design stages. MCL system was developed for solving simultaneous linear equations [70]. Students teach the agents by creating example solutions to linear equations. Then, it uses its built-in knowledge of the structure of linear equations to learn problem solving strategies in the form of rules from the users' examples. Additionally, both of them are implemented by using two learning strategies: learning from examples and learning by tutored problem solving.

Similarity to Betty's Brain, in [71] authors designed and implemented the KORI (KORea university Intelligent agent), in which primary students play a role of a science tutor. Those students teach the agents by drawing a concept map and posing questions. Then, KORea system provides only immediate feedback to that agent abut content. However, this system doe not provided ant guided metacognitve feedback. Furthermore, [71] explained how the agent generates its own knowledge by the inference engine and updates it through the feedback mechanism.

2.5 The Limitation of the Existing Teachable Agent Systems

Table 2.2 shows a comparison of the designs of previous systems. It includes the teaching domain, the takes of the user , targeted audience, the roles of the agents , scaffolding tools and providing the feedback and its type. All the system the take of the user is a teacher, but Chan & Chou's system is only the system the task of the user could change to be tutee.

Teachable agents have been built for many domains such as math, Biology , programming...etc. These teachable agents environments were very appropriate to any domain. Furthermore, Learning by teaching paradigm is vey suitable for any educational level such as in primary school [5][65][6], middle school [62] [63][67] , high school [2] and undergraduate [8][9][69]. Significantly, most the pervious systems show the students performance in learning by teaching is better that traditional teaching process.

As mentioned previously, learning by teaching has three phases: preparing to teach, teaching process and gaining feedback. All of the pervious system considered the first two phases (preparing and teaching process). In contrast, Betty's Brain , SimsStudent and GOH also considered the third phases. For that, containing all these phases make the learning by teaching very effectively. Thus, our system will focus on these phases especially the feedback.

Furthermore, the type of feedback is vey significant to improve the performance of the learners for future about the domain. As mentioned previously, providing feedback about content (correct or incorrect answers) does not help the user to prepare themselves in future learning. Only Betty's Brain and SimStudent were implemented to provided metacognitive support to the user. Metacognitive help in SimStudent is to support the student to select the appropriate problems but it does not help the problem solve to improve their own skills to solve those equations. However, [4] report how metacognitive feedback in Betty's Brain helps the user to improve their performance in future learning. Thus, our system will provided metacognitive feedback in order to improve the thinking process of problem solver.

The simplicity of the using the system is very important to the user. If a system is very complicated to use, such as includeing complex interface. This lack of an explicitly shared representation restricts the students' learning abilities. Some of the previous systems are the sample to used such as Betty' Brain, GoH and SimStudent. However, Moby system was very complicated to use comparing to other systems because it includes the complex interface. DENISE system has the same limitation.

In [66] the authors recommend to the agent which takes the role as students or peer tutee should not have any prior knowledge because it would affect the tutor's learning outcome. Some systems is build agents that pretend to learn from the users, but they have full knowledge of the domain internally [51] such as GoH. Conversely, Betty and SimStudent does not have any knowledge before the user teach her. Therefore , our agents will not have any prior knowledge.

Table 2.2: Summary of All Previous Teachable Agent Systems

No	System	Domain	Task of the User	Targeted Audience	Number of Agents	The Role of Virtual Agents	Scaffolding Tools	Types of feedback
1	Betty's Brain	Ecosystem	Tutoring	Middle School Students	2	Betty as student agent and Mr. Davis as mentor agent	Concept map, posing questions and taking quiz	content and metacognitive feedback.
2	GoH	History	Tutoring	For 5th and 6th grade	1	As troublemaker peer agent	Concept map and timeline	Content feedback
3	A Teachable Agent Game	Basic Arithmetic	Tutoring	Primary school students	1	Ideal peer tutee	4 categories of games: Find Pair, Pack Many, Remove All, and Divide.	No
4	SimStuden	Linear equation	Tutoring	6th and 8th grade	2	As students and the second agent as meta-tutor	A set of cells representing either the left- or right-hand side of an equation or a transformation	No
5	Chan,& Chou's system	LISP Programming	Tutoring and Tuteeing	Undergraduate students	1	Tutee and tutor	Diagnosis-Hint-Tree	Corrective feedback
6	AdventurePlayer system	Solving problems (involving the delivery of packages between cities on the specific map)	Tutoring	Undergraduate students	1	Tutee	Choosing on the map	No
7	Moby	Biology	Tutoring	High school	2	Competitive Peers	Propositional and matrix menu	No
8	DENISE	Economics	Tutoring	Undergraduate students	1	Competitive Peers	A dialog template	No
9	MCL system	Linear equation	Tutoring	Unidentified	1	Tutee	Creating example solutions to linear equations	No
10	KORI	Ecosystem	Tutoring	Undergraduate students	1	Tutee	Concept map and posing questions	No

Chapter 3

System Design

This chapter is dedicated to explaining our system design. The main goal of this chapter is to describe what framework we used in designing the system, what problem-solving skills and programming problem-solving contexts we chose to include in the system, and the reasons for the selection. Then, we present the steps that are needed to design the system from designing the virtual agent and processes in our system. Also, this chapter presents the architecture of the system and how the user interface was designed and developed for different phases of the system.

3.1 Adapting Polya's Approach in Solving Programming Problems

As mentioned in the last chapter, Polya's approach introduces the thinking process of how to solve the mathematical problems. As we know that introductory programming is close to mathematical problems [3]. As a result, it will be suitable to apply Polya's approach on solving basic programming problems.

The author of [32] suggests some strategies for each stage in Polya's approach because he claims that these strategies would assist the programmer to get rid of confusion and concentrate on problem solving rather than writing code. Figure 3.1 shows what the most strategies would be useful for our problems.

As mentioned previously, we do not focus on implementing the solution using any programming language because the novice programmer needs to enhance their thinking process about the problem rather than writing the code and dealing with syntax in a particular language. Concentrating on the characteristics and syntax of the used programming language would shift those beginner programmers to focus on how to write the code of the problem rather than how to comprehend the substantial algorithmic model to solve the problem in an accurate way [13]. As result of focusing on syntax could produce any solution to the problem without enhancing the ability of students to be sure if the provided solution is accurate or not.

As mentioned before, the novice programmers find non-routine problem solving is very hard because they do not know what and how to do although they know the cognitive skills such as variables, logical and comparison operators and selection statements. However, they need metacognition that

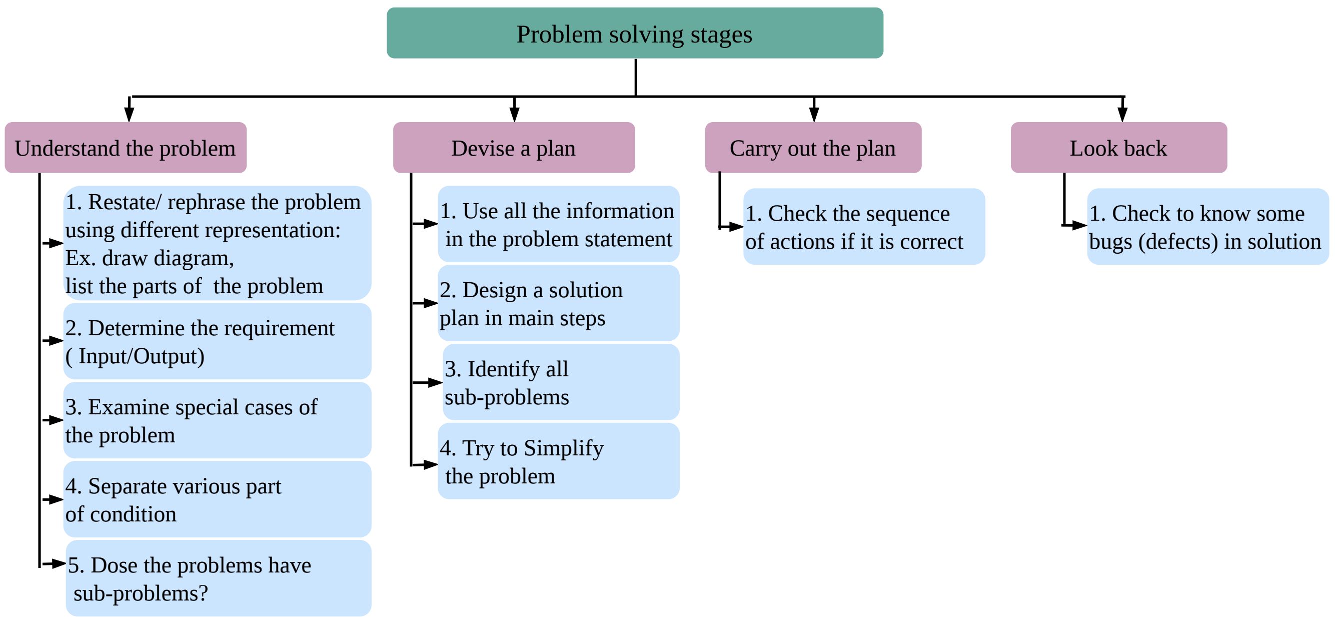

Problem solving stages
Understand the problem
1. Restate/ rephrase the problem using different representation: Ex. draw diagram, list the parts of the problem
2. Determine the requirement (Input/Output)
3. Examine special cases of the problem
4. Separate various part of condition
5. Dose the problems have sub-problems?
Devise a plan
1. Use all the information in the problem statement
2. Design a solution plan in main steps
3. Identify all sub-problems
4. Try to Simplify the problem
Carry out the plan
1. Check the sequence of actions if it is correct
Look back
1. Check to know some bugs (defects) in solution

Figure 3.1: Problem Solving Stages Based on Polya's Approach

help them to find a solution with non-routine problem [36]. We will discuss the process explained in Figure 3.1 on how to solve the following example:

(A) Write an algorithm to calculate N! ?

(B) Write an algorithm to calculate x^y (x and y are integer numbers)?

(C) Write an algorithm to test each number between 1 to 99 determine if it is odd and prints the odd numbers?

The first stage in solving the three problems above is to correctly understand what each problem requires. This step could be a significant challenge to students if they have not faced these problems before. Consider the example (A), it seems very simple to understand. However, if the beginner focuses only on positive integers and forgets the zero factorial, that means the students missed examining the special case of the problem before testing the solution. That refers to that the student does not understand the problem completely.

Furthermore, novice programmers could not separate various part of conditions in this stage. Consider the example (B), students could focus only in the case (y) if it is greater than zero and they do not pay attention when (y) is equal to zero or less than zero. That means students do not notice the various parts of condition of this problem. Thus, students do not understand the problem completely and determine the steps required to solve the problem.

After the stage of understanding the problem completely, the next stage is how to devise a plan to solve the problem. Authors of [30] explained that in cases where the problem was misunderstood, it is likely to produce solutions which sometimes provide correct results. Consider the example (C), students may understand only printing odd numbers which outputs all odd numbers individually by adding 2 as shown in Figure 3.2. However, this solution explains the student does not use all the information in the problem statement which is testing the number if it is odd number or not. That means students lack the ability to evaluate their own solution.

```
OddNumbers
   1.For counter =1 to 99 with step =2 do
          1.1.Print counter
End
```

Figure 3.2: Pseudocode of Finding Odd Numbers without testing

After device a plan stage, students need to carry out the plan. They have to write down the sequence of actions necessary to solve the problem and ensure the order of the actions if it is correct. Hint, in this stage, we talk about writing pseud-code before implementing using any programming language. For example, students could notice what the problem needs such as how to read the width and length and then how to calculate the area of a rectangle. That shows the students know the required steps to solve the problem. However, if they do not know the order the steps sequence correctly (eg. changing the ordering between reading input and calculate the area of rectangle), that means students have weaknesses in verifying the order of the steps sequence of a solution correctly.

The last stage, students have to look back at the solution in order to find if there is any bug. There are many examples for possibility to have defects in the solution such as the start and the end of a loop statement and also in logical operator such as < , <=. We expected if the students find this bug before testing code, that could help them to improve their thinking process of the problem solving.

However, our goal is not to teach novice programmers about the problem solving skills directly, but we want them to enhance their thinking process about the problem after receiving feedback from our system.

3.2 The Context of the Chosen Problems

At this stage, we have developed 16 pairs of problems. The first stage in the proposed system, the user will teach the virtual agent about the first problem and then the second stage is to let the virtual agent to solve the second problem from the pair based on the way taught by the student.

Our selection of the problems was based on the near-isomorphic problems such as structured problem representation, the sequences of process and number of the cases. However, in some cases the difference between two problems is not a big deal in our project because we want to focus on the user's thinking about the problem solving strategies as known as metacognitive skills rather than cognitive skills.

The 16 pairs of problems are grouped in 6 classes. First group contains very simple problems. Second group includes problem that should be solved by using conditions. Third group encompasses problems that need iterative process to be solved. Fourth group contains the recursive problems. Fifth group are the problems that should use nested loops to be solved. In the sixth group, we try to use complexed problems that may require several skills to be solved. In this this group, we selected string matching problems. Table 3.1 shows these problems.

Table 3.1: The 16 Pairs of Chosen Problems

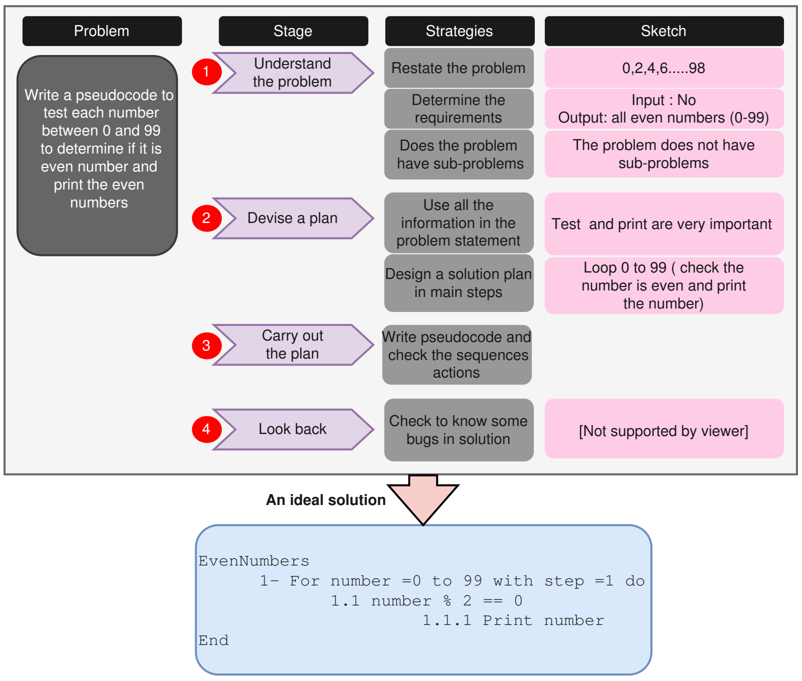
Group Type	No	Problems in Teaching Stage	Problems in Quiz Stage
Group 1: Simple Problems	1	Write an algorithm to change the time from seconds to minutes	Write an algorithm to change the time from minutes to hours
	2	Read in a number representing a temperature in Celsius and print it out as a value in Fahrenheit. If the Celsius value is C, then the Fahrenheit value F is calculated as follows: F = (9 / 5) * C + 32.	Read in a number representing a temperature in Fahrenheit and print it out as a value in Celsius. If the Celsius value is C, then the Fahrenheit value, F, is calculated as follows: C = (5 / 9) * F - 32.
	3	Write an algorithm to calculate the area of a circle and print the result : use the formula : A=(3.1416)*r^2	Write an algorithm to calculate the volume of sphere and print the result : use the formula : V=(4/3) * (3.1416)*r^3
	4	Write an algorithm to calculate the average of homework grades	Program calculate the average temperature of seven days
Group 2: Using Conditions	5	Write an algorithm to change a numeric grade to pass/no pass grade Hint: >= 50 pass <50 no pass	Write a pseudocode to read a real number and print (positive) when it is greater than zero and negative otherwise
	6	Write an algorithm to change a numeric grade to a letter grade • IF numeric grade between 100 and 90 = "A" • IF numeric grade between 89 and 80 = "B" • IF numeric grade between 79 and 70 = "C" • IF numeric grade between 69 and 60 = "D" • IF numeric grade <60= "D"	Write an algorithm that performs the following: Ask a user to enter the year of birth and print out to the user's generation as follows: • IF year in between 1946 and 1964 = "Baby Boomers" • IF year in between 1965 and 1984 = "Generation X" • IF year in between 1985 and 2000 = "Generation Y" • IF year >2000 = "Generation Z"
Group 3: Using Iterative Process	7	Write an algorithm to find the biggest of a set of integers	Write an algorithm to find the smallest of a set of integers
	8	Write an algorithm to test each number between 1 to 99 determine if it is even and prints the even numbers.	Write an algorithm to test each number between 1 to 99 determine if it is odd and prints the odd numbers.
	9	Write a pseudo code to print all multiples of 5 between 1 and 100 (including both 1 and 100)	Write a pseudo code to print all multiples of 4 between 1 and 100 (including both 1 and 100)
	10	Write an algorithm to calculate N!	Write an algorithm to calculate X^y (X and Y are integer numbers)
	11	Write to convert binary to decimal	Write to convert decimal number to binary
	12	Sort list of characters in the alphabetical order	Sort the list of numbers in ascending order
Group 4: Recursive Problems	13	Write a recursive algorithm in pseudo-code to find the combination of n objects taken k at a time using the definition C(n, k) =,1 if k=0 or n=k = c(n-1, k) + c(n-1, k-1) if n >k >0	Write a recursive algorithm in pseudo-code to calculate the Fibonacci sequence of n objects taken k at a time using the definition Fib(n)=0 if n=0 =1 if n=1 = Fib(n-1)+Fib(n-2) if n>1
	14	Write a recursive algorithm to find the greatest common divisor (GCD) of two integers using the definition GCD(x,y)= x if y=0; GCD(y, x mod y) otherwise	Write a recursive algorithm to find the least common multiple (LCM) of two integers using the definition LCM (x,y) = x if y=0; LCM x*y/ GCD (x,y) otherwise
Group 5: Nested Loops	15	Alternating,disk: you have a row of 2n disks of two colors n dark and n light. They alternate dark, light, dark , light and so on. You want to get all the dark disk to the right hand end and all the light disk to the left hand end. The only moves you are allowed to make are those that interchange the positions of two neighboring disk	rearrange the order of words: so that all vowel on the right hand and all the consonants on the left hand and print how many Vowels and how many consonants with print how many letters in Vowel as well in constant
Group 6: Advanced Problems	16	Write an algorithm that can determine if substring is part of a given string " Some researchers found this result" and print " find or not"	Write an algorithm that check a given (HTHTH-HTHTTTHH) attempts of throwing a coin if the tails occurred three in sequence and print "Find or not"

As we have 16 pairs, the 32 problems are assigned into the teaching stage and the quiz stage. The quiz stage contains the complicated problem from the pair. For example, the problem of finding X^y is complicated than finding N! because it contains more conditions. The reason we are doing this is because we want students to monitor and reflect on their own strategist for solving the problem after providing feedback to their student agent. That will support the students to enhance their ability of the thinking process.

For the current stage in our project, we have selected randomly 5 pairs out of the aforementioned 16 pairs. Our goal is to examine these problems and investigate the relationship between expected errors might committed by users and their level of metacognitive skills.

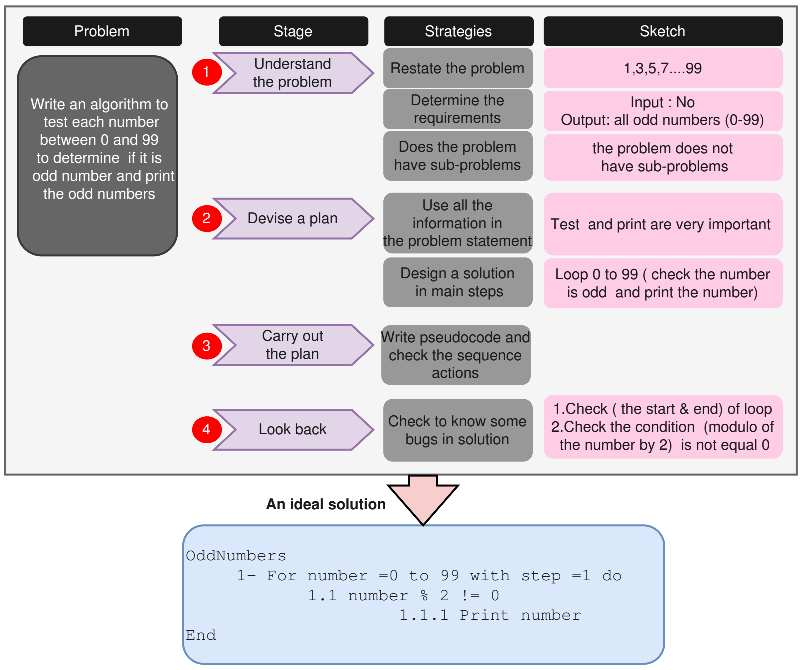
The first pair is about finding odd or even numbers from a specific range of numbers and print these numbers as shown in Figure 3.3 and Figure 3.4. These figures also show the approach of a problem-solving based on the polya's approach we have covered in section 3.1 (fig). For each problem, we analysed the problem and provide one of the ideal solutions that student may produce. However, there are some expected errors that could be committed by the user in both problems. They would not know which is the start and end numbers of the iterative statement. In the examples provided, the start and the end numbers are 0 and 99 respectively. In addition to that, the loop could be started from number 1 in the odd example. The solution would be correct but in this case, students have the lack of ability in verifying whether the solution of the problem is linked to their knowledge of the task requirements.

There are other expected errors. The student might not test the numbers whether even or odd, or might not print the results. As mentioned previously, the student does not use all the information provided in the problem statement. That means student lacks the ability to evaluate their own solution. Furthermore, the student may not know how to write the formula to test the numbers whether odd or even.


Problem
Stage
Strategies
Sketch
Write a pseudocode to test each number between 0 and 99 to determine if it is even number and print the even numbers
1
Understand the problem
Restate the problem
0,2,4,6.....98
Determine the requirements
Input : No
Output: all even numbers (0-99)
Does the problem have sub-problems
The problem does not have sub-problems
2
Devise a plan
Use all the information in the problem statement
Test and print are very important
Design a solution plan in main steps
Loop 0 to 99 (check the number is even and print the number)
3
Carry out the plan
Write pseudocode and check the sequences actions
4
Look back
Check to know some bugs in solution
[Not supported by viewer]
An ideal solution

```
EvenNumbers
     1- For number =0 to 99 with step =1 do
           1.1 number % 2 == 0
                  1.1.1 Print number
End
```

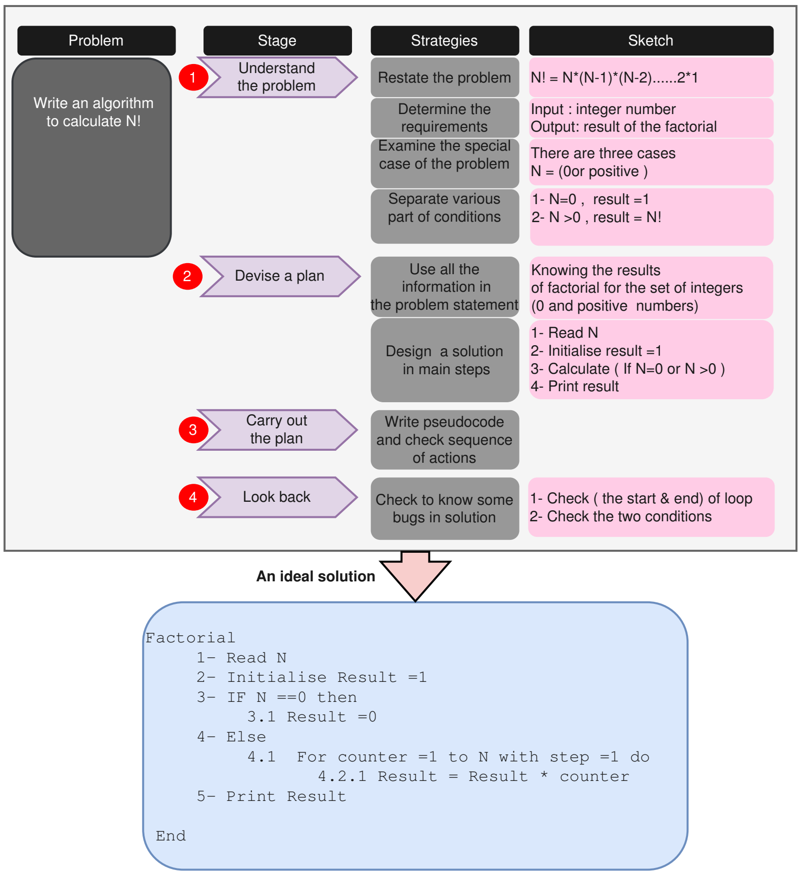
Figure 3.3: Solving the Even Numbers Problem Based on The strategy shwon in Figure 3.1


Problem
Stage
Strategies
Sketch
Write an algorithm to test each number between 0 and 99 to determine if it is odd number and print the odd numbers
1
Understand the problem
Restate the problem
1,3,5,7....99
Determine the requirements
Input : No
Output: all odd numbers (0-99)
Does the problem have sub-problems
the problem does not have sub-problems
2
Devise a plan
Use all the information in the problem statement
Test and print are very important
Design a solution in main steps
Loop 0 to 99 (check the number is odd and print the number)
3
Carry out the plan
Write pseudocode and check the sequence actions
4
Look back
Check to know some bugs in solution
1.Check (the start & end) of loop
2.Check the condition (modulo of the number by 2) is not equal 0
An ideal solution

```
OddNumbers
     1- For number =0 to 99 with step =1 do
           1.1 number % 2 != 0
                  1.1.1 Print number
End
```

Figure 3.4: Solving the Odd Numbers Problem Based on The strategy shwon in Figure 3.1

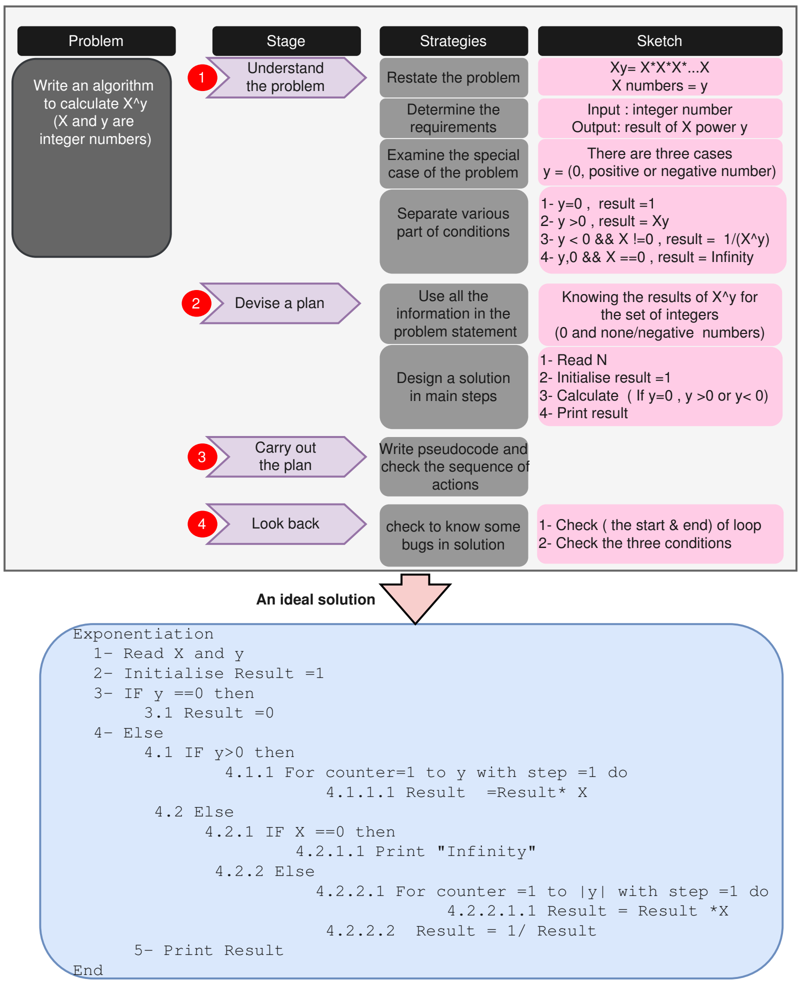
The second pair of problems is to calculate factorial and exponentiation as shown in Figure 3.5 and Figure 3.6. These figures also show the approach to problem-solving. There is more than one ideal solution but we choose one at this stage. However, there are several expected errors from novice programmers could be committed in both examples. Those programmers may not examine all special cases. In this case, the results will be different according to the type of integers whether zero, positive or negative number as in calculating exponentiation. That means the students do not recognize all cases for the problem and that leads to misinterpreting the problem improperly. There is another error which is forgetting some of the main steps such initializing the variable. That leads students do not design what the required steps to solve the problem carefully. However, this error could be fixed very quickly if the user implement these examples in any programming language but this step is very important in carrying out the plan because this variable will be used in the formula that calculate the factorial or exponentiation.



Problem
Stage
Strategies
Sketch
Write an algorithm to calculate N!
1
Understand the problem
Restate the problem
N! = N*(N-1)*(N-2)......2*1
Determine the requirements
Input : integer number
Output: result of the factorial
Examine the special case of the problem
There are three cases
N = (0or positive)
Separate various part of conditions
1- N=0 , result =1
2- N >0 , result = N!
2
Devise a plan
Use all the information in the problem statement
Knowing the results of factorial for the set of integers (0 and positive numbers)
Design a solution in main steps
1- Read N
2- Initialise result =1
3- Calculate (If N=0 or N >0)
4- Print result
3
Carry out the plan
Write pseudocode and check sequence of actions
4
Look back
Check to know some bugs in solution
1- Check (the start & end) of loop
2- Check the two conditions
An ideal solution

```
Factorial
     1- Read N
     2- Initialise Result =1
     3- IF N ==0 then
          3.1 Result =0
     4- Else
          4.1  For counter =1 to N with step =1 do
                 4.2.1 Result = Result * counter
     5- Print Result

 End
```

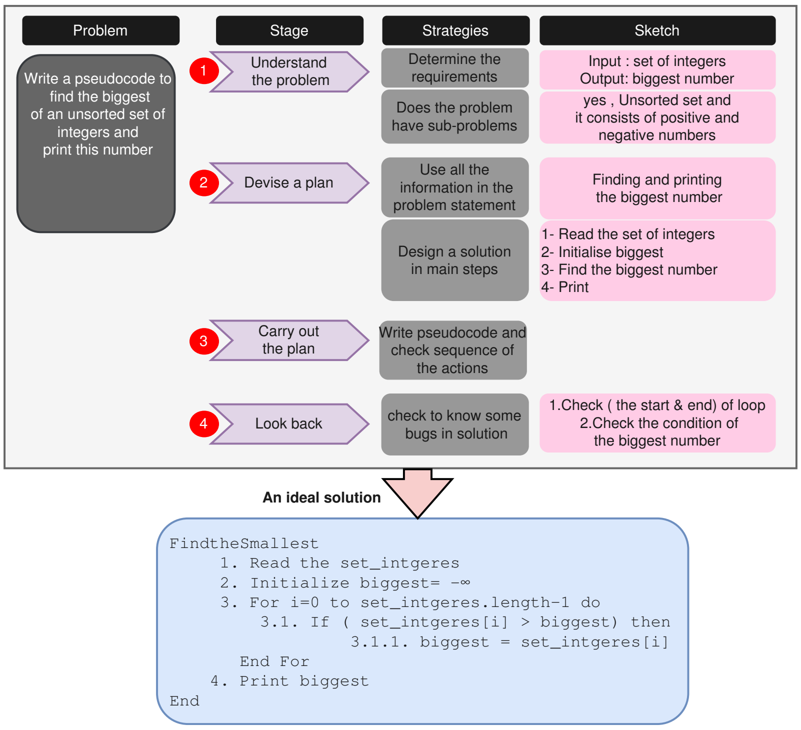
Figure 3.5: Solving the Factorial Problem Based on The strategy shwon in Figure 3.1


Problem
Stage
Strategies
Sketch
Write an algorithm to calculate X^y (X and y are integer numbers)
1
Understand the problem
Restate the problem
Xy= X*X*X*...X
X numbers = y
Determine the requirements
Input : integer number
Output: result of X power y
Examine the special case of the problem
There are three cases
y = (0, positive or negative number)
Separate various part of conditions
1- y=0 , result =1
2- y >0 , result = Xy
3- y < 0 && X !=0 , result = 1/(X^y)
4- y,0 && X ==0 , result = Infinity
2
Devise a plan
Use all the information in the problem statement
Knowing the results of X^y for the set of integers (0 and none/negative numbers)
Design a solution in main steps
1- Read N
2- Initialise result =1
3- Calculate (If y=0 , y >0 or y< 0)
4- Print result
3
Carry out the plan
Write pseudocode and check the sequence of actions
4
Look back
check to know some bugs in solution
1- Check (the start & end) of loop
2- Check the three conditions
An ideal solution

```
Exponentiation
  1- Read X and y
  2- Initialise Result =1
  3- IF y ==0 then
       3.1 Result =0
  4- Else
       4.1 IF y>0 then
               4.1.1 For counter=1 to y with step =1 do
                         4.1.1.1 Result  =Result* X
        4.2 Else
             4.2.1 IF X ==0 then
                      4.2.1.1 Print "Infinity"
              4.2.2 Else
                        4.2.2.1 For counter =1 to |y| with step =1 do
                                     4.2.2.1.1 Result = Result *X
                         4.2.2.2  Result = 1/ Result
      5- Print Result
End
```

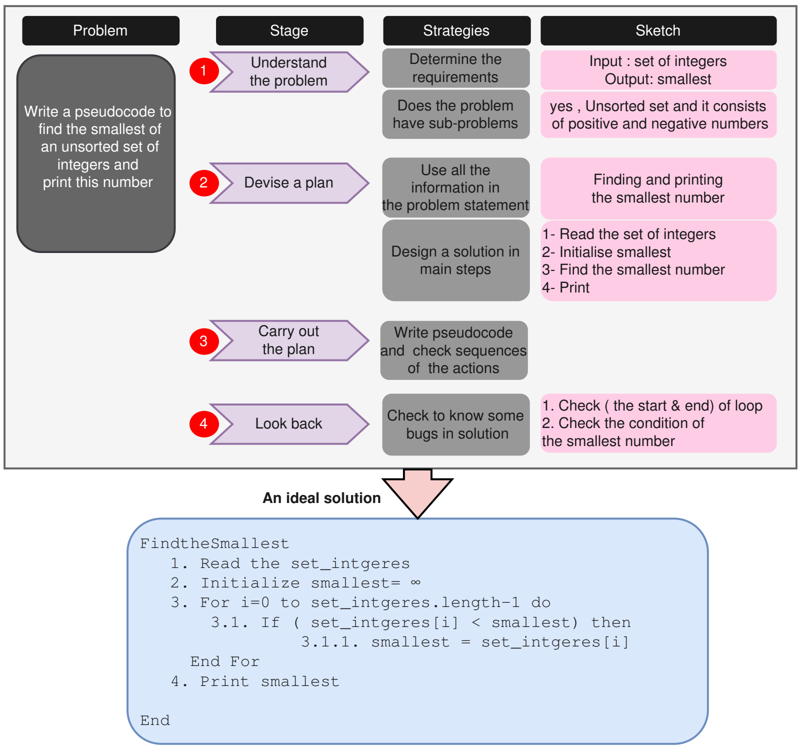
Figure 3.6: Solving the Exponentation Problem Based on The strategy shwon in Figure 3.1

The third pair of problems is to find the smallest and biggest number of unsorted set of integers as shown in Figure 3.7 and Figure 3.8. Similar to the pervious pairs of problem, these figures also show the approach to problem-solving and providing an ideal solution. However, the expected error in these examples could be the value of initializing the number to test it with the list of integer numbers. For example, if the user initializes this variable equal to zero and the list contains these numbers -1, -4, -5, -2 and the problem is to find the biggest number. The expected result is -1 but in this case, the result will be 0 because the wrong way of initialization of the variable. This means students lack of evaluate the solution.


Problem
Stage
Strategies
Sketch
Write a pseudocode to find the biggest of an unsorted set of integers and print this number
1
Understand the problem
Determine the requirements
Input : set of integers Output: biggest number
Does the problem have sub-problems
yes , Unsorted set and it consists of positive and negative numbers
2
Devise a plan
Use all the information in the problem statement
Finding and printing the biggest number
Design a solution in main steps
1- Read the set of integers
2- Initialise biggest
3- Find the biggest number
4- Print
3
Carry out the plan
Write pseudocode and check sequence of the actions
4
Look back
check to know some bugs in solution
1.Check (the start & end) of loop
2.Check the condition of the biggest number
An ideal solution

```
FindtheSmallest
     1. Read the set_intgeres
     2. Initialize biggest= -∞
     3. For i=0 to set_intgeres.length-1 do
         3.1. If ( set_intgeres[i] > biggest) then
                  3.1.1. biggest = set_intgeres[i]
       End For
    4. Print biggest
End
```

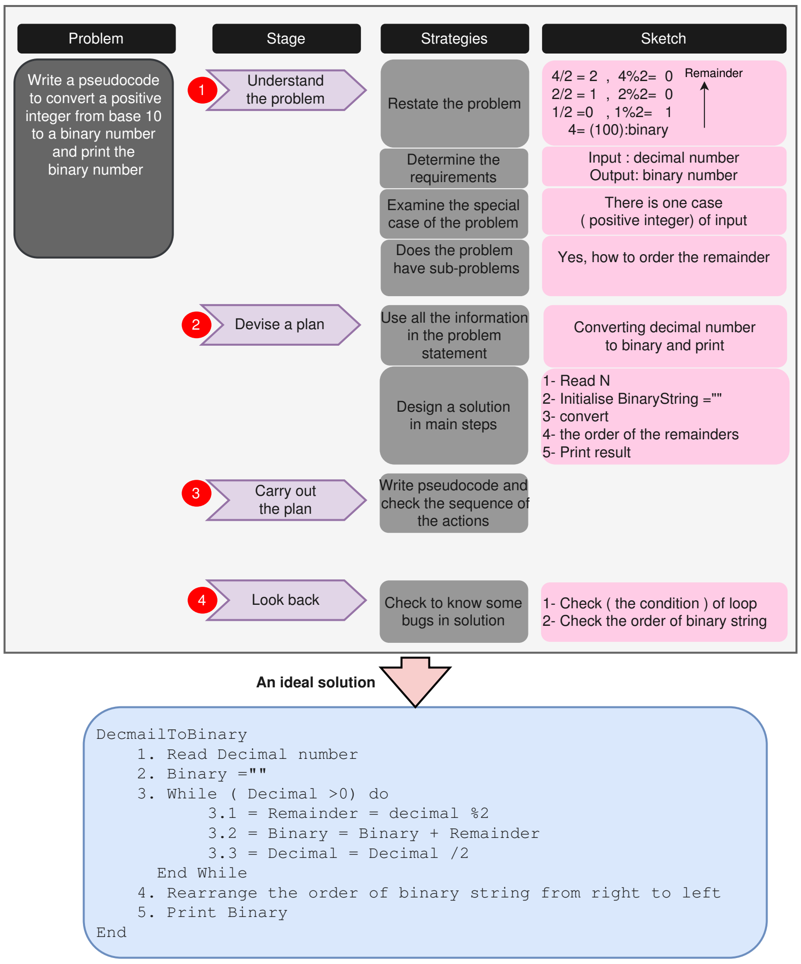
Figure 3.7: Finding the Biggest Number Problem Based on The strategy shwon in Figure 3.1


Problem
Stage
Strategies
Sketch
Write a pseudocode to find the smallest of an unsorted set of integers and print this number
1
Understand the problem
Determine the requirements
Input : set of integers Output: smallest
Does the problem have sub-problems
yes , Unsorted set and it consists of positive and negative numbers
2
Devise a plan
Use all the information in the problem statement
Finding and printing the smallest number
Design a solution in main steps
1- Read the set of integers
2- Initialise smallest
3- Find the smallest number
4- Print
3
Carry out the plan
Write pseudocode and check sequences of the actions
4
Look back
Check to know some bugs in solution
1. Check (the start & end) of loop
2. Check the condition of the smallest number
An ideal solution

```
FindtheSmallest
   1. Read the set_intgeres
   2. Initialize smallest= ∞
   3. For i=0 to set_intgeres.length-1 do
       3.1. If ( set_intgeres[i] < smallest) then
                3.1.1. smallest = set_intgeres[i]
     End For
   4. Print smallest

End
```

Figure 3.8: Finding the Smallest Number Problem Based on The strategy shwon in Figure 3.1

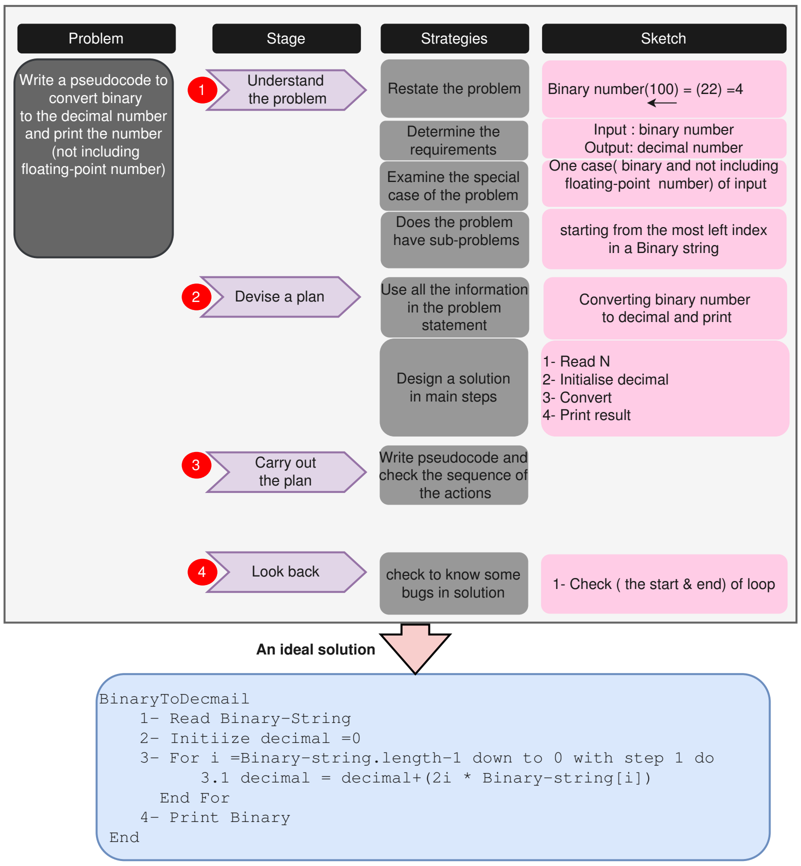
The next pair of problems is to convert decimal to binary numbers and vice versa as shown in Figure 3.9 and Figure 3.10. In terms of expected errors for both problems is similar to the pervious examples. However, these problems have other sub-problems. For example, to convert decimal to binary number, we need to rearrange the order of remainders after getting them from right to left. Similarly, there is a sub-problem in converting binary to decimal number problem. It is starting from the most left index of Binary String. If the student does not recognize them, that means the student lack of understanding the problem and determining the steps required to solve the problem.



Problem
Stage
Strategies
Sketch
Write a pseudocode to convert a positive integer from base 10 to a binary number and print the binary number
1
Understand the problem
Restate the problem
4/2 = 2 , 4%2= 0
2/2 = 1 , 2%2= 0
1/2 =0 , 1%2= 1
4= (100):binary
Remainder
Determine the requirements
Input : decimal number
Output: binary number
Examine the special case of the problem
There is one case (positive integer) of input
Does the problem have sub-problems
Yes, how to order the remainder
2
Devise a plan
Use all the information in the problem statement
Converting decimal number to binary and print
Design a solution in main steps
1- Read N
2- Initialise BinaryString =""
3- convert
4- the order of the remainders
5- Print result
3
Carry out the plan
Write pseudocode and check the sequence of the actions
4
Look back
Check to know some bugs in solution
1- Check (the condition) of loop
2- Check the order of binary string
An ideal solution

```
DecmailToBinary
    1. Read Decimal number
    2. Binary =""
    3. While ( Decimal >0) do
           3.1 = Remainder = decimal %2
           3.2 = Binary = Binary + Remainder
           3.3 = Decimal = Decimal /2
      End While
    4. Rearrange the order of binary string from right to left
    5. Print Binary
End
```

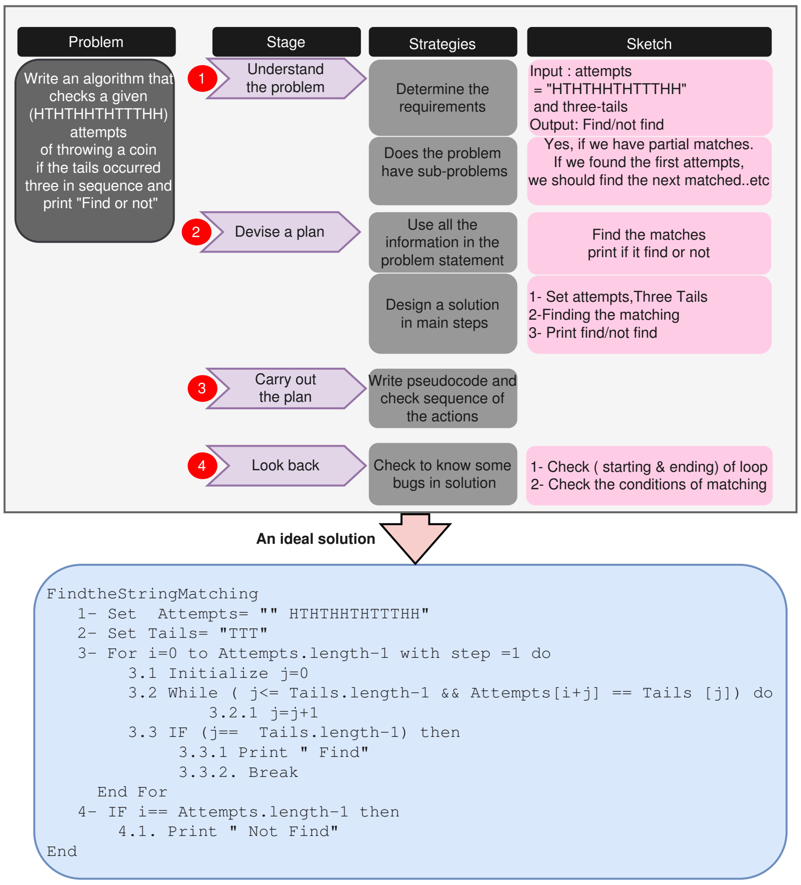
Figure 3.9: Converting From Decimal to Binary Based on The strategy shwon in Figure 3.1


Problem
Stage
Strategies
Sketch
Write a pseudocode to convert binary to the decimal number and print the number (not including floating-point number)
1
Understand the problem
Restate the problem
Binary number(100) = (22) =4
Determine the requirements
Input : binary number
Output: decimal number
Examine the special case of the problem
One case(binary and not including floating-point number) of input
Does the problem have sub-problems
starting from the most left index in a Binary string
2
Devise a plan
Use all the information in the problem statement
Converting binary number to decimal and print
Design a solution in main steps
1- Read N
2- Initialise decimal
3- Convert
4- Print result
3
Carry out the plan
Write pseudocode and check the sequence of the actions
4
Look back
check to know some bugs in solution
1- Check (the start & end) of loop
An ideal solution

```
BinaryToDecmail
    1- Read Binary-String
    2- Initiize decimal =0
    3- For i =Binary-string.length-1 down to 0 with step 1 do
          3.1 decimal = decimal+(2i * Binary-string[i])
      End For
    4- Print Binary
 End
```

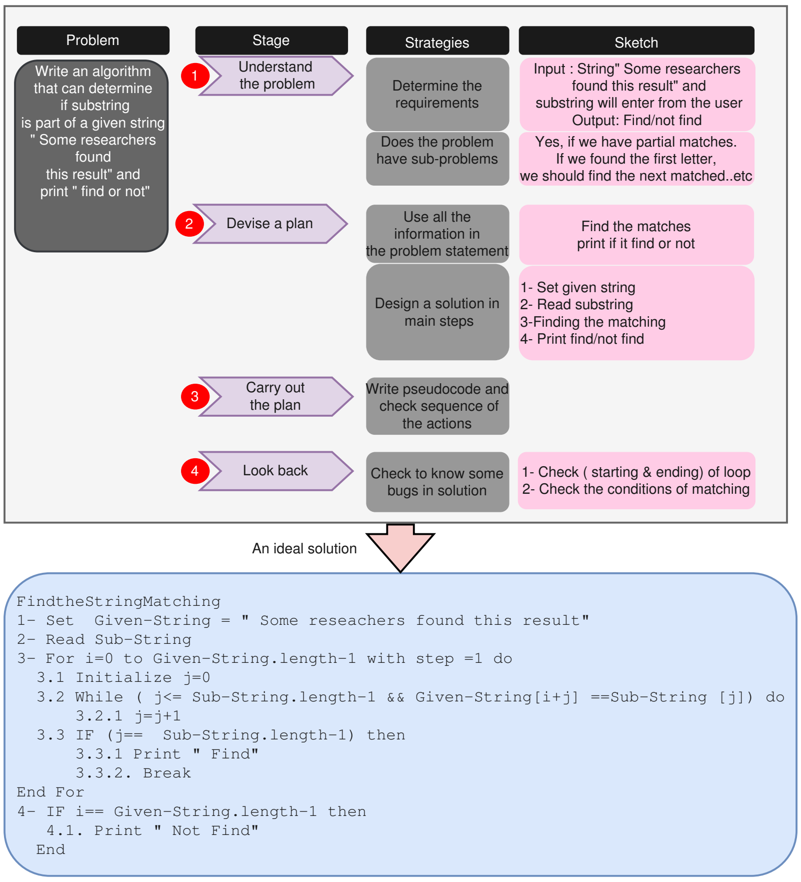
Figure 3.10: Converting From Binary to Decimal Based on The strategy shwon in Figure 3.1

The solutions of the next problems are very similar as shown in Figure 3.11 and Figure 3.12. However, the difference is only the value of initializing the input of the problem. The challenges for these problems is how to use nested loops with finding all valid shifts with which a given substring occurs in a given sentence. Furthermore, there are expected errors in both problem. One of those errors is the inconsistent checking between the sentence and a sub-string. That means the student lack of determining the steps required to solve the problem and Lack of verify whether solution is correct.


Problem
Stage
Strategies
Sketch
Write an algorithm that checks a given (HTHTHHTHTTTHH) attempts of throwing a coin if the tails occurred three in sequence and print "Find or not"
1
Understand the problem
Determine the requirements
Input : attempts = "HTHTHHTHTTTHH" and three-tails Output: Find/not find
Does the problem have sub-problems
Yes, if we have partial matches. If we found the first attempts, we should find the next matched..etc
2
Devise a plan
Use all the information in the problem statement
Find the matches print if it find or not
Design a solution in main steps
1- Set attempts,Three Tails
2-Finding the matching
3- Print find/not find
3
Carry out the plan
Write pseudocode and check sequence of the actions
4
Look back
Check to know some bugs in solution
1- Check (starting & ending) of loop
2- Check the conditions of matching
An ideal solution

```
FindtheStringMatching
   1- Set  Attempts= "" HTHTHHTHTTTHH"
   2- Set Tails= "TTT"
   3- For i=0 to Attempts.length-1 with step =1 do
        3.1 Initialize j=0
        3.2 While ( j<= Tails.length-1 && Attempts[i+j] == Tails [j]) do
                3.2.1 j=j+1
        3.3 IF (j==  Tails.length-1) then
             3.3.1 Print " Find"
             3.3.2. Break
     End For
   4- IF i== Attempts.length-1 then
       4.1. Print " Not Find"
End
```

Figure 3.11: Find a Sub-string in a String Problem Based On The strategy shwon in Figure 3.1


Problem
Stage
Strategies
Sketch
Write an algorithm that can determine if substring is part of a given string " Some researchers found this result" and print " find or not"
1
Understand the problem
Determine the requirements
Input : String" Some researchers found this result" and substring will enter from the user Output: Find/not find
Does the problem have sub-problems
Yes, if we have partial matches. If we found the first letter, we should find the next matched..etc
2
Devise a plan
Use all the information in the problem statement
Find the matches print if it find or not
Design a solution in main steps
1- Set given string
2- Read substring
3-Finding the matching
4- Print find/not find
3
Carry out the plan
Write pseudocode and check sequence of the actions
4
Look back
Check to know some bugs in solution
1- Check (starting & ending) of loop
2- Check the conditions of matching
An ideal solution

```
FindtheStringMatching
1- Set  Given-String = " Some reseachers found this result"
2- Read Sub-String
3- For i=0 to Given-String.length-1 with step =1 do
  3.1 Initialize j=0
  3.2 While ( j<= Sub-String.length-1 && Given-String[i+j] ==Sub-String [j]) do
      3.2.1 j=j+1
  3.3 IF (j==  Sub-String.length-1) then
      3.3.1 Print " Find"
      3.3.2. Break
End For
4- IF i== Given-String.length-1 then
   4.1. Print " Not Find"
  End
```

Figure 3.12: Finding Three Tails Attempts Problem Based on The strategy shwon in Figure 3.1

3.3 Designing our System

3.3.1 Designing the virtual Agents: Amy and Ms. Sarah

We designed and developed two virtual agents that could interact with the user in the interest of providing interactive learning environment. The first agent takes the role of an active student which is called Amy as shown in Figure 3.13 . As mentioned previously, she can learn from the user about problem solving strategies. Also, she can provide explanations about what she understood and ask the user to confirm about her explanation. For example, if the user teaches her correctly about how to convert a binary number to a decimal number. She might say "Thank you for teaching me. Now I know about how to convert binary to decimal number. start with the binary numbers and multiply each binary digit by its weight. Then after multiplying all the digits, I will add the results. Is my explanation correct?". In some cases, she provides feedback to the user.

The second agent is a mentor agent which is called Ms. Sarah as shown in Figure 3.14. She can provide guidance and suggestions. Clearly, she grades Amy's quizzes and provides feedback to the user and Amy related to the task. She also evaluates the user's strategies and allows the user to ask her an advice.

Figure 3.13: The Virtual Agent

Figure 3.14: The Virtual Agent Ms. Sarah

3.3.2 Describing the Processes of the Proposed System

In general, we will divide our teachable agents environment to three stages. The first stage is the teaching process. That enables the user to teach problem solving to a virtual agent which called "Amy". The problem will be provided from the system. Amy takes the role of an active student. During this stage, the student will take the role of a teacher to teach the agent Amy. Amy will record what the user teaches her about problem solving strategies. Also, Amy can ask questions if she did not understand.

After finish this stage, the user can request Amy to take a quiz to ensure the quality of teaching problem. In the quiz stage, Amy can take the quiz for another problem. This problem is near-isomorphic to the first problem taught by the user. Then, she will apply what she understood from the user. For example, if the user teaches her correctly, Amy will perform better, and her solution will be like an expert solution. In contracts, if the user did not teach her correctly in the first problem. Amy will perform based on the knowledge passed by the user, and in this case Amy most likely will not provide a correct solution.

The third stage is providing feedback about the performance of the agent Amy. After finishing the quiz stage, Ms. Sarah (she is a mentor agent) will provide metacognitive feedback list to the user and Amy. Furthermore, Ms. Sarah will require from the user to re-teach Amy about her mistakes or the incomplete tasks for the second problem. Then, user will start to re-teach Amy again in order to make Amy able to provide a correct solution to the problem. If the user re-teaches each error from the feedback list correctly, then Amy will explain her understanding and asking user to confirm her explanation. If user say "Yes", Amy will ask for help on the next error from Ms. Sarah list. Conversely, if the user say "No", the system will allow the user to write his explanation on that error and so forth. Ms. Sarah (the mentor agent) can provide advises and some hints about the targeted problem in order to remind the user of the required steps to solve any problem

3.3.3 Data Flow Diagram of the Proposed System

We design data flow diagram as illustrated in Figure 3.15. It describes system's high-level modeling, and it shows sets of the processes, data stores and dataflow of the system and how they are linked to each other. In the beginning, the user will teach the virtual agent Amy the first problem. Then, the user strategies for solving the problem will be stored in D1. Amy has learned the user strategies which used in the first problem. Then she will use the same strategies to solve the second problem. After finishing the second problem, Ms. Sarah will evaluate the solution of the second problem and she will provide metacognitive feedback to the user and Amy. After receiving the feedback, in case of incorrect solution, the user will teach Amy again about the second problem. In addition to that, if the user teaches Amy correctly, Amy will present her explanations and ask confirmation from the user. Then, if the user agrees with Amy's explanation, these new strategies acquired will be stored in D2. In contrast, if the user disagrees with Amy's explanation, the system will allow the user to write his own explanation and it will be stored in D

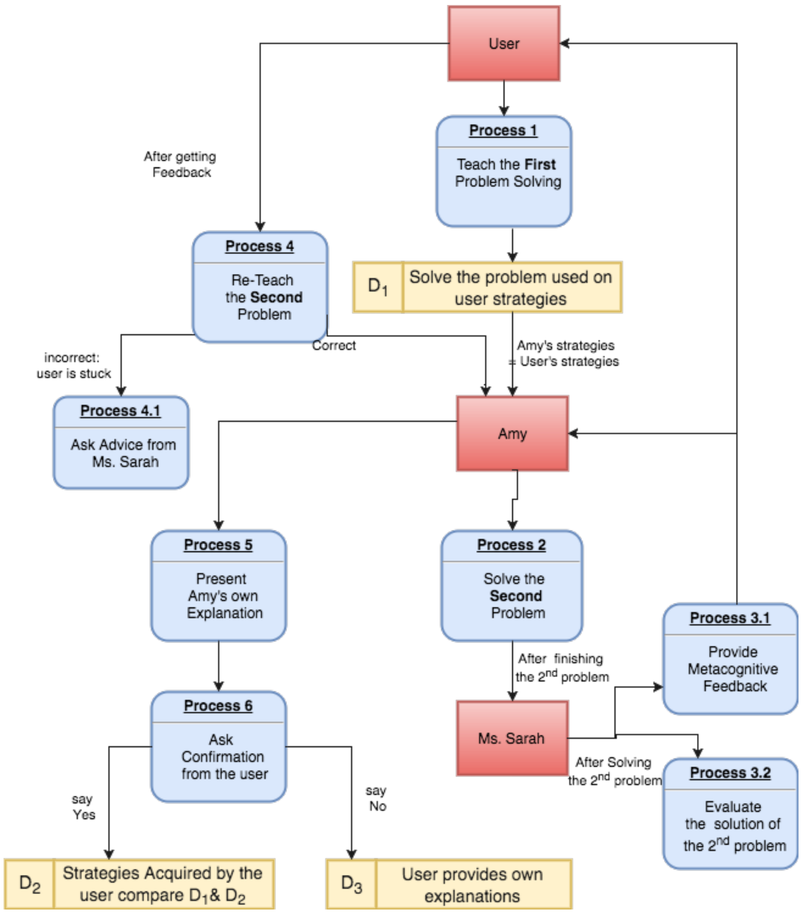

User
Process 1
Teach the First Problem Solving
After getting Feedback
Process 4
Re-Teach the Second Problem
D1
Solve the problem used on user strategies
incorrect: user is stuck
Correct
Amy's strategies ≠ User's strategies
Process 4.1
Ask Advice from Ms. Sarah
Amy
Process 5
Present Amy's own Explanation
Process 2
Solve the Second Problem
Process 3.1
Provide Metacognitive Feedback
After finishing the 2nd problem
Process 6
Ask Confirmation from the user
Ms. Sarah
After Solving the 2nd problem
Process 3.2
Evaluate the solution of the 2nd problem
say Yes
say No
D2
Strategies Acquired by the user compare D1& D2
D3
User provides own explanations

Figure 3.15: Data Flow Digram of the Proposed System

3.3.4 Architectural Design of the Proposed System

We translated all the requirements of the teachable agents' environment into the architectural design as shown in Figure 3.16. As mentioned previously, the system has two animated agents which Amy and Ms. Sarah to interact with the student in order to provide interactive learning environment. Also, there are two other agents will be used in the system to handle data relevant to the specified functionalities. These agents are the Tracker, and Interfacer. The main purpose of these Agents is to interact with the animated agents when some events are generated by these animated agents and the user.

- Amy

To make an efficient learning-by-teaching environment, the agent Amy needs to demonstrate qualities of an active student. In order to be an active and good student, she will provide an explanation about her understanding after the user teaches her again about the mistakes or missing tasks. For that, Tutors (The users) obtain profound understanding from interactions with tutee (Amy) that includes explaining and discovering misconceptions [72] [52].

- Ms. Sarah

Ms. Sarah is the mentor agent which is an expert about the domain knowledge in problem-solving. She will provide feedback in order to help the user to improve their metacognitive strategies. In addition, she will evaluate Amy's solution and she also will ask the user to help Amy for providing a correct solution.

- Tracker

The tracker uses an automated approach to determine the user's progress and notify other agents Amy and Ms. Sarah. For example, if the user requested Amy to take a quiz without teaching her, the tracker detects that and trigger an event to inform Amy. Another example, if the user spent longer time to teach Amy, in this case another type of trigger will be issued to Ms. Sarah to provide some helps.

- Interfacer

Interfacer acts as a broker between all the system structures and the reasoning mechanism and the Graphical User Interface (GUI). In the beginning, the user will interact with the GUI. Then, Interfacer will display the updated view in the GUI. When Amy and Ms. Sarah need to interact with the user as a composing message (i.e. feedback or explanation). The Interfacer will display that message to the user.

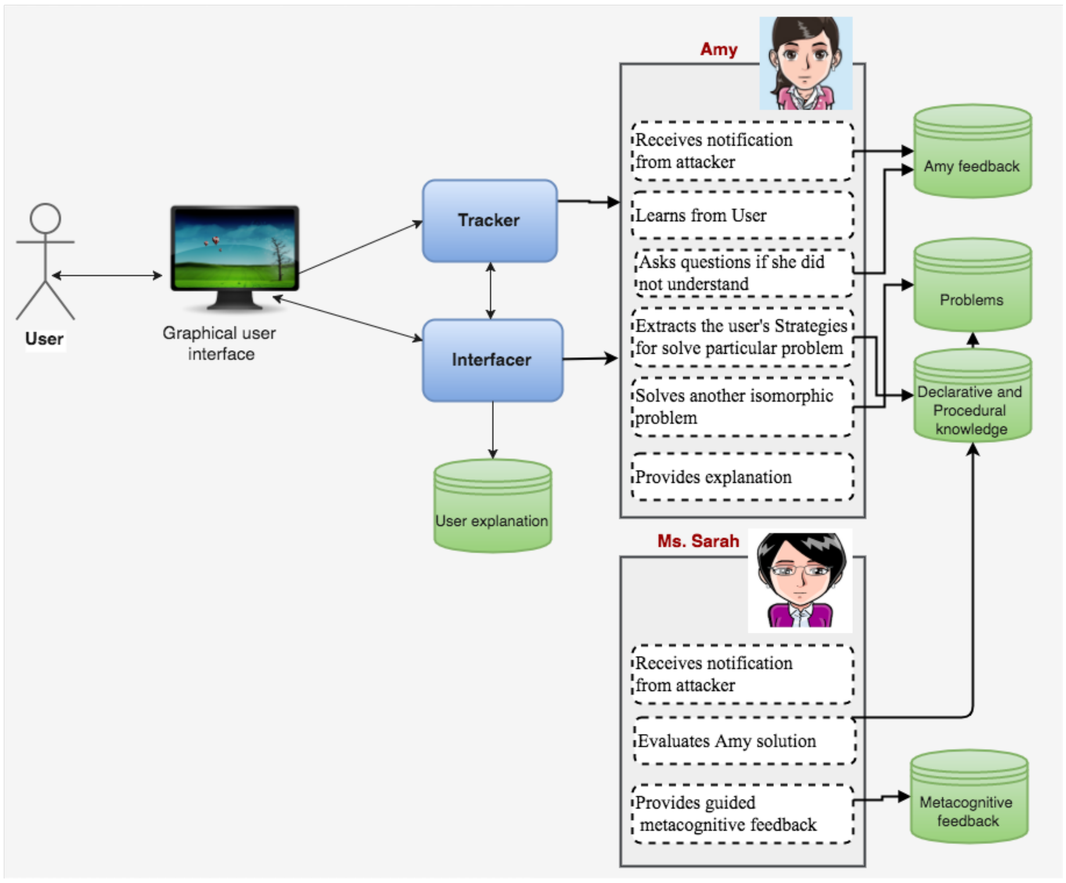

Amy
Receives notification from attacker
Learns from User
Asks questions if she did not understand
Extracts the user's Strategies for solve particular problem
Solves another isomorphic problem
Provides explanation
Amy feedback
Problems
Declarative and Procedural knowledge
User
Graphical user interface
Tracker
Interfacer
User explanation
Ms. Sarah
Receives notification from attacker
Evaluates Amy solution
Provides guided metacognitive feedback
Metacognitive feedback

Figure 3.16: Architectural design of the Proposed System

3.3.5 Feedback Design

In general, the temporary assistance from a teacher helps students to complete the tasks efficiently. Vygotsky [40] defined this area as the zone of proximal development. Due to the targeted users of our system will be the novice programmers, they surely need feedback and guidance to help them to progress through solving the programming problems. Providing feedback to the user about their performance and progress helps them to complete the tasks effectively and interactively [43].

One of the main goal of this system is to provide guided metacognitive feedback. Ms. Sarah has the responsibility to do this task. This feedback is organized as successive hints that provide the answer to the current problem step. Ms. Sarch feedback contains two stages.

Firstly, she provides clues about the reasons for the incorrect answer of Amy. Secondly, she suggests specific activities to the user in order to help Amy. Table 3.2 shows some examples of Amy's behaviour and what the feedback will be provided from Ms. Sarah. Furthermore, if the user could not help Amy about the feedback that means the user needs more information about the feedback. For that, Ms. Sarah will provide hints as next level of the feedback to assist the user in order to teach Amy again. For instance, if Amy missed some of main processes such as initializing a variable, the user read this message "Amy's answer is not complete. There is one of the main steps is missing. Please review her process and find this step". If the user still cannot help Amy. Ms. Sarah will provide corrective feedback, she may say "The result variable was not initialized".

Table 3.2: Some Feedback Example From Ms. Sarah

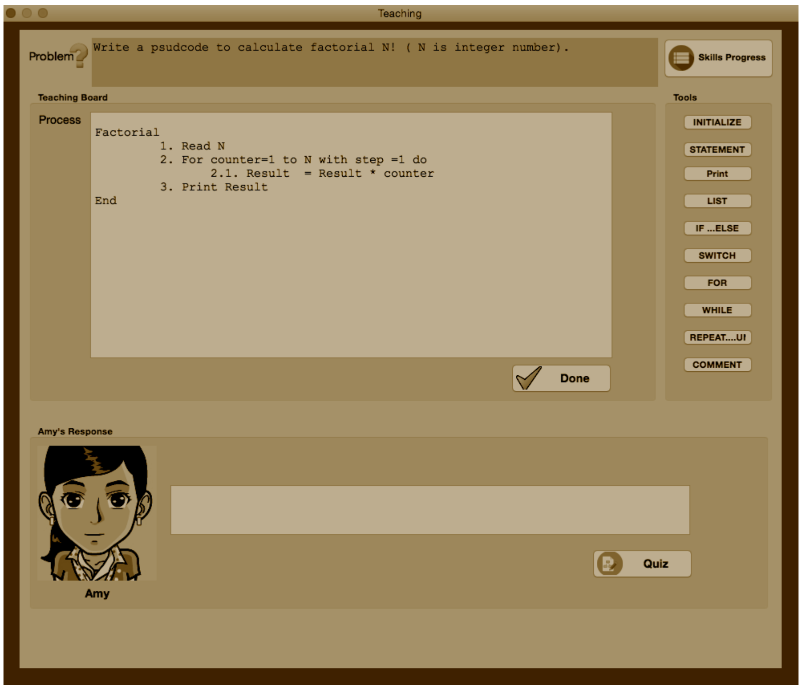
No	Amy's behaviour	Agent Feedback
1	Did not know the subject such as: how to convert binary to decimal	Amy's answer is wrong because she does not know how to convert binary number to decimal number. You should read about it and then re-teach her this information.
2	Forgot some special case such as 0!	Amy's answer is incomplete because this problem has more than one cases. Please read the problem statement and think all these cases and then help her.
3	Missed some sub-problems : such as how to order the remainder in convert decimal to binary	Amy's answer is not complete because there is sub-problem missing which is order the remainder. You should identify the sub problem and then re-teach her this part.
4	Did not determine the requirement like: read input in Factorial problem	Amy does not determine the input of algorithm. Could you please determine it?
5	Did separate various part of conditions. Such as x^y if y >0 or y <0 has another solution	Amy's answer is incomplete because this problem has more than one conditions. Please help her to find two parts of conditions.
6	Missed some information in the problem statement such as forget test if number is even or not	Amy's answer is incomplete because she does not use all information on the problem statement. Please read the problem statement and find missing information and then help her
7	Did not break down the problem as (step by step) such as in matching string no using two pointers	Amy does not know how to break down the problem correctly there is something missing. Please help her to break down the problem correctly.
8	Missed one of main process such as initialization	Amy's answer is not complete. There is one of the main steps is missing. Please review her process and find this step.
9	Incorrect ordering of action such as Result = L*W Read L*W	Amy's answer is wrong because the ordering of action is not correct. You should help Amy to order the actions correctly
10	Incorrect starting or ending of iteration.	Amy's answer is wrong because there is a logical error in Iteration. Please help her to find the bug(s)
11	Incorrect condition expression such as >and >=	Amy's answer is wrong because there is a logical error in condition. Please help her to find the bug(s)
12	The value of Initialization is wrong	Amy's answer is not right due to the value of Initialization. Please help her to fix it.
13	Using incorrect operator precedence such as x= a*b+c x= a*(b+c)	Amy's answer is not right due to using incorrect operator precedence. Please help her to fix it.

There is another feedback in the system which is provided by Amy. This feedback is inquisitive rather than explicit instruction for solving the problem. She focuses only on the interactions between herself and the user such as ineffective behaviors and making corrective suggestions. For example, if the user does not teach Amy and wants to request her to take a quiz. Amy might say *"Excuse me. You have not taught me anything. Please teach me before you send me to take the quiz"*. Appendix L shows more examples about Amy's feedback.

3.3.6 Prototype of The System

To design a our Teachable Agent learning environments, we design the user interface to be friendly interface. That means the User Interface should allow the user to move among all phases in seamless way.

The graphical user interface consists of two main interfaces teaching and quiz interfaces. The teaching interface as shown in Figure 3.17 has some particular tools that helps users to generate pseudo-code. These tools are located in the right side of the interface. On the top of the window, the problem statement will be provided from the system to user. On the middle, the user can write the pseudo-code by using the tools in the right side along with writing the required text. In the bottom of the interface, Amy agent is located along with a part where Amy could provide her feedback to the user. On the lower right corner of the interface, there is a quiz button which enables the user to send Amy to take a quiz. Also, the upper right corner, there a skills progress button that the user can click to see Amy's overall progress as shown in Figure 3.18.



Write a psudcode to calculate factorial N! (N is integer number).

```
Factorial
        1. Read N
        2. For counter=1 to N with step =1 do
               2.1. Result  = Result * counter
        3. Print Result
End
```

Figure 3.17: The Teaching Interface

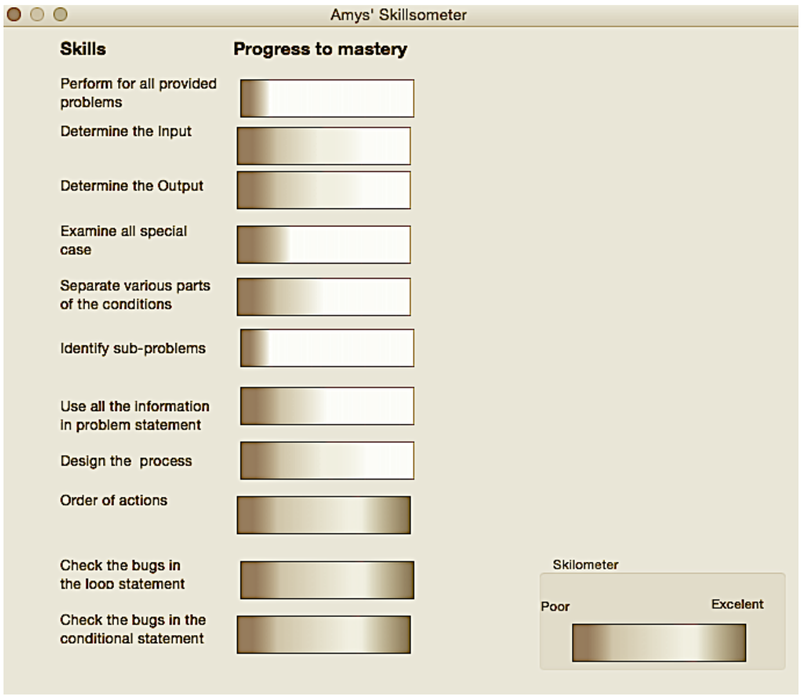
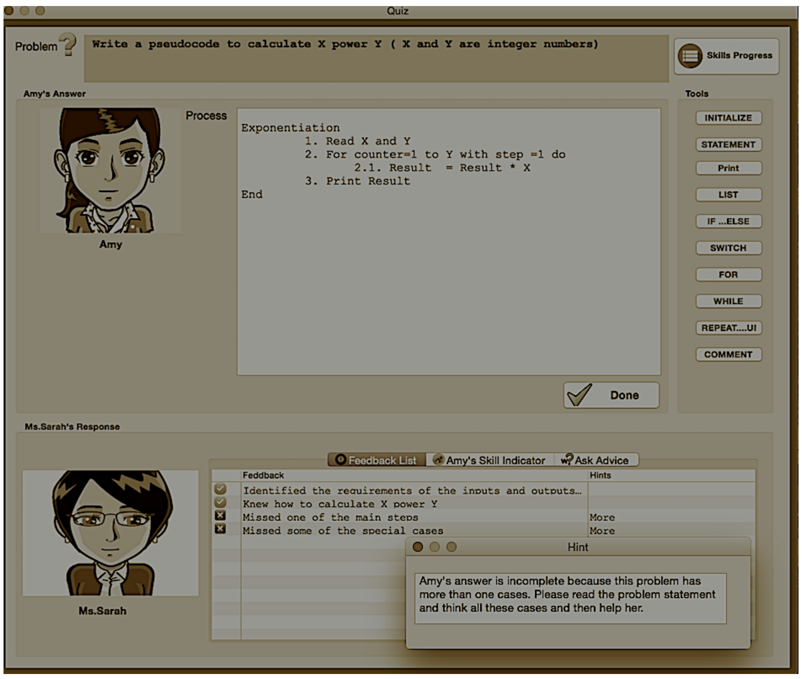

Amys' Skillsometer
Skills
Progress to mastery
Perform for all provided problems
Determine the Input
Determine the Output
Examine all special case
Separate various parts of the conditions
Identify sub-problems
Use all the information in problem statement
Design the process
Order of actions
Check the bugs in the loop statement
Check the bugs in the conditional statement
Skilometer
Poor
Excelent

Figure 3.18: The Progress of the Skills Window

Quiz interface is the second interface in our system. This interface has many features in common with the Teaching Interface. However, it includes extra features such as the feedback list provided by Ms. Sarah as shown in Figure 3.19. The bottom part (Ms. Sarach response) of the interface is the most interesting part in this interface. It has three components. The first component is feedback list. It displays the feedback and the situation of this feedback if it is corrected or not. Also, it provides explanations of this feedback if user click on the word "More" as shown in Figure 3.19. The second component is Amy's skill indictor that shows the accuracy measurements for the current problem as shown in Figure 3.20. The third component is "Ask advice".



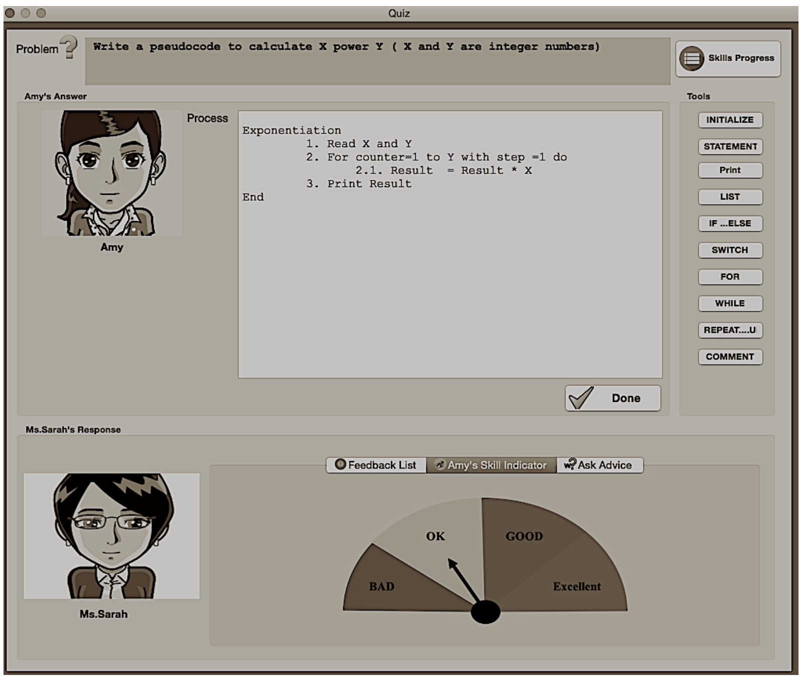
Write a pseudocode to calculate X power Y (X and Y are integer numbers)

```
Exponentiation
        1. Read X and Y
        2. For counter=1 to Y with step =1 do
              2.1. Result  = Result * X
        3. Print Result
End
```

	Feddback	Hints
✓	Identified the requirements of the inputs and outputs…	
✓	Knew how to calculate X power Y	
☒	Missed one of the main steps	More
☒	Missed some of the special cases	More

Amy's answer is incomplete because this problem has more than one cases. Please read the problem statement and think all these cases and then help her.

Figure 3.19: Some Hits from Ms. Sarah to Show the Issues of the Solution

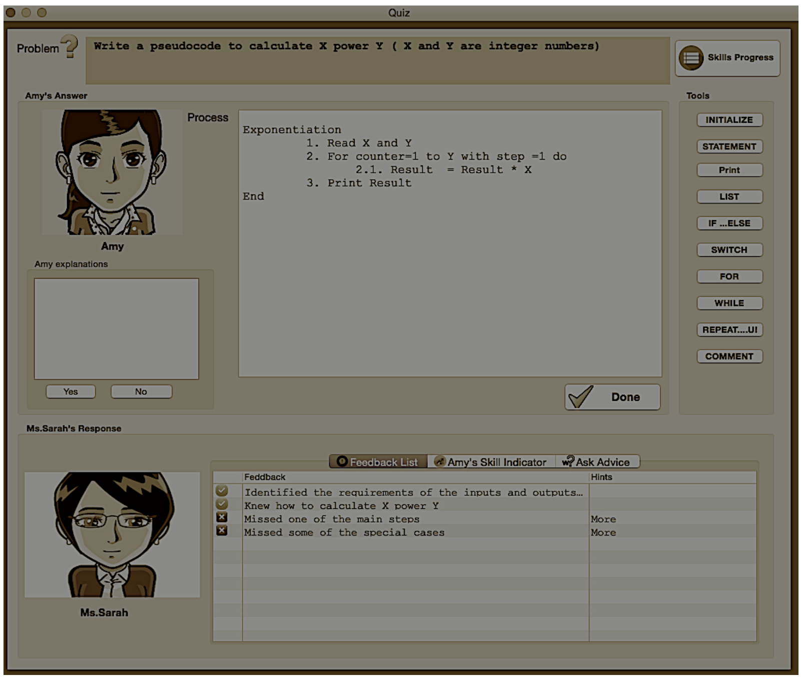


Write a pseudocode to calculate X power Y (X and Y are integer numbers)

```
Exponentiation
        1. Read X and Y
        2. For counter=1 to Y with step =1 do
              2.1. Result  = Result * X
        3. Print Result
End
```

Figure 3.20: Amy's Skill Indicator for a Particular Problem

Furthermore, there is a hidden feature in this interface, and it is located under the agent Amy as shown in Figure 3.21. It comes up after the user re-teaches Amy correctly about one of the feedback of Ms. Sarah. This feature is a place where Amy can provide an explanation on how she understood the user, and she will ask the user to confirm her explanation.



Write a pseudocode to calculate X power Y (X and Y are integer numbers)

```
Exponentiation
        1. Read X and Y
        2. For counter=1 to Y with step =1 do
              2.1. Result  = Result * X
        3. Print Result
End
```

	Feddback	Hints
✓	Identified the requirements of the inputs and outputs...	
✓	Knew how to calculate X power Y	
☒	Missed one of the main steps	More
☒	Missed some of the special cases	More

Figure 3.21: Amy's Explanation

3.4 A Scenario of How the System Works:

To give a clear overview of how the system works, we are going to present a scenario and go through it step by step. In beginning, if the user uses this system as the first time, the system requires his login information as shown in Figure 3.22. Then, the user will be provided a problem by the system. The problem is for example "write a pseudocode to calculate N!". The user will teach the agent and solve the problem. After finishing, the user will click the button "Done" and then the user can click the button "Quiz" as shown in Figure 3.19. However, if he does not send Amy to take the quiz after finishing the teaching. She will provide feedback saying "Thank for teaching me, you can send me to take quiz".

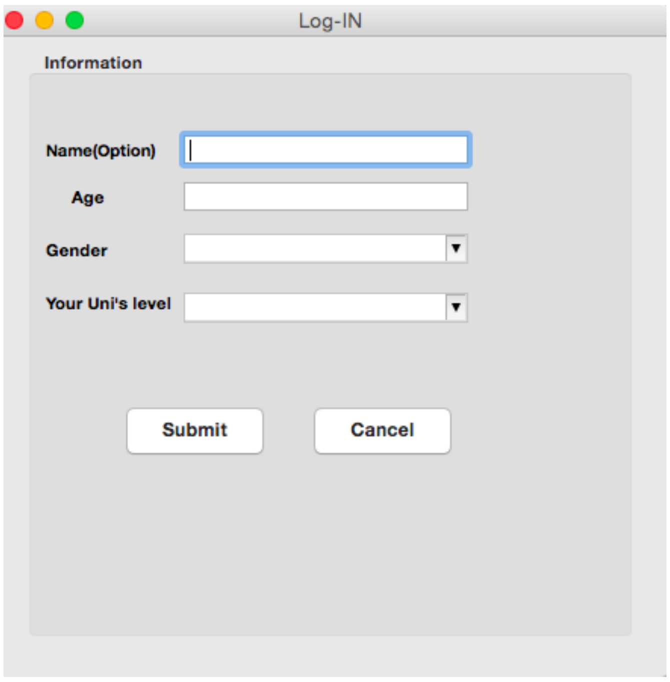


Figure 3.22: Login Window

Then, Amy will take the quiz for a similar problem such as "write a pseudocode to calculate X^y". Amy will solve this problem by herself with using the same strategies the she was taught by the user. After finishing, Ms. Sarah provides feedback to the user and Amy about her solution as shown in Figure **??**. In case is there are errors, the user will teach Amy again about this problem based on the feedback provide by Ms. Sarah. For example, if the user wants to fix the Amy' solution if she missed some of special cases for instance. The user can read more information about this feedback after clicking the word "More" as shown in Figure 3.19. However, if the user does not show any progress. Ms. Sarah will provide more hints about the feedback. She may say "Amy does not deal with the case y is less than zero". However, if the user teaches Amy correctly, she will provide an explanation about what she understood. She may say" Thank you for helping me. Now I know this problem has three special cases when y is positive number ,negative number and zero. Then, she asks for confirmation from the user about her explanation. She may say "Is my explanation correct?".

3.5 Tools Used to Develop the System

The proposed system is implemented in Java (java SE-1.8). In addition, we use the Standard Widget Toolkit (SWT). It is an open source widget toolkit for Java. This toolkit helps to design the Graphic User Interface.

Because of dealing with pseudocode (which is informal high-level language), we will need to use some tools that support natural language processing. There are many tools available with java SE SDK that support natural language processing. Low-level Java has String libraries that includes String, StringBuffer, StringTokenizer, StringBuilder classes. These classes have several methods that perform matching, text replacement, and searching. Furthermore, java supports regular expressions and provides many techniques to use regular expressions.

Chapter 4

Implementation

As we know, the ultimate aim of the educational systems is to facilitate student's learning and to stimulate their learning interest. For that, we designed effective educational interventions as discussed in the last chapter. In this chapter, we will discuss the implementation of the proposed system. This chapter is organized into the following sections. Firstly, it gives an overview of the activities in the proposed system. As well, it presents the introductory explanation of the modules in the system with providing the screen-shots of these activities. After that, the components of the implementation process are described in details.

4.1 Overview of Activities of the Proposed System

As mentioned previously, the ultimate goal of all the educational systems is to facilitate student's learning and to stimulate their learning interest. In general, the teaching process enhances students to think deeply, orally and repeatedly. There are some recommendations to identify the fundamental requirements of Teachable Agents (TA) systems which are summarized as follows: first, the TA system should at the very beginning give students enough hints to help them to know what they need to teach and what the next task that they need to achieve. This may help students to immerse themselves into the teaching role effectively. Second, the TA system should have an explicit representation of the appropriate knowledge, which can help students clearly represent their knowledge to the agent and reciprocally understand the TA's reasoning process. Additionally, the TA system should provide diverse feedback to students throughout the teaching process in order to improve their performance. The feedback may enhance student's reflection and influence their self-monitoring and self-evaluation.

We have designed our system's modules (as listed below) in order to achieve the aforementioned goals. Some of these activities are done by the human student (the tutor). Other activities are provided from our system and the agents (Amy and Ms. Sarah) to the tutor. The interaction with our proposed system follows a sequence of activities whose emphasis is on problem-solving skills and metacognitive support. These activities are:

1. **Module 1:** Selection of the problems - students can use it to select a specific problem to be taught to the TA (Amy).
2. **Module 2:** Teaching workspace - A workspace to teach the TA (Amy)

3. **Module 3:** Amy's feedback and her queries - this module is to present Amy's feedback or her queries to her tutor.

4. **Module 4:** Quiz workspace - this module is to enable Amy to take a quiz based on the knowledge she gathered from her tutor in the teaching module.

5. **Module 5:** Ms. Sarah's Feedback - this module is to grade and evaluate what Amy does in the quiz. Ms. Sarah also provides guided metacognitive feedbacks to Amy if there are some mistakes in her solution.

6. **Module 6:** Re-teaching - based on the feedback explained (Module 5), the tutor can re-teach Amy again based on these hints in order to get the ideal answer.

7. **Module 7:** Explanation - this module enables Amy to provide an explanation about what she understands from the tutor after fixing her mistakes.

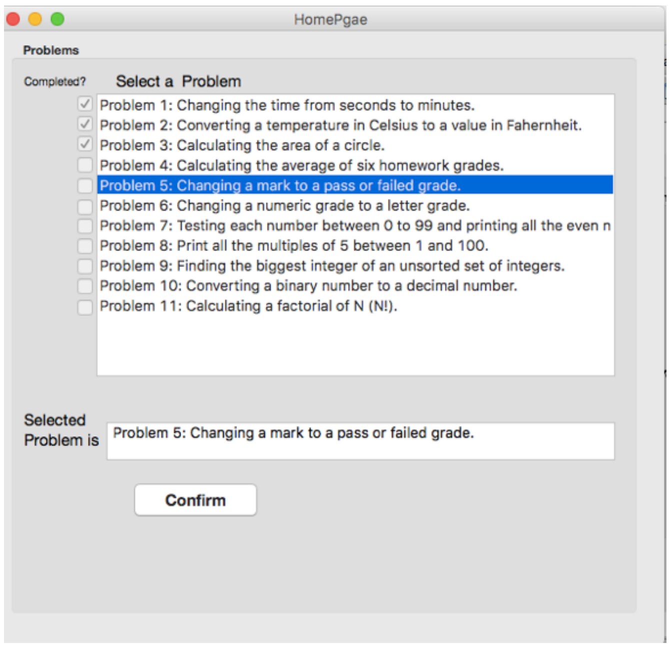
8. **Module 8:** Indicator - this module enables to show the quality of the solution of the current problem.

9. **Module 9:** Skillmeter - this module is to display Amy's overall progress in skills.

10. **Module 10:** Previous answers - this module is to present what the tutor and Amy solutions for the previous problems that have been done.

4.1.1 Selection of Problems Module

This module is considered as the home page in our system. That means when a student runs our system, this page will present all the programming problems. Thus, the human student can choose one problem to teach the agent, Amy. Figure 4.1 shows a screen-shot of this module.



- ✓ Problem 1: Changing the time from seconds to minutes.
- ✓ Problem 2: Converting a temperature in Celsius to a value in Fahernheit.
- ✓ Problem 3: Calculating the area of a circle.
- Problem 4: Calculating the average of six homework grades.
- Problem 5: Changing a mark to a pass or failed grade.
- Problem 6: Changing a numeric grade to a letter grade.
- Problem 7: Testing each number between 0 to 99 and printing all the even n
- Problem 8: Print all the multiples of 5 between 1 and 100.
- Problem 9: Finding the biggest integer of an unsorted set of integers.
- Problem 10: Converting a binary number to a decimal number.
- Problem 11: Calculating a factorial of N (N!).

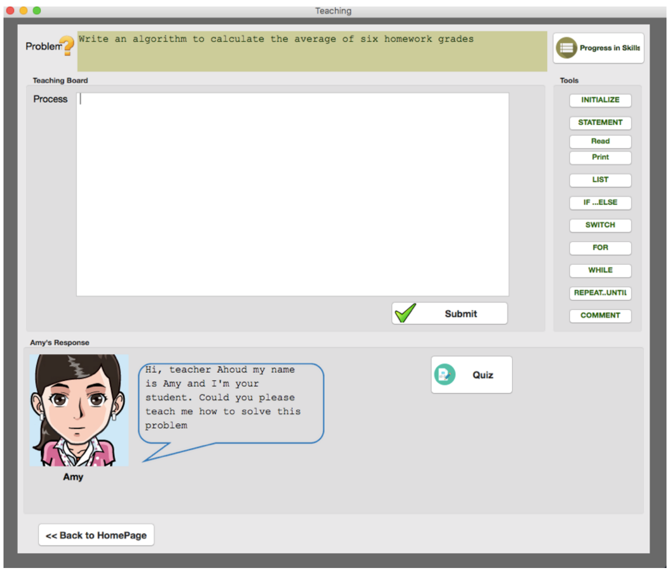
Selected Problem is: Problem 5: Changing a mark to a pass or failed grade.

Figure 4.1: The Homepage of the System

4.1.2 Teaching Module

After selecting the problem, the human student can teach Amy about how to solve the selected problem. This module enables the student to teach the TA, Amy. Hence, Amy will learn from the tutor (student) how to solve the current problem using pseudo-code. Amy will build her knowledge based on the tutor's sequence steps of solving the problem.

Figure 4.2 shows the screen-shot of the teaching module. In the right side of this module, we provide all required keywords that can be used in writing the pseudo-code of solving any problem. For example, for repetitions structure, the keywords include While...End; Repeat ... Until; ForâĂęEnd. For Selection, the keywords include IF...ELSE End, Switch()...Case End. In addition to that, Read, Print, Comment, List, Initialize are the keywords that a student can use. Anyway, the tutor is free to use these keywords or she can write those words by herself.



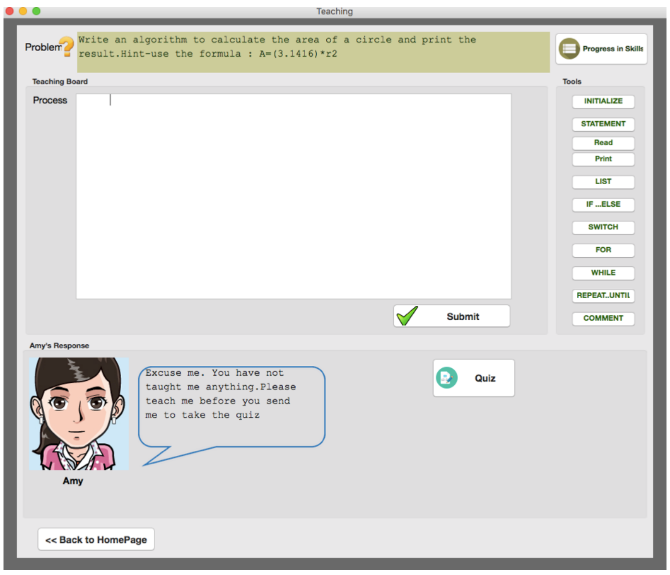
Write an algorithm to calculate the average of six homework grades

Hi, teacher Ahoud my name is Amy and I'm your student. Could you please teach me how to solve this problem

Figure 4.2: The Teaching Workspace

4.1.3 Amy's Feedback and Inquiry Module

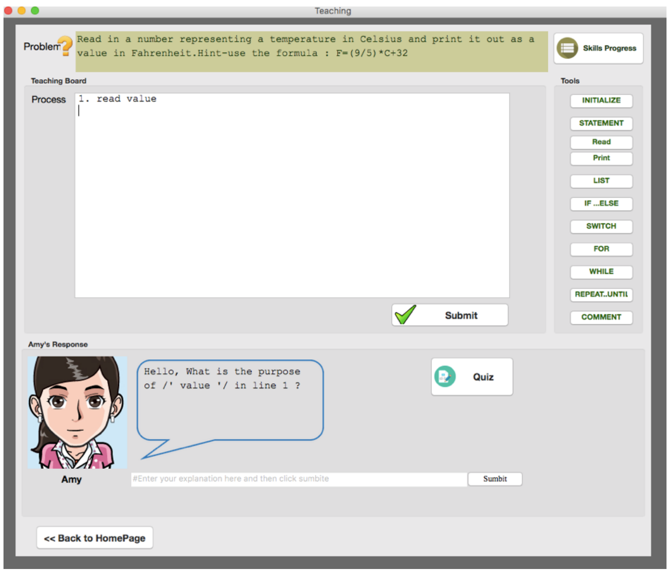
This module enables to present Amy's feedback or her query during the teaching stage. This allows Amy to interact with the tutor in case if the tutor did not behave in an understandable way such as requesting Amy to take the quiz without teaching her. Figure 4.3 shows one example of Amy's Feedback. Furthermore, in this stage, Amy can enquire the tutor, especially about ambiguous variables names. Figure 4.4 shows one example of Amy's inquiry. Also, the tutor can write his own answer to Amy.



Write an algorithm to calculate the area of a circle and print the result.Hint-use the formula : A=(3.1416)*r2

Excuse me. You have not taught me anything.Please teach me before you send me to take the quiz

Figure 4.3: Amy Feedback



Read in a number representing a temperature in Celsius and print it out as a value in Fahrenheit.Hint-use the formula : F=(9/5)*C+32

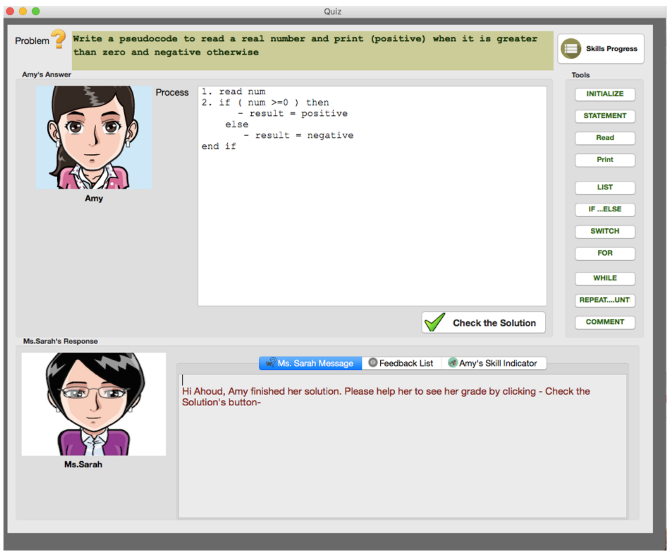
```
1. read value
```

Hello, What is the purpose of /' value '/ in line 1 ?

Figure 4.4: Amy Inquiry

4.1.4 Quiz Module

After Amy is taught by the tutor in the teaching module, she can take a quiz to answer an isomorphic question. The quiz module enables the tutor to assess the knowledge that Amy obtained during the teaching stage. As a result, Amy will approach the question based on what she has been taught. Figure 4.5 shows an example of Amy's solution. As it can be seen, there is another animated agent in this stage which is Ms. Sarah. The following section will explain her role.



Write a pseudocode to read a real number and print (positive) when it is greater than zero and negative otherwise

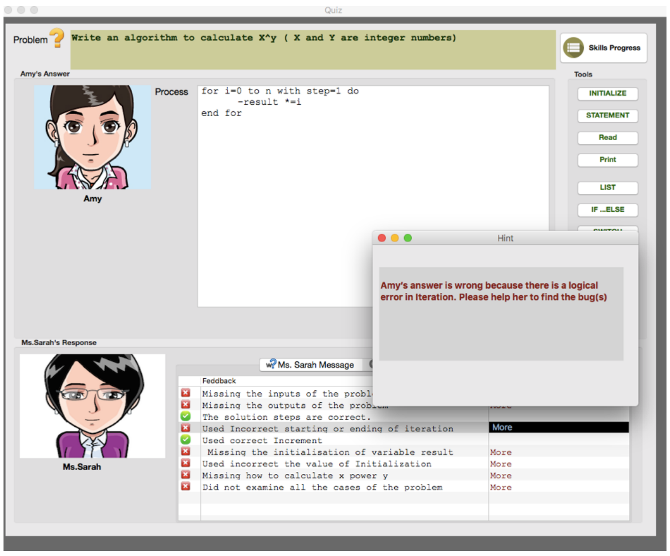
```
1. read num
2. if ( num >=0 ) then
       - result = positive
     else
        - result = negative
end if
```

Hi Ahoud, Amy finished her solution. Please help her to see her grade by clicking - Check the Solution's button-

Figure 4.5: Quiz Workspace

4.1.5 Ms. Sarah Feedback Module

In the Quiz module, Amy needs to see the correctness of her answer. Thus, we provide Ms. Sarah Feedback module. In this module, Ms. Sarah will evaluate Amy's answer. Then, Ms. Sarah provides metacognitive feedback list to Amy. Figure 4.6 shows the feedback list for one of the questions. That will help the tutor to figure how to fix the incorrect parts. Hence, Ms. Sarah will send a message to the tutor in order to help Amy to improve her solution. The tutor can get more information about the feedback by clicking "More". Here, Ms. Sarah will provide information about the incorrect parts. Firstly, she provides clues about the reasons for the incorrect answer of Amy. Secondly, she suggests specific activities to the user in order to help Amy. Usually, she will ask the tutor to re-teach Amy again.



Write an algorithm to calculate X^y (X and Y are integer numbers)

```
for i=0 to n with step=1 do
      -result *=i
end for
```

Amy's answer is wrong because there is a logical error in Iteration. Please help her to find the bug(s)

	Feddback	
✗	Missing the inputs of the probl	
✗	Missing the outputs of the prob	
✓	The solution steps are correct.	
✗	Used Incorrect starting or ending of iteration	More
✓	Used correct Increment	
✗	Missing the initialisation of variable result	More
✗	Used incorrect the value of Initialization	More
✗	Missing how to calculate x power y	More
✗	Did not examine all the cases of the problem	More

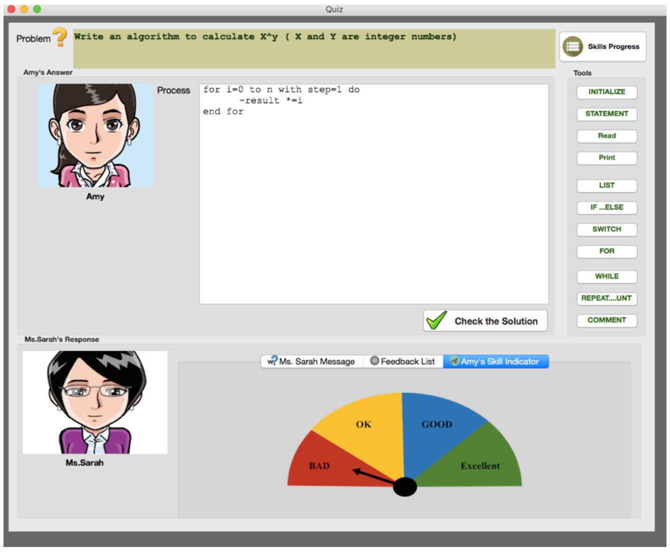
Figure 4.6: Sarah Feedback

4.1.6 Re-teaching Module

After reading the feedback, the tutor will re-think about their strategies and then try to fix Amy solution. Hints: he or she can use the same tools in teaching activity which are on the right sides in quiz interface as shown in figure 4.6. Thus, the tutor will earn many skills such as debugging errors, monitoring the solution process, self-explanation, planning, evaluating and revising skills. All these skills will be considered in teaching and re-teaching activities.

4.1.7 Indicator Module

Any programming problem can have several potential correct answers. However, the levels of these answers vary from acceptable to excellent answers. As a result, we create a module to evaluate the accuracy of the answers that is done by Amy. Ms. Sarah will be responsible to assess Amy's answer for any given problem. Ms. Sarah will represent her evaluation in the skill indicator as shown in the figure 4.7. There are four levels in this indicator: Bad, OK, Good and Excellent.



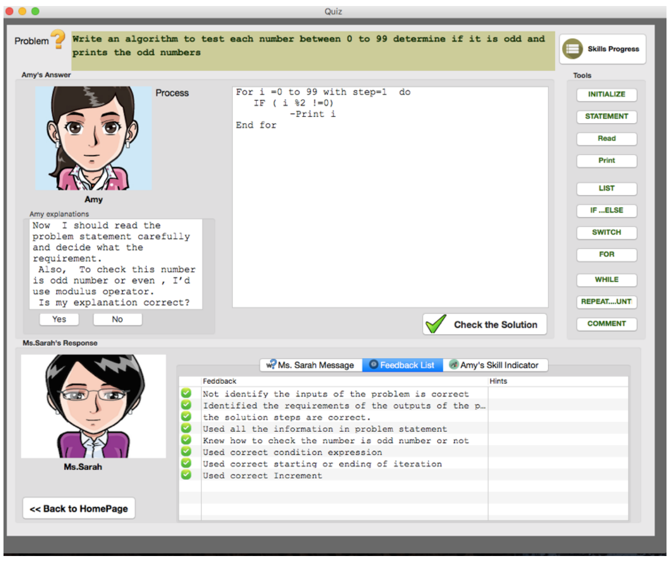
Write an algorithm to calculate X^y (X and Y are integer numbers)

```
for i=0 to n with step=1 do
        -result *=i
end for
```

Figure 4.7: Skill Indicator

4.1.8 Explanation Module

In this module, Amy would provide an explanation after the tutor re-teach her correctly for each of her mistakes of the first attempt. Then, she will ask to confirm her explanation from the tutor. There are several aims of this module. Firstly, this module shows that Amy is an active student. The authors in [ ] recommend that the teachable agent should be an active, creative, self-motivated, and curious during the teaching stage because that enables their tutors to improve the domain and teaching skills. Secondly, we can ensure the tutor has acquired some new strategies. Figure 4.8 shows one example about Amy's explanations.



Write an algorithm to test each number between 0 to 99 determine if it is odd and prints the odd numbers

```
For i =0 to 99 with step=1  do
   IF ( i %2 !=0)
         -Print i
End for
```

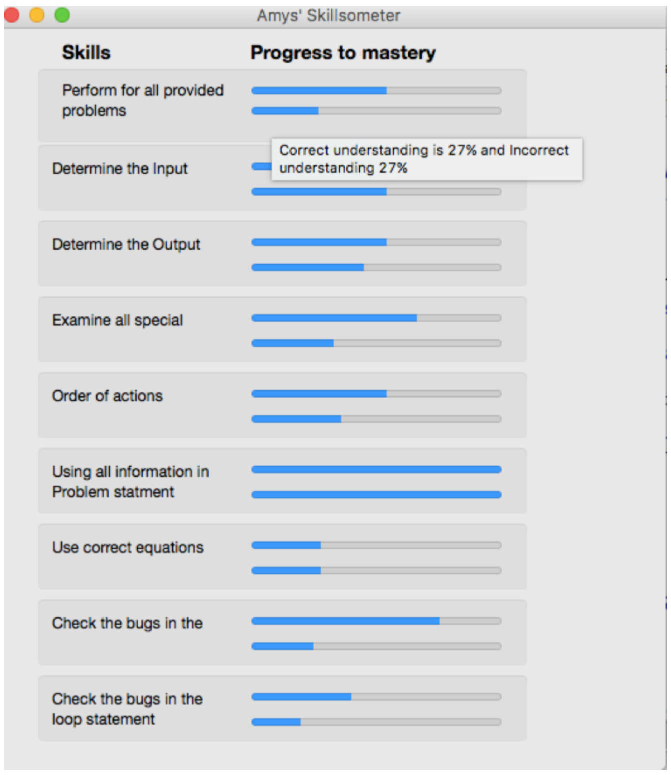
Now I should read the problem statement carefully and decide what the requirement.
 Also, To check this number is odd number or even , I'd use modulus operator.
 Is my explanation correct?

	Feddback	Hints
✓	Not identify the inputs of the problem is correct	
✓	Identified the requirements of the outputs of the p…	
✓	the solution steps are correct.	
✓	Used all the information in problem statement	
✓	Knew how to check the number is odd number or not	
✓	Used correct condition expression	
✓	Used correct starting or ending of iteration	
✓	Used correct Increment	

Figure 4.8: Amy Explanation

4.1.9 Skill Meter Module

This module presents Amy's overall progress for all the problems she has approached as shown in figure. The tutor can access this module during the teaching and quiz stages. This module shows various skills of the learning progress for Amy such as the performing all the problems, determining the inputs and outputs, examining all special cases, checking the bugs in conditional and loop statements. As shown in Figure 4.9, for each skill, there are two progress bars. The first bar is to show the relative amount of problem that was covered. The second progress bar is to show the measurement of the correct and incorrect understanding. The tutor can get this information as percentage numbers by hovering the mouse on the bars.

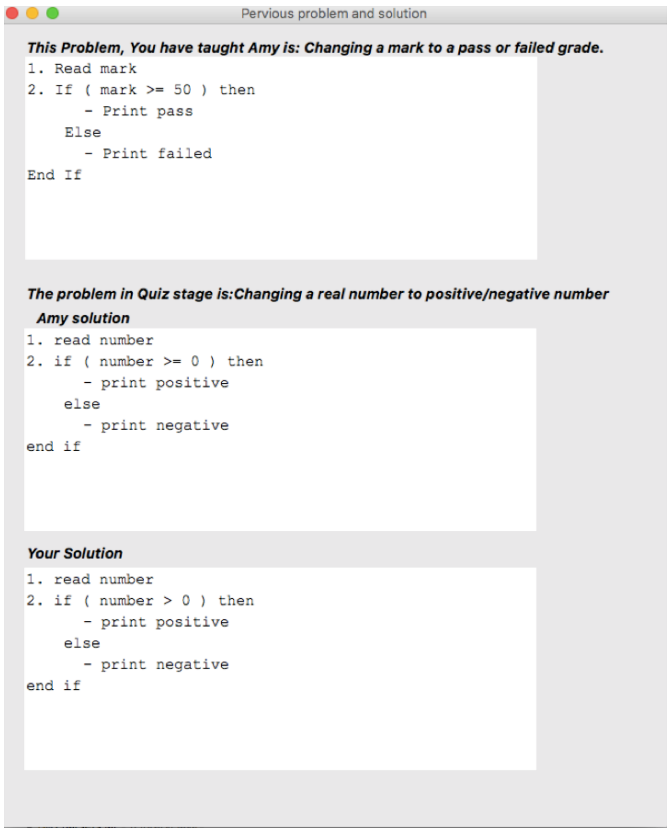


Correct understanding is 27% and Incorrect understanding 27%

Figure 4.9: Skill Meter

4.1.10 Previous Answers Module

The tutor can see the previous answer for a specific problem when teaching Amy. Also, this tutor can see how Amy answer the isomorphic problem. In addition to that, the tutor can see how he/she teaches Amy correctly in order to get the ideal solution. Figure 4.10 shows a screen-shot of this activity.


Pervious problem and solution

This Problem, You have taught Amy is: Changing a mark to a pass or failed grade.

```
1. Read mark
2. If ( mark >= 50 ) then
      - Print pass
    Else
      - Print failed
End If
```

The problem in Quiz stage is:Changing a real number to positive/negative number

Amy solution

```
1. read number
2. if ( number >= 0 ) then
      - print positive
    else
      - print negative
end if
```

Your Solution

```
1. read number
2. if ( number > 0 ) then
      - print positive
    else
      - print negative
end if
```

Figure 4.10: Previous Solutions

4.2 Main Classes in our System

There are 29 created classes in this system. The main classes are briefly described below in the following subsections. These classes are organized into three groups as shown in Figure 4.11: User Interface Classes, Model Classes, and Database classes. Having this structure of classes would make the code more readable and ensure modifiability because it would not affect the entire model if we want to change or update any component of the interface.

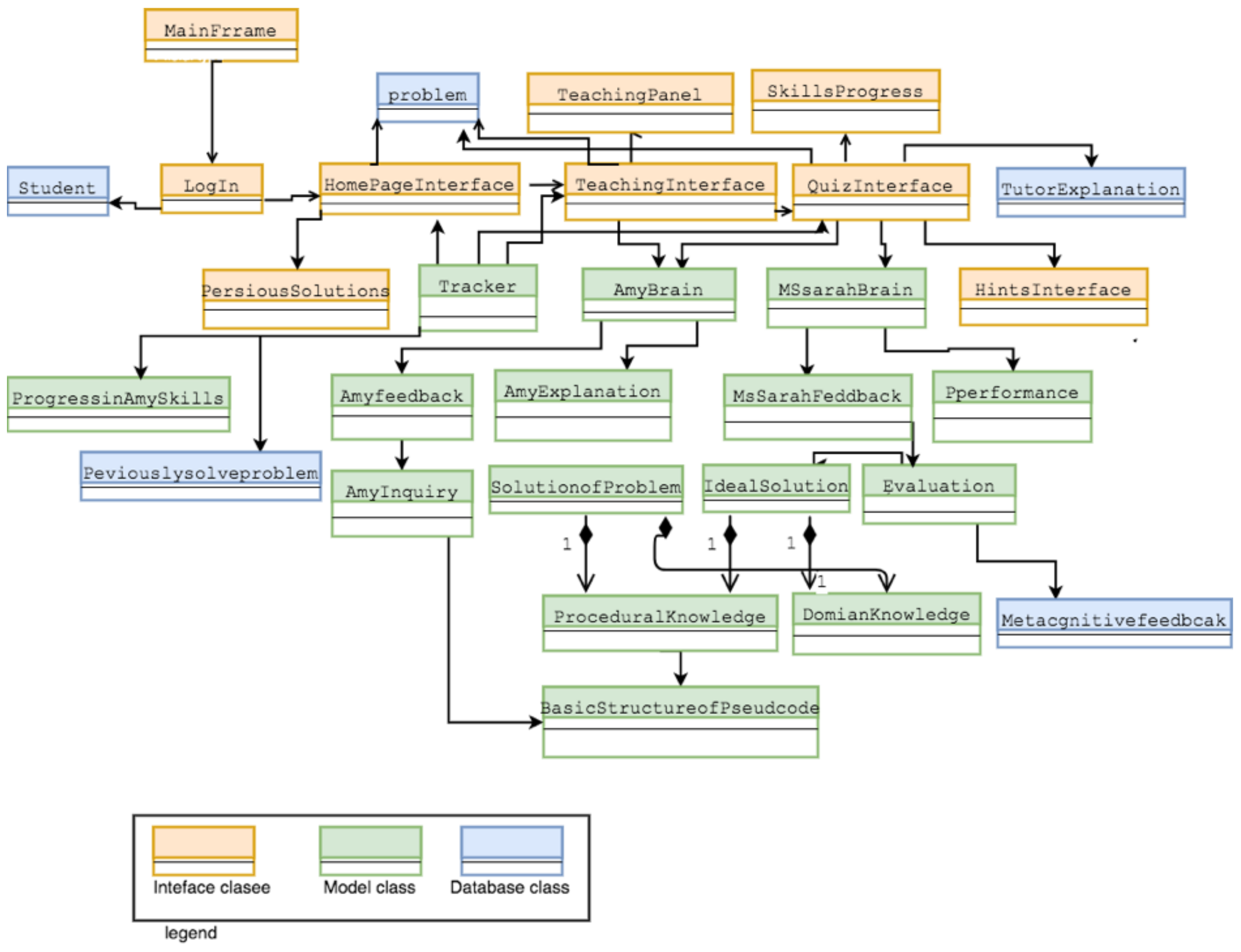


Figure 4.11: Classes of our System

4.2.1 User Interface Classes in the System

User Interface classes define the abstractions necessary for human-computer interaction. Basically, the benefits of user interface class diagram are:

1. To define GUI components such as buttons, text boxes, text fields, etc.
2. To define the required events of each GUI components that capture the user interaction.

There are seven User Interfaces Classes in our systems. The Table 4.1 shows the name of these classes and its description. Figure shows a class relation diagram (UML diagram) with the main user interface classes in the system.

Table 4.1: User Interfaces Classes in our systems and the description

No	Class Name	Class Description
1	MainFrame	Creates all GUI Frames and add them to the application
2	LogIn	Frame for the tutor information such as name, age, gender and university level.
3	HomePageInteface	Frame for provided problems on the system
4	TeachingPanel	Panel for Teaching the agent Amy. It contains some buttons which helps the user to teach the agent (If .. Else, Initialize the variable, or for)
5	TecahingIntefcae	Frame for teaching the agent Amy. It presents the problem statement and some tools for providing the solution and Amy's feedback during this stage if it necessary.
6	QuizIntefcae	Frame for the Quiz stage where the agent Amy's knowledge can be assessed. It presents the problem statement for the quiz and some tools for re-teaching the agent. Also, it presents Ms. Sarah response and feedback.
7	AmySkilloMeterIntefcae	Frame presents Amy's overall progress.
8	HintsIntefcae	It presents hints from Ms. Sarah to show the issues of the solution of Incorrect behaviour
9	PerviousSolutionsInteface	Frame presents previous problems that were taught to Amy along with the provided answer from the tutor to Amy and what Amy has answered during the quiz stage.

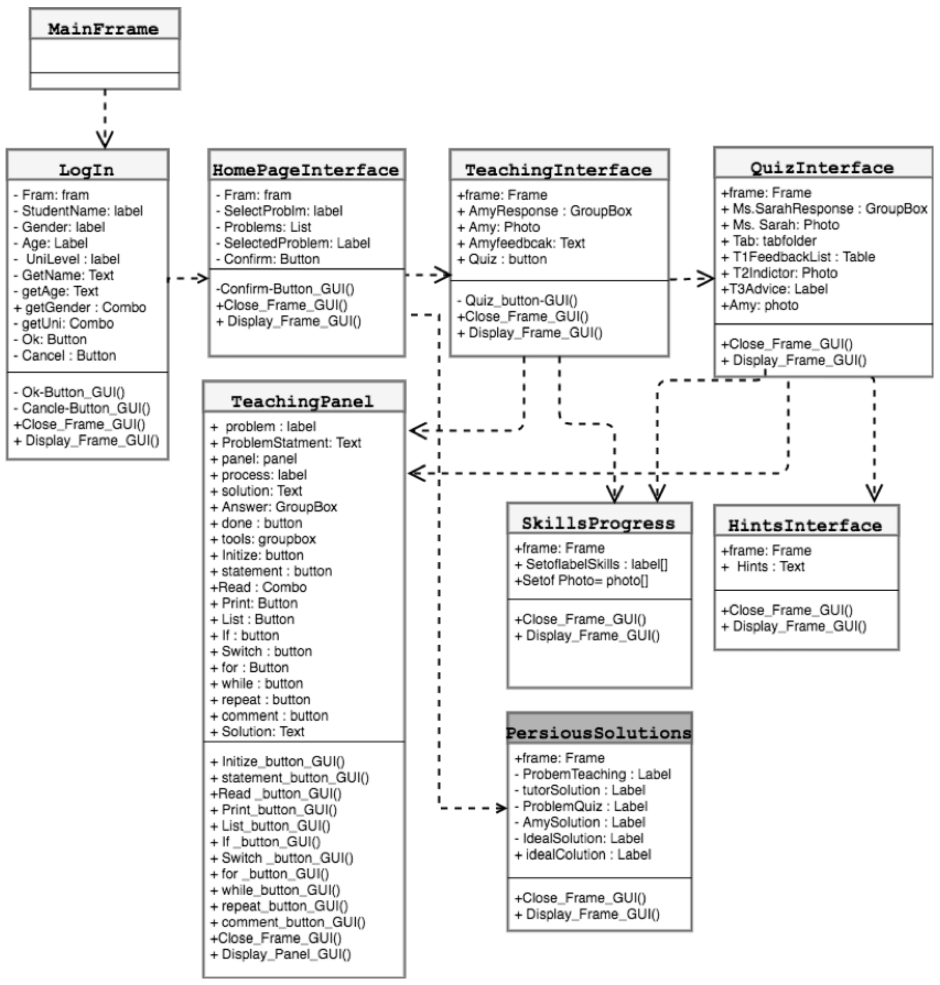


Figure 4.12: The UML of all User Interfaces classes and relationships among them

4.2.2 Model Classes in the System

These classes are considered as the bridge between the Database class and the view. They contain the main components of the systems such as AmyBrain, Ms.SarahBrain, Tracker and Analyzer. We will firstly explain AmyBrain and Ms.SarahBrain in details, respectively. Then, we will explain other classes. Tracker class is to determine the user's progress and inform other agents Amy and Ms. Sarah about what the tutor is doing. AmySkillProgressController is to measure what the correct and incorrect understanding of specific skills of all provided problems.

4.2.2.1 AmyBrain

In this section, we will explain the framework of the agent Amy. This framework is illustrated in Figure 4.13 as UML static structure diagram. The role of Amy is to be an active student during the teaching and the quiz stages. As discussed in the previous sections, Amy is a virtual student that can learn from the tutor (human student). She can also provide feedback and ask the tutor when she could not understand some parts of the solutions. Based on the knowledge that Amy obtained from the tutor during the teaching stage for a particular problem, she can apply this knowledge to another isomorphic problem and try to solve it by herself. So, that can help to show the quality of the knowledge that the tutor passes to Amy. Table 4.2 displays classes that implement Amy's brain. The following subsections provide an explanation on how we implement each of Amy's roles.

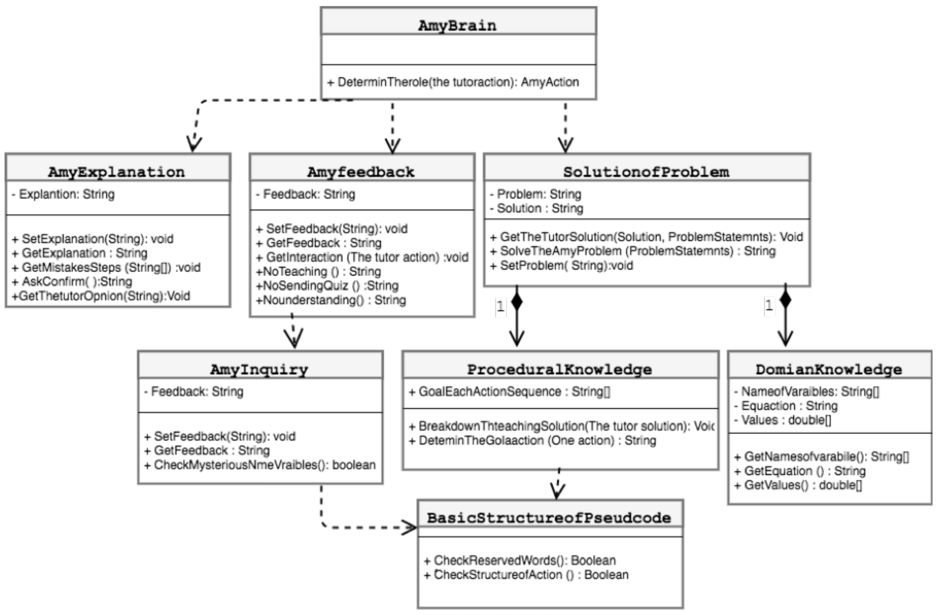

AmyBrain
+ DeterminTherole(the tutoraction): AmyAction
AmyExplanation
- Explantion: String
+ SetExplanation(String): void
+ GetExplanation : String
+ GetMistakesSteps (String[]) :void
+ AskConfirm( ):String
+GetThetutorOpnion(String):Void
Amyfeedback
- Feedback: String
+ SetFeedback(String): void
+ GetFeedback : String
+ GetInteraction (The tutor action) :void
+NoTeaching () : String
+NoSendingQuiz () :String
+Nounderstanding() : String
SolutionofProblem
- Problem: String
- Solution : String
+ GetTheTutorSolution(Solution, ProblemStatemnts): Void
+ SolveTheAmyProblem (ProblemStatemnts) : String
+ SetProblem(String):void
1
1
AmyInquiry
- Feedback: String
+ SetFeedback(String): void
+ GetFeedback : String
+ CheckMysteriousNmeVraibles(): boolean
ProceduralKnowledge
+ GoalEachActionSequence : String[]
+ BreakdownThteachingSolution(The tutor solution): Voic
+ DeteminTheGolaaction (One action) : String
DomianKnowledge
- NameofVaraibles: String[]
- Equaction : String
- Values : double[]
+ GetNamesofvarabile(): String[]
+ GetEquation () : String
+ GetValues() : double[]
BasicStructureofPseudcode
+ CheckReservedWords(): Boolean
+ CheckStructureofAction () : Boolean

Figure 4.13: Showing all the classes that are needed to build AmyBrain

Table 4.2: Showing all the classes name and the description for building AmyBrain

No	Class Name	Class Description
1	AmyBrain	A class to handle control the role Amy learn, solve, provide feedback, inquire and explanation
2	SolutionofProblem	Class to get the solution from the tutor and provide the solution for quiz
3	DomianKnowledge	Class to provide all declarative knowledge for each problems.
4	ProceduralKnowledge	Class to determine all action sequence for solving problem for each problem.
5	AmysFeedbcak	Class to provided Amy's feedback to the tutor depending on his interactions with Amy
6	AmyInquiry	This class is to extend from Amy's Feedback. It is to provide Amy's Inquiry if the tutor used some mysterious variable names
7	BasicStructureofPseudocode	Class to contain all the basic operation and structure about Pseudocode
8	AmyExplanation	It provides the Amy explanation about what she learns from the user after re-teaching correctly

Teaching Amy
The tutor (human student) teaches Amy by solving the programming problem using pseudo-code. Pseudo-code is an artificial and informal language that helps programmers to develop algorithms. As the tutor provides the pseudo-code as a solution, Amy monitors the way and the strategy that

the tutor uses. As a result, Amy can learn from the tutor those strategies for solving the isomorphic problem. Regardless the solution quality provided by the tutor whether the tutor teaches Amy a correct, incomplete, or incorrect solution, Amy will follow the same strategy to solve the isomorphic problem.

As mentioned before, Amy does not know the steps or strategies for solving any problem, but she knows all the basic operations of pseudo-code, structure and the goal from every steps in pseudo-code structure That means, she knows what the goal from read statement, print statement, selection and iterative statements. As it ban be seen, she behaves like novice programmers which they usually know the basic cognitive skills in the programming structures: variables and constants; logical and comparison operators; selection statements; iterative statements; and arrays. Furthermore, she knows the declarative knowledge of each problem. We mean by declarative knowledge in the problem is domain knowledge. For example, she knows the base of a hexadecimal is 16 and binary is 2. Another example, she knows the equation for checking the number is odd or even. The human teachers usually do not provide the declarative knowledge with the quiz for solving any programming problem because they expect that the students know it before. For that, we build Amy like novice programmers who usually know this domain knowledge and the basic computer operations and structure.

However, she will learn from the tutor about the procedural knowledge of the problem. The definition of procedural knowledge is the action sequence for solving problem (Rittle-Johnsan & Wagner 1999). In other words, she will learn the steps of solving specific problems. First, she will get the tutor solution and the problem. Then, she will break down the solution into some steps. After that, she will consider the goal from each steps and how their linking to other steps. Thus, she will learn from the tutor.

In terms of how she will solve the second problem in quiz stage, first, she accesses domain knowledge of this problem. Then, she will follow the same procedural knowledge of the first problem in teaching stage with changing name variable by suitable name of the problem and domain knowledge.

Amy's Feedback and Inquiry

As mentioned previously, Amy provides feedback and inquiry in some cases during teaching Stage. This feedback is inquisitive rather than explicit instruction for solving the problem. She focuses only on the interactions between herself and the tutor such as ineffective behaviours and making corrective suggestions. For example, if the tutor does not teach Amy and wants to request her to take a quiz, Amy might say *"Excuse me. You have not taught me anything. Please teach me before you send me to take the quiz"*. Another example, if the tutor taught Amy and did not request her to take a quiz. She might say *"Thank you for teaching me, you can send me to take quiz"*. All the information that help Amy to decide which suitable feedback is needed to send to the tutor is depending on the sending information from Tracker class.

Furthermore, Amy inquires in some cases during teaching stage if she faces some ambiguous actions or mysterious variable names. As we know, sometimes the name variables like i, j and x if the teacher did not explain what those variables mean, it may make the student confused. For that, Amy will ask about the meaning of these mysterious variable names. For example, for finding the biggest integer from unsorted integer list, the tutor could initialize the biggest number like x variable in line 3, Amy will say *"what do you mean by 'x' variable in line 3?"*. For that, Amy will have the knowledge for all basic operations actions in pseudo-code such as Read, Print, If and Else, For, While, Repeat, and Until. Also, she knows the structure for each one. For example, "READ Seconds" she knows the seconds is a variable name. Appendix K shows all the Amy feedback and inquiry.

Amy explanation
After the tutor re-teaches Amy correctly about her mistakes, Amy will provide what she understood and asks to confirm her explanation. First, she knows all her mistakes or incomplete tasks based on Ms. Sarah's feedback. Then, she monitors the tutor during the re-teaching process of her mistakes. After that, she will provide explanation about each mistake if the tutor re-teach her correctly. Finally, she asks to confirm her explanation. For example, if Amy did not consider some special cases of the problem such as if 'y' is less than zero for finding X^y. Then, the tutor teaches her that correctly. She will say *"Thanking you for teaching, this problem has more than one case. I should consider all cases. One of them is y less than zero. Is my explanation correct?"*. Appendix L shows some of Amy explanations especially her explanations depending on the tutor behaviour on the problem.

4.2.2.2 Ms. SarahBrain

Now, we will focus on the agent framework (Ms. Sarah). The framework is illustrated in Figure 4.16 as UML static structure diagram. The role of Ms. Sarah is to mentor the agent Amy and the user. Ms. Sarah can provide guidance and suggestions. She has several works can do it to the human tutor and Amy. Firstly, she grades Amy's solutions and provides metacognitive feedback to the tutor and Amy. Secondly, she can evaluate Amy performance for solving the problem to display the performance as Amy's skill Indicator for current problem in quiz. The Table 4.3 shows some classes that Ms. Sarah need them to do her role for assessing and monitoring Amy's solution and interacting with the tutor and Amy.

Table 4.3: Showing all the classes name and the description for building Ms. SarahBrain

No	Class Name	Class Description
1	MsSarahBrain	Class to handle control the role Ms.Sarah grades, evaluates, Amy's solution provide feedback to the tutor and Amy
2	Performance	Class of range of performance and indicator to Amy's skills
3	MsSarahFeddback	Class to provided Ms.Sarah feedback to the tutor and Amy what correct and incorrect
4	Evaluation	Class to evaluate and check many parts in solution what is correct and incorrect compare it with ideal solution which will be represent as class

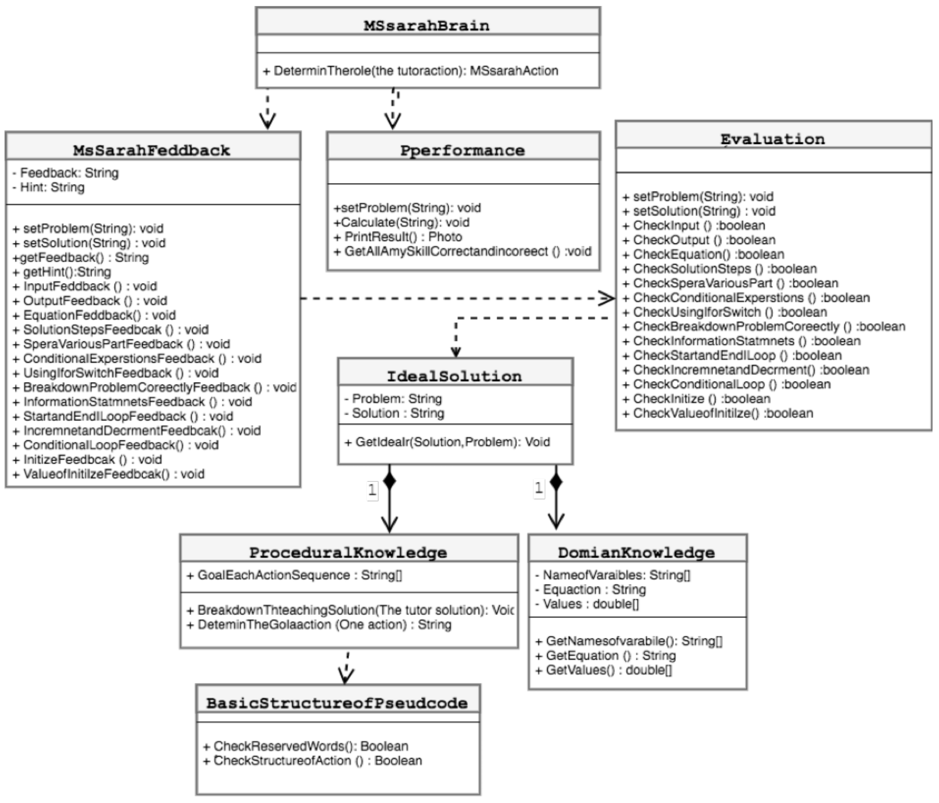
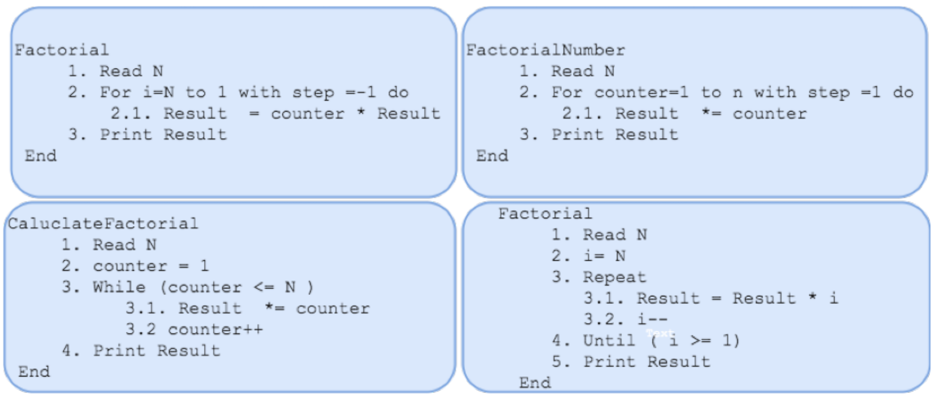

MSsarahBrain
+ DeterminTherole(the tutoraction): MSsarahAction
MsSarahFeddback
- Feedback: String
- Hint: String
+ setProblem(String): void
+ setSolution(String) : void
+getFeedback() : String
+ getHint():String
+ InputFeddback () : void
+ OutputFeedback () : void
+ EquationFeddback() : void
+ SolutionStepsFeedbcak () : void
+ SperaVariousPartFeedback () : void
+ ConditionalExperstionsFeedback () : void
+ UsingIforSwitchFeedback () : void
+ BreakdownProblemCoreectlyFeedback () : void
+ InformationStatmnetsFeedback () : void
+ StartandEndlLoopFeedback () : void
+ IncremnetandDecrmentFeedbcak() : void
+ ConditionalLoopFeedback() : void
+ InitizeFeedbcak () : void
+ ValueofInitilzeFeedbcak() : void
Pperformance
+setProblem(String): void
+Calculate(String): void
+ PrintResult() : Photo
+ GetAllAmySkillCorrectandincoreect () :void
Evaluation
+ setProblem(String): void
+ setSolution(String) : void
+ CheckInput () :boolean
+ CheckOutput () :boolean
+ CheckEquation() :boolean
+ CheckSolutionSteps () :boolean
+ CheckSperaVariousPart () :boolean
+ CheckConditionalExperstions () :boolean
+ CheckUsingIforSwitch () :boolean
+ CheckBreakdownProblemCoreectly () :boolean
+ CheckInformationStatmnets () :boolean
+ CheckStartandEndlLoop () :boolean
+ CheckIncremnetandDecrment() :boolean
+ CheckConditionalLoop () :boolean
+ CheckInitize () :boolean
+ CheckValueofInitilze() :boolean
IdealSolution
- Problem: String
- Solution : String
+ GetIdealr(Solution,Problem): Void
1
1
ProceduralKnowledge
+ GoalEachActionSequence : String[]
+ BreakdownThteachingSolution(The tutor solution): Voic
+ DeteminTheGolaaction (One action) : String
DomianKnowledge
- NameofVaraibles: String[]
- Equaction : String
- Values : double[]
+ GetNamesofvarabile(): String[]
+ GetEquation () : String
+ GetValues() : double[]
BasicStructureofPseudcode
+ CheckReservedWords(): Boolean
+ CheckStructureofAction () : Boolean

Figure 4.14: Showing all the classed that needed to build Ms. SarahBrain

Ms. Sarah Feedback List
As mentioned before, she will grade Amy solution and provide feedback what the correct and incorrect in her solution. In addition, she also will ask the human tutor to help Amy for providing a correct solution. For more information about the feedback of Ms. Sarah was discussed in section 3.3.5

As we know the correct solutions for any could be represent in different ways. In other words, programming problem has more than one solution. For that, we build Ms.Sarah to consider all the situation for every solution. For example, in Figure 4.15 four solutions are provided the same results.

In terms how she will deal with all the situation, she will firstly access to the declarative and procedural knowledge of the problem, then she will verify the steps of the problem by using the methods in Evaluation class. Then, she will decide which the suitable feedback to have to send to the tutor and Amy.



```
Factorial
     1. Read N
     2. For i=N to 1 with step =-1 do
         2.1. Result  = counter * Result
     3. Print Result
 End
```

```
FactorialNumber
     1. Read N
     2. For counter=1 to n with step =1 do
         2.1. Result  *= counter
     3. Print Result
 End
```

```
CaluclateFactorial
     1. Read N
     2. counter = 1
     3. While (counter <= N )
           3.1. Result  *= counter
           3.2 counter++
     4. Print Result
 End
```

```
   Factorial
        1. Read N
        2. i= N
        3. Repeat
            3.1. Result = Result * i
            3.2. i--
        4. Until ( i >= 1)
        5. Print Result
     End
```

Figure 4.15: Four solutions are provided the same results for solving the same problem.

Depending on the our problems as discussed on the section —, strategist will be verified form Ms. Sarah. They are as following:

1. Determining the input and output.
2. Examining all special cases.
3. Order of actions.
4. Use correct actions in right place
5. Using all information in problem statement.
6. Using correct equations.
7. Checking the bugs in the conditional statement.
8. Checking the bugs in the loop statement such start, end and increment or decrement for loop.

Display the Performance
Ms. Sarah will calculate the performance of Amy and the tutor skills to present problem as the indicator. First, she will decide all the strategist that needed to solve the specific problem. Then, she will evaluate all the correct and incorrect. Finally, she sums all the correct strategies divide by all the total strategies that needed problem. Then, it is decided which is Bad, Ok, Good and Excellent. The next equation presents how to calculate.

$$\text{Result} = \frac{\sum_{i=1}^{n} correct\ Stragiest(i)}{Count\ all\ Stratgies} \begin{cases} \text{IF result from 0.0 to 0.25 Then, indictor indicates to Bad} \\ \text{IF result form 0.26 to 0.5 Then, indictor indicates to Ok} \\ \text{IF result form 0.6 to 0.99 Then, indictor indicates to Good} \\ \text{IF result equals 1.0 Then, indictor indicates to Excellent} \end{cases}$$

Figure 4.16: How to calculate the performance and classify the results as Bad, Ok, Good, and Excellent

4.2.2.3 Tracker

As mentioned previously , the tracker uses an automated approach to determine the user's progress and notify other agents Amy and Ms. Sarah. For that, for every events that the tutor does in the Graphical user interface (GUI) it will be notified to the Tracker. Then it will detect what is. Also, one of its role is to follow Amy progress for each problem in order to show her progress in skills as progress bars for her tutor. Figure 4.17 shows the Tracker class as UML static structure diagram.

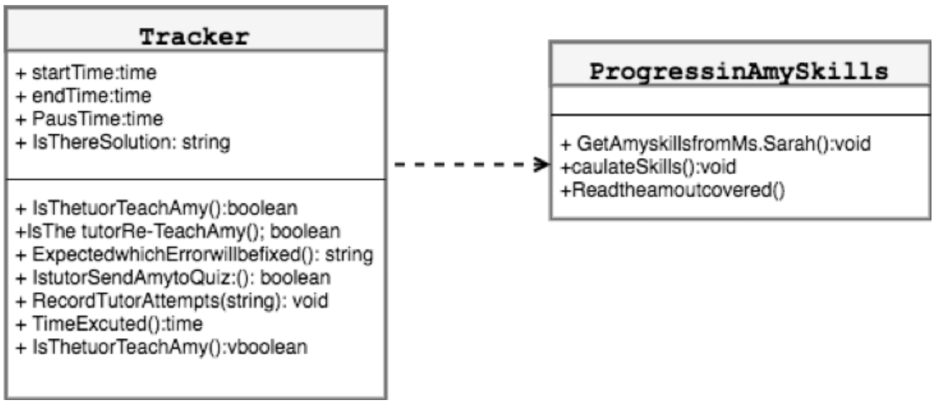

Tracker
+ startTime:time
+ endTime:time
+ PausTime:time
+ IsThereSolution: string
+ IsThetuorTeachAmy():boolean
+IsThe tutorRe-TeachAmy(); boolean
+ ExpectedwhichErrorwillbefixed(): string
+ IstutorSendAmytoQuiz:(): boolean
+ RecordTutorAttempts(string): void
+ TimeExcuted():time
+ IsThetuorTeachAmy():vboolean
ProgressinAmySkills
+ GetAmyskillsfromMs.Sarah():void
+caulateSkills():void
+Readtheamoutcovered()

Figure 4.17: The Tracker class as UML static structure diagram

4.2.3 Database Classes in The System

In order to interact with database components in terms of adding, deleting and updating, we needed to build some classes. There are seven database classes as shown in Table –. The main goal from theses classes is to keep the tutor's information, problems provided, tutor's attempts to teach Amy and summaries about the tutor's metacognitive strategies and acquired strategies. Table shows every class and and class descriptions.

Table 4.4: Database Classes in our systems and their description

No	Class Name	Class Description
1	Student	It is possible to reach all student's information such as name, age, gender and their Uni year
2	Problems	It keeps the description of the developed problem for teaching and quiz stage and the expert solutions for each problem.
3	TutorExplanation	It keeps the user opinion about Amy explanations. If he agree or disagree and why or the tutor answer for Amy inquiry.
4	Metacognitive feedback	It retrieves which is suitable feedback that Ms. Sarah decide
5	PrviousslySolvedProblems	It keep the number of problem that have done, nd what the tutor solutions and Amy solution

4.2.4 Generation of Log Files

In order to achieve our aim of the examining the effects of our system on the novice programmers' skills, we implemented log files. Also, this files will help us in our experiment which will be discussed in the chapter 5. The system automatically recorder user events interactions. We build three log files for recorder the user interaction. The first file is to recode the user interactions for selected the problems in homepage. The specific parameter in this file are user name, time of opening the frame, user attempts before clicking on confirm button with time and, selected the previously solved problem and confirming the final chosen problem.

The second file is to recode the user interactions in teaching stage. The specific parameters in second file are username, problem tutored and number, time of opening the frame, time of starting teaching User Attempts for teaching, final solution, Amy feedback and inquire about she misunderstands and User explanation, Time of Sending Amy to Take Quiz , Time of selected to Back to Homepage and progress in skills.

The third file is to recode the user interactions in Quiz stage. Every action from opening the frame until the close it will be recorded such what the agents do and how the user act with them. The parameters in the file user name, problem and number in this stage, Time of Opening the frame, Amy solution, Time of Click -Check solution button- for helping Amy, Time of starting Re-teaching, user attempts for reteaching, time of selected skills Indicator, time of reading Ms. Sarah message, Time of selected Feedback list, Selected More for more details, Amy explanation and user opinion, time of getting ideal solution. Every action will be recorded it with the time stamp. Appendix M shows screenshots of theses the files.

Chapter 5

Experimental Study of the system

The main goal of this chapter is to outline how we design the experiment, how the pilot study will be conducted, and how the participants will be recruited and allocated to the different groups in the experiment. Also, it presents all the materials and documents that will be used in the experiment.

5.1 Objective

As mentioned previously, feedback in general improves students' performance in problem-solving. For that, it becomes important in computer-based learning environments. There are many types of feedback. Firstly, metacognitive feedback is like guided feedback that is to assist a student in solving problem but without giving the correct solution. Secondly, another type is called Knowledge of Correct Response (KCR). It informs the learner of the correct answer to a specific problem with no additional information. Therefore, our research questions are the following:

1. What is the effect of KCR and guided metacognitive feedback on novice programmer's own problem solving skills (planning, monitoring, evaluation, debugging skills)?
2. How can KCR and guided metacognitive feedback affect novice programmers to adjust their approach for teaching the teachable agent (Amy)?

Since our research questions are regarding two different type of feedback, we will observe and measure the effect of these two types on the students who will use our systems.

5.2 Experimental Design

The experiment design has two groups: experimental group and control group. The first group will use the system which provides guided metacognitive feedback form Ms. Sarah in the Quiz stage (as presented in the previous chapters). The Control group will use the second system which has the same features of the previous system. However, Ms. Sarah in this system only provides the correct answer to the human students. Figure 5.1 shows the two groups of subjects each with specific conditions.

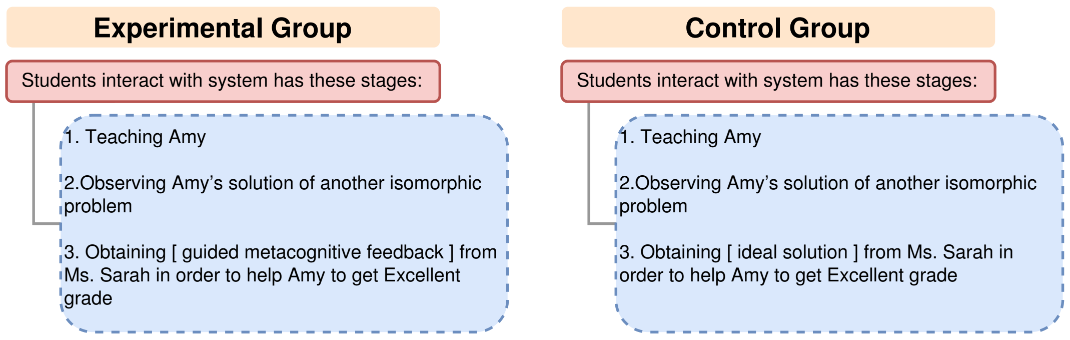

Experimental Group
Students interact with system has these stages:
1. Teaching Amy
2.Observing Amy's solution of another isomorphic problem
3. Obtaining [guided metacognitive feedback] from Ms. Sarah in order to help Amy to get Excellent grade
Control Group
Students interact with system has these stages:
1. Teaching Amy
2.Observing Amy's solution of another isomorphic problem
3. Obtaining [ideal solution] from Ms. Sarah in order to help Amy to get Excellent grade

Figure 5.1: difference between control and Experimental groups

5.2.1 Procedural Skill Test

The experimental design has three procedural skills tests: pre-test, post-test and delayed-test. The aim of these tests is to assess participants' proficiency in programming problems. For each test, there are two parts. The firs part is to estimate students' knowledge in terms of whether or not they would be able to solve a given problem by answering "Yes or "No" to the question. For example: "Do you think you can solve the given problem correctly?". The first part in Appendix A ,B and C show it pre-test, post-test and delayed-test, respectively. After completing the first part, the participants will do the second part which require to programming problems into pseudo-code. The second part in Appendix A ,B and C present the second part for pre-test, post-test and delayed-test, respectively

Regarding the description of containing the three tests, they provide isomorphic programming problems. We develop these problems into three constructs for structured program: sequence, selections and repetitions. The pre-test includes four problems: (one problem for sequence (i.e. simple instructions), two problems for selections (two-way selection and multiple selections) and one problem for repetition). In addition, the post-test includes four problems; (one problem for sequence (i.e. simple instructions), two problems for selections (two-way selection and multiple selections) and one problem for repetition). Also, the delayed-test includes 4 problems; (one problem for sequence (i.e. simple instructions), one problem for selections (two-way selection) and two problems for repetition). As can be seen, the number of the problems that have the repetition of constructs has been increased in the delayed test. We believe the repetition of constructs needs more control and monitoring skills. Especially, they also are a combination of sequence and selections of constructs.

In terms of the given time for each test, the time was given for pre-test is 30 minutes for the two parts. Also, the post-test and delayed-test should be completed in 40 minutes. We did not decide how the time should be consumed for the first part and the second part in order to avoid confusing participants. Anyway, we will inform them about the instructions and time of the tests.

The scoring of each answer of the problems is given as follows: 2.0 points for an ideal answer, 1.0 point for a partially correct answer, 0.5 points for an incomplete answer and 0 for an incorrect answer [20] [31]. In this study, we are interested in comparing between the experimental and the control group and not in absolute measures of performance improvement in either group separately. Appendix D shows ideal solutions of three tests.

To clarify more about the reasons for differentiating the score of ideal and correct solutions, it is because the main goal our system (as mentioned earlier). Our goal is to enhance the novice programmer's skills to get the ideal solution for any problem. For that, we want to measure student's prediction of their knowledge and their actual performance. We use the Knowledge Monitoring Assessment(KMA) method [73] for measuring KMA. Especially, (KMA) appears to be a naturalistic and robust measure of knowledge monitoring that has good reliability and excellent internal validity [74].

Table 5.1 presents the KMA with an eight score value for a, b, c, d, e, f, g, h, A score of 1 is given for the following situations of (a) and (h). A score of -1 is given for the following situations of (g) and(b). A score of -0.5 for all the others. After that, The mean of the KMA scores over all the problems solved, yields the current KMA state of the Student. Table 5.2 presents the classification of the KMA value, students with a KMA value between -1 and -0.25 are categorized as someone with a weakness for estimating correctly their knowledge level in the majority of situations. Whereas a student who is average in correctly estimating their knowledge level is indicated by a KMA value between -0.25 and 0.5. A high KMA (0.5 -1) shows that they are able to estimate their knowledge level correctly most of the time. Appendix D shows all the ideal solutions of the problems in the three tests.

Table 5.1: KMA values for performance and prediction

Performance	Prediction	
	Know	Do not Know
Provides ideal solution	a	b
Provides partially correct solution	c	d
Provides incomplete solution	e	f
Provides incorrect solution.	g	h

Table 5.2: Classification of KMA

KMA Value	Classification	Description
-1 to -0.25	Low	Weakness at estimating knowledge level correctly in majority of situations
-0.25 to 0.5	Average	Average in estimating knowledge level correctly, but makes frequent slightly wrong or completely wrong estimation
0.5 to 1	High	Most of the time makes correct estimation

5.2.2 Motivational test

We are interested in the measurement of metacognition of participants before starting using our systems. There are many surveys that can achieve the targets such as Metacognitive Awareness Inventory MAI [13], Learning and study strategies inventory (LASSI) [75] and The Motivated Strategies for Learning Questionnaire (MSLQ) [76]. MSLQ is a very well established measurement of not only metacognition, but motivation [77]. It contains 81- items, 7-point Likert scale (1 = not at all true of me and 7 = very true of me). There are essentially two sections to the MSLQ, a motivation section, and a learning strategies section. The motivation section consists of 31 items that assess students' goals and value beliefs for a course, their beliefs about their skill to succeed in a course, and their anxiety about tests in a course. The learning strategy section includes 31 items regarding students' use of different cognitive and metacognitive strategies. In addition, the learning strategies section includes 19 items concerning student management of different resources. We select this MSLQ because it covers some limitations in others such as answering the items in a specific way depending on the domain the learner has in mind. For instance, the item "I summarise what I have learned after I finish" in evaluation component, may be rated high if the learner thinks of her behaviour in literature or history, or low if she refers to her pattern in maths.

Furthermore, we pickup what the suitable items for programming problem course. Appendix E shows this questionnaire. It includes 7 items for motivation part and 14 items for learning strategies. The reasons of selecting this number of items in learning strategies part is because it includes the self-regulated and metacognitive strategies. Before this part, we set up some questions for getting personal and education background.

5.2.3 Evaluation Survey

In order to evaluate the system from participants' perspective, we collect their feedback on our system at end of the experiments. Appendix F shows the usability questionnaires. The first part from question 1 to 10 is based on [78].

Other questions from 11 to 16 are to assess what the participants think about the benefits of the system to develop their thinking process on solving problems. In addition, there are two open end questions to explain what the most positive and negative aspects from participant perspectives.

5.3 Structure of study

5.3.1 Preparation of Study

Before study days, we installed the system on the computers' lab. Also, we stuck the participant ID on the screen that can be seen in order to keep their personal information such their names to be anonymous and confidential. Appendix G shows the IDs for both control and experimental groups. Absolutely, we printed and checked all questionnaires and rearrange them to avoid any mistakes during the study. Furthermore, we recorded video on how to use the system. We will play it to the participants in the first session.

Furthermore, we send email to the group of computer science students as the experiment advertisement as shown in Appendix J. After they wanted to take part in the experiments, we inform to participants the days and time of the study days and the location. In addition, we collected their signature to attend all the sessions using Consent Form as shown in Appendix H. Finally, the materials used in the research work are presented in Appendix I.

5.3.2 Tasks of Study

Each participant taking part in the experiment was required to attend three sessions on different days. Table 5.3 shows the structure for each session with the time limited for each subtask in that day. The first session is on 11/10/2017 at 11:00 am, and the second session is on 15/10/2017 at 11:00

am. Delayed-Test is administered one week after the second session and was on 22/10/2017 at 11:00 am. The first and second sessions are two hours. The third sessions is 40 minutes

Our reasons for make more than one session is to make good atmosphere for participants and more trust for using the system. Especially we want to measure the improvement of own skills and every participant could have their own ability. Thus one session can be difficult to measure the progress of participants if participant lacks the confidence of using the system. Also, make one session would force us to make the time of sessions long. Then, that would affect on participants that could be fatigued.

Furthermore, we make these sessions are not consecutive days in order to measure any long-term retention of acquired skills, especially we deal with metacognition.

Table 5.3: Organization of the experiment sessions.

Sessions	Task	Duration
The first session	Motivated Strategies for Learning Questionnaire	10 minutes
	Pre-test: Part I and Part II.	30 minutes
	Video for how to use system	10 minutes
	Using our system	1 hour and 10 minutes
The second session	Using our system	1 hour and 10 minutes
	Post-test: Part I and Part II.	40 minutes
	Evaluation and Usability Questionnaire	10 minutes
The third session	Delayed-test: Part I and Part II.	40 minutes

5.4 Participants and The Place of the Conducted Experiment

The study was agreed to be conducted in Saudi Arabia in the College of Computer at Al-lieth in Umm Al-Qura University due to the easiness to conduct the study and collect data. We will select novice programmers who study at least one programming course and passed this course. All the participants have started to study programming courses in the first year in university as required in the plan of the department. Also, the programming course often focus on the features of programming language more than how to think about solving problems. We will divide the participants into two groups randomly without any prior knowledge of their programming skills. Due to the time constraint, we only consider female participants.

Chapter 6

Results and Discussion

The main goal of this chapter is to summarise and discuss the results of the data analysis. Also, it presents our findings for the research questions. In addition, it also presents an analysis of the participants' opinions about the system.

6.1 Participants

The participants consisted of a total 21 undergraduate students of Computer Science department. We randomly divide them into two groups: 11 participants in the experimental group and 10 participants in the control group as shown in Table 6.1. All of them were female. Their ages ranged between 18 and 25 years. Furthermore, all of them studied two programming courses which are Introduction Computer Science and Computer Programming. They have also passed at least one of the programming course.

All the participants except one attended all the three sessions. One participant from the experimental group was absent in third session. Thus,the number of participants in the experimental group was 10 but the control group did not change.

Table 6.1: The number of participants in both groups

	Experimental group	Control group
Number of Participants	11 (Novice Programmers)	10 (Novice Programmers)

6.2 Results of Motivational Test of Participants

As discussed in chapter 5, we want to the measure meta-cognition in participants. They were asked to fill-in Motivated Strategies for Learning Questionnaire (MSLQ) which had 21 items (includes 7 items for motivations part and 14 items for learning strategies as shown in Appendix E). Each item had a 7-point Likert scale (1= not at all true of me and 7 = very true of me). Students were asked to use their behaviour in most recent programming course when filling in the questionnaire.

In terms of The Motivation part, Table 6.2 shows the average of each question for all participants in the experiment. Most participants feel anxious when they take the exam. That would negatively impact their performance. Interestingly, the average rating for control of learning is 3.95 ; i.e. they do not believe if they did not understand the course material that does not mean they did not try hard enough to understand the course. Furthermore, they prefer course material that really challenges them. So, they prefer the programming problem solving because it is considered challenging.

Table 6.2: The average for each questions for all participants in the experiment in Motivation part

Value component	Definition	The questions	The average of each question
Intrinsic Goal orientation	It refers to the participants' perception of the reasons why they are engaging in a learning task	In a class like this, I prefer course material that really challenges me so I can learn new things.	6
		The most satisfying thing for me in this course is trying to understand the content as thoroughly as possible.	5.52
Extrinsic Goal orientation	It completes Intrinsic Goal orientation and concerns the degree to which students perceive themselves to be participating in a task for reasons such as grades, rewards, performance, evaluation by others, and competition.	The most important thing for me right now is improving my overall grade point average, so my main concern in this class is getting a good grade	6.67
Task Value	It refers to the student's evaluation of the how interesting, how important, and how useful the task is	I am very interested in the content area of this course.	6.00
Control of Learning	It refers students' beliefs their the effort to learn	If I do not understand the course material, it is because I did not try hard enough.	3.95
Expectancy for success and self efficacy	It refers to performance expectations and relates specifically to task performance	I am confident I can understand the most complex material presented by the instructor in this course.	4.43
Test anxiety	It has been found to be negatively related to expecting expectancies academic performance	I have an uneasy and upset feeling when I take an exam.	5.57

In terms of learning strategies part, we have 8 strategies that were asked from the participants. The table 6.3 shows theses strategies with average rating for them. The average rating for help seeking is 4.81. That means the participant are willing to ask for support from others such as peer or teachers. We also want to measure the meta-cognition. The self-regulation is one of aspects we measured. This was measured using 6 questions. The average rating for self-regulation strategies is 5.33. That means the participants has achieved some level of self-regulation.

Table 6.3: The average for each questions for all participants in the experiment in learning strategies part

Value component	Definition	The questions	The average of each question
Rehearsal	It involves reciting or naming items from a list to be learned. These strategies are best used for simple tasks and activation of information in working memory rather than the acquisition of new information in long term memory	When studying for this class I read my class notes and the course readings over and over again.	4.57
Elaboration Strategies	It helps students to store information into long-term memory by building internal connections between items to be learned	I try to relate ideas in this subject to those in other courses whenever possible	5.52
		When I study for this class I pull together information from different sources such as lectures, readings and discussions	4.38
Organization Strategies	It help the learner select appropriate information and also construct connections among the information to be learned.	When I study the readings for this course I outline the material to help me to organize my thoughts	5.43
Self-regulation	It is one of the components of metacognition. Self-regulation activities include many activate such as planning, monitoring look back	I ask myself questions to make sure I understand the material I have been studying in this class	5.24
		When studying for this course I try to determine which concepts I do not understand well.	5.57
		When I study for this class I set goals for myself in order to direct my activities in each study period	4.19
		When I become confused about something I am reading for this class I go back and try to figure it out	6.29
		If course materials are difficult to understand, I change the way I read the material	5.14
		Even when course materials are dull and uninteresting I manage to keep working until I finish.	5.57
Collaborating	It refers work jointly on an activity, especially to produce or create something	When studying for this course I often try to explain the material to a classmate or a friend.	6.33
Help seeking	The degrees of asking to support from others	Even if I have trouble learning the material in this class I try to do the work on my own without help from anyone.	4.81
Time management	It is the process of planning and exercising conscious control over the amount of time spent on specific activities	I make good use of my study time for this course	5.43
Critical Thinking	It refers to the degree to which students report applying previous knowledge to new situations in order to solve problems	Whenever I read or hear an assertion or conclusion in this class I think about possible alternatives.	4.33

6.3 Results for RQ1

In order to answer Research Question 1 (RQ1), we wanted to explore the effect of knowledge of correct response (KCR) and guided meta-cognitive feedback on novice programmers' problem-solving skills.

In order to investigate the impact of both types of the aforementioned feedback, we verified the results by comparing the pre-test and post-test, we also compared the pre-test and delayed-test for both groups (Experimental and Control).

We marked each question for each participant as follows: 2 points if the participant provided a correct solution for the question, 1 point if the participant provided a partially correct solution, 0.5 point for incomplete solution and 0 for incorrect solution or if the participant did not provide any solution. Then, we calculated the total for all the questions in each test. Hence, each test has the same number of questions and isomorphic problems as mentioned in Chapter 5.

To compare the pre-test and post-test for participants in the experimental group, Figure 6.1 shows the performance for those participants in both tests. We found that 9 out of 11 participants have been improved. In other words, the performance of the experimental group has been improved by 82%.

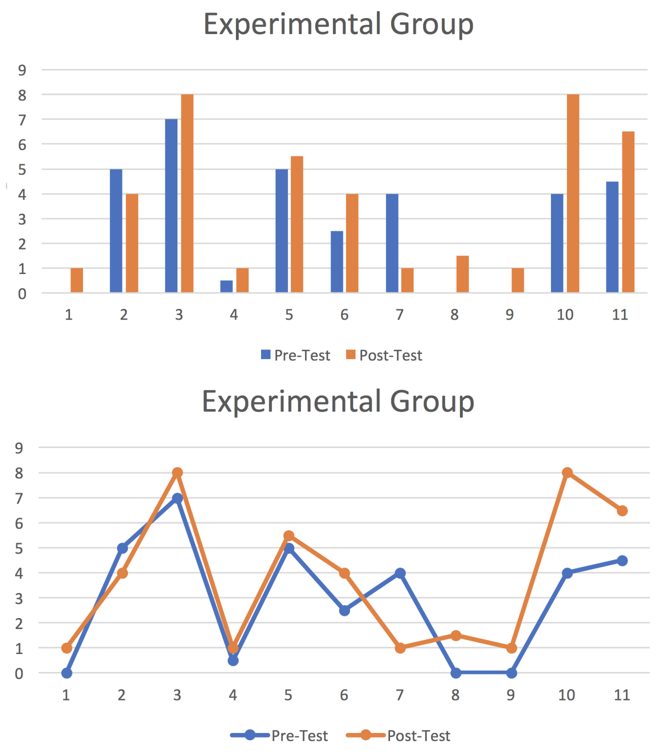


Figure 6.1: Comparing The Score of Participants for Both Tests (Pre-Test and Post-Test) in The Experimental Group

To compare the pre-test and post-test for participants in the control group, Figure 6.2 shows the performance for those participants in both tests. We found that only 3 out of 10 participants have improved. That means the performance of only 30% of the participants in control group has improved.

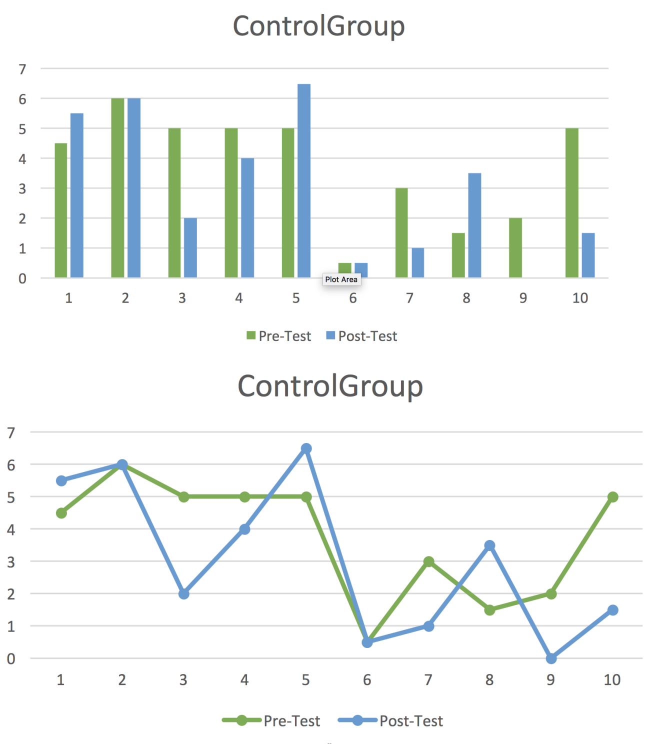


Figure 6.2: Comparing The Score of Participants for Both Tests (Pre-Test and Post-Test) in The Control Group

As mentioned previously, the problems in delayed-test are more challenging compared to pre-test and post- test. To compare the pre-test and delayed-test for participants in the experimental group, Figure 6.3 shows the performance of the experimental group in pre-test and delayed-test. We found that 7 out of 10 participants have been improved. That means the performance of the experimental group has been improved by 70%.

To compare the pre-test and delayed-test scores for participants in the control group, Figure 6.4 shows the performance for those participants in both tests. We found that only 5 out of 10 participants have been improved. That means the performance of the control group has been improved only by 50%.

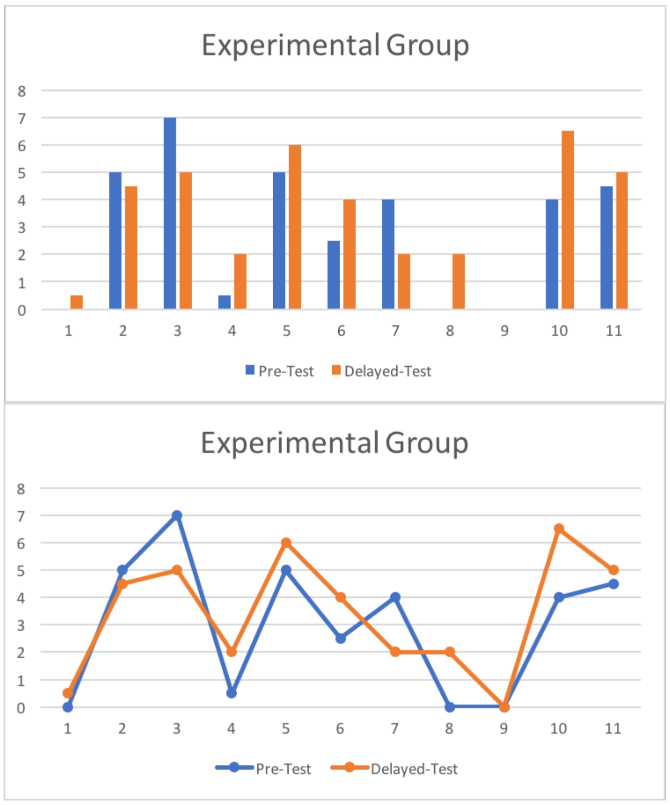


Figure 6.3: Comparing the Score of Participants for Both Tests (Pre-Test and Delayed-Test) in The Experimental Group

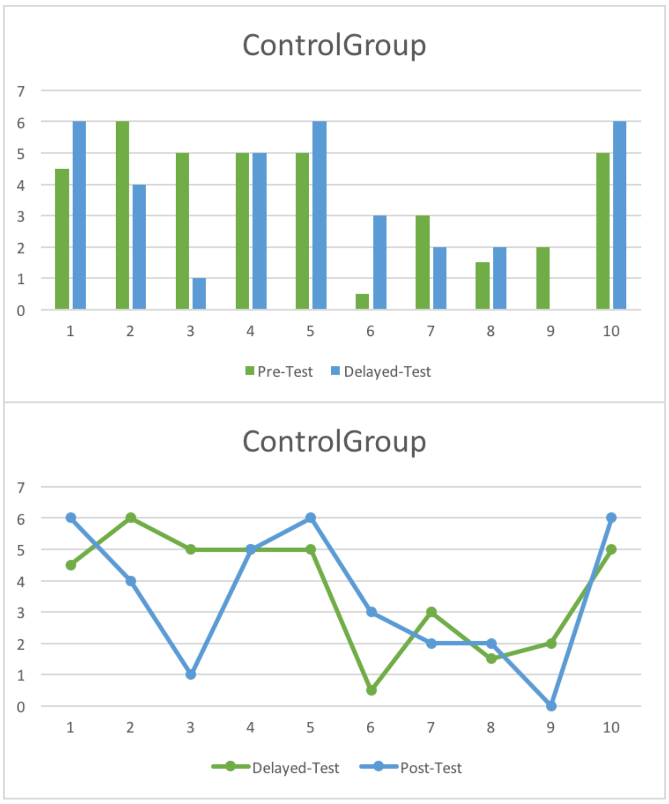


Figure 6.4: Comparing The Score of Participants for Both Tests (Pre-Test and Delayed-Test) in the Control Group

As we can see, some of the participants in both groups have improved, and we believe that the reason of this is the learning by teaching effect which participants have done in both groups. We can also notice a high percentage of experimental group participants have been improved more than the control group because in the experimental group, we enable the participants to use their meta cognitive skills during solving problems whereas in the control group we provide the ideal solution in the case if the participants did not provide a correct solution.

In order to investigate the participants (who have improved in both groups), we divided the participants into three groups based on their pre-test scores as shown in table 6.4 below. The total scores of the pre-test and the post test is 8. Table 6.5 shows how we divided the three groups of participants for both experimental and control groups.

Table 6.4: Classification of groups based prior knowledge

Scores	Groups based prior knowledge
0 to 3	Low Prior Knowledge (LPK)
4 to 5	Average Prior Knowledge (APK)
6 to 8	High Prior Knowledge (HPK)

Table 6.5: the numbers of participants based prior knowledge split

Groups	HPK	APK	LPK
Experimental Group (11 Patricians)	1	5	5
Control Group (10 Patricians)	1	5	4

As indicated above, only 3 participants have improved in the control group in the post-test. After dividing participants based on the Table 6.4, we found that those participants belong to the average prior knowledge (APK) group whereas participants in both HPK and LPK did not improve.

When we did the same procedure for the experimental group, we found that participants, who their scores improved in the post-test, are from all of the three groups (HPK, APK, and LPK). Those participants are 1 participant from the HPK, 3 participants from APK, and 5 participants from LPK. Interestingly, two of the participants from APK group achieved scores as the participants in the HPK group and one participant in the LPK group achieved scores as the participants in the APK group.

6.4 Changes in Knowledge Monitoring Accuracy (KMA)

In order to assess the knowledge monitoring which is part of the metacognition component, we use KMA as discussed in chapter 5. Firstly, we calculated the mean of the KMA scores over all the problems solved for each participant for both experimental and control groups in the three procedural tests. Table 6.6 shows the experimental group for all tests (pre-test, post-test, and delayed-test). Table 6.7shows the control group for all test (pre-test, post-test, and delayed-test).

Table 6.6: KMA value of the experimental group for three tests. Hints (Ref) in the table means this participant was not attended in this test

Participants	KMA value of pre test	Classification of pre test	KMA value of post test	Classification of Post test	KMA value of delayed test	Classification of delayed test
P1	-1	Low	-0.75	Low	-0.875	Low
P2	0.25	Average	-0.125	Average	0.25	Average
P3	0.625	High	0.75	High	0.625	High
P4	-0.375	Low	-0.75	Low	0.25	Average
P5	0.125	Average	0.25	Average	0.25	Average
P6	0.625	High	-1	Low	0.25	Average
P7	-0.25	Average	-0.75	Low	1	High
P8	0	Average	-0.625	Low	0.5	High
P9	-0.5	Low	-0.375	Low	Ref	Ref
P10	-0.75	Low	0.75	High	-0.25	Average
P11	-0.125	Average	0.625	High	-0.625	Low

Table 6.7: KMA value of the control group for three tests

Participants	KMA value of pre test	Classification of pre test	KMA value of post test	Classification of Post test	KMA value of delayed test	Classification of delayed test
P1	0.125	Average	0.25	Average	0.25	Average
P2	-0.25	Average	0.5	High	-0.5	Low
P3	0.125	Average	-0.5	Low	-0.375	Low
P4	0.125	Average	-0.125	Average	-0.125	Average
P5	0.125	Average	0.625	High	1	High
P6	-0.375	Low	-0.375	Low	0.25	Average
P7	0.125	Average	-0.75	Low	0.5	High
P8	-0.625	Low	-0.25	Average	1	High
P9	0	Average	-0.5	Low	-1	Low
P10	0.125	Average	-0.25	Average	0.625	High

In order to compare between the two groups, we measure the means of the KMA scores for both groups among theses tests. Table 6.8 shows descriptive values between the two groups. By looking at the tables, we can see the mean of the score of the KMA for the experimental group has improved in between pre and post-test as predicted. In contrast, in control group, there is no improving in monitoring their knowledge.

Table 6.8: The average of KMA in three tests for both experimental and control groups

The average of KMA	in pre-test	in post-test	in delayed-test
Experimental group	-0.5	-0.181818182	0.1375
Control group	-0.1	-0.1375	0.1625

Looking at the distribution of KMA organized by the categories (low, average and high) the groups presented the following distribution. Both figure 6.5 and 6.6 show the the distribution of KMA for both groups for the pre-test and post-test.

For the control group, the distribution at the beginning of the experiment was: 80.0% had an average KMA, 20.0% a low KMA and 0.0% presented a high KMA in the pre-test. This distribution changed at the end of the experiment (post-test) showing the increase of the high KMA group with 20.0% of the students and falling into the average category with 40.0%. There was also an increase of low KMA students (40.0%).

For the experimental group, the distribution at the beginning of the experiment was 46.0% had an average KMA, 3.6.0% a low KMA and 18.0% presented a high KMA. We can observe a redistribution of these percentiles at the end of the experiment, where 55.0% presented a low KMA, 18.0% had an average KMA, and 27.0% a high KMA.

Interestingly, there is increased of high KMA students in the experimental group as shown in Figure 6.6. That means most of the time they make correct estimation.

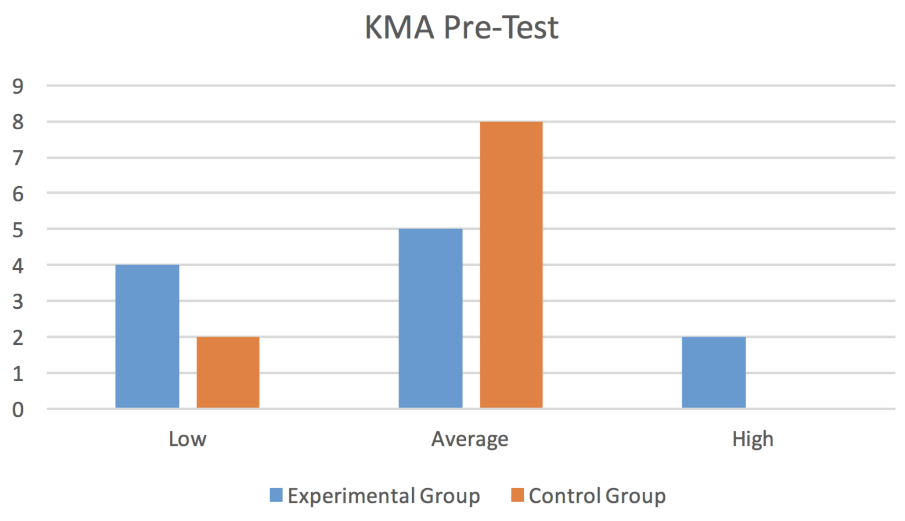


Figure 6.5: The three classified of KMA Score for both groups for Pre-test

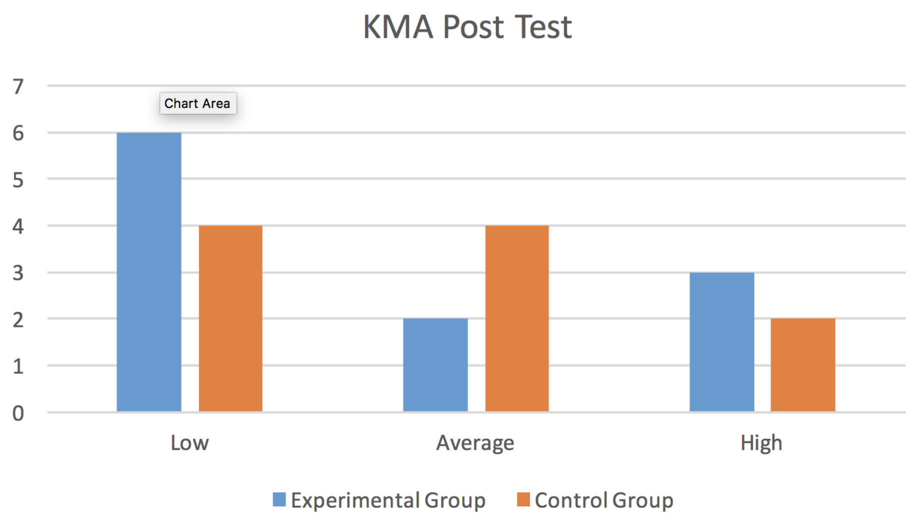


Figure 6.6: The three classified of KMA Score for both groups for Post-test

In terms of the delayed test, Figure 6.7 shows the KMA classification of delayed test for both group. This distribution is changed at the end of the experiment in the delayed test. It becomes 30% in high KMA of experimental group and becomes 50% and 20% in average and low KMA respectively. By comparing the KMA for the three tests, we can see that the KMA of high group have been increased for the experimental group as well for the control group.

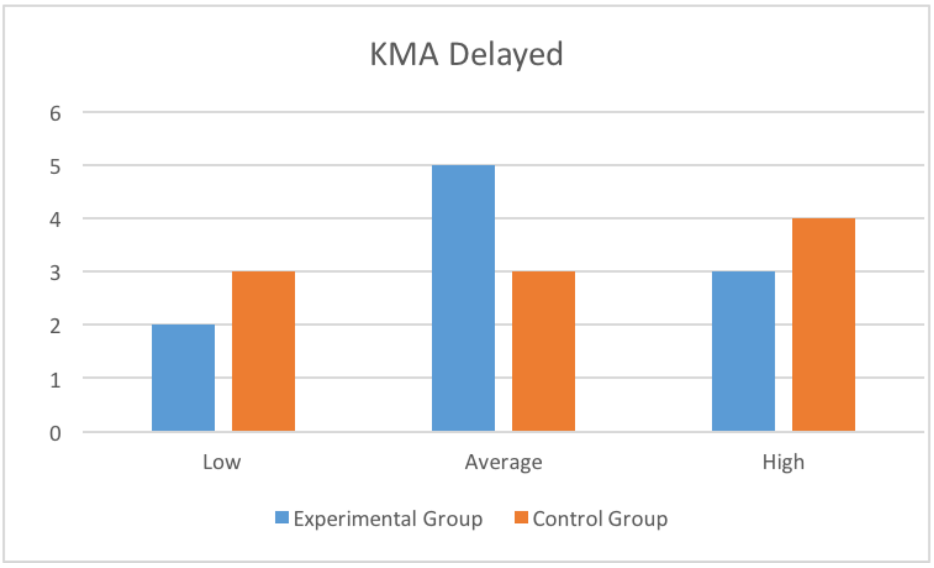


Figure 6.7: The three classified of KMA Score for both groups for Delayed-test

6.5 Results for RQ2

The Research Question 2 (RQ2), as explained in Chapter 1, is "How can KCR and meta-cognitive feedback affect on novice programmers to adjust their approach for teaching the teachable agent

(Amy)?" For this question, we analyse the performance for teaching Amy. We were looking at the log files for each participant, and we mark the degree of teaching Amy for the selected problems in the Teaching stage. Then, we measure the average for each problem in log files for both groups as shown in figures 6.8 and 6.9. Hence, the challenges of the problems increase from problem 1to problem 11. All participants taught Amy the problems from 1 to 8. However, Participants did not teach Amy the problem 9, 10, and 11 because the limitation of the time. All participants spent the limited time in the experiments to teach Amy. That is similar to [6][53] [54], the learning by teaching paradigm motives students spend more time and effort to get their goal from teaching the agents.

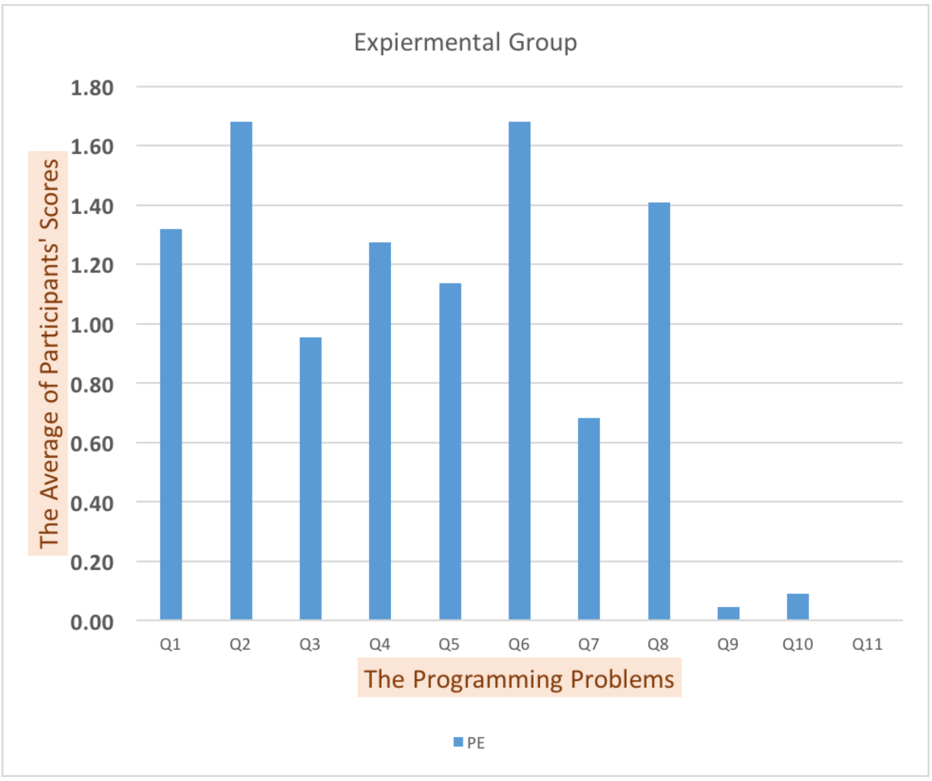


Figure 6.8: The average for each problem in log files for Experimental group

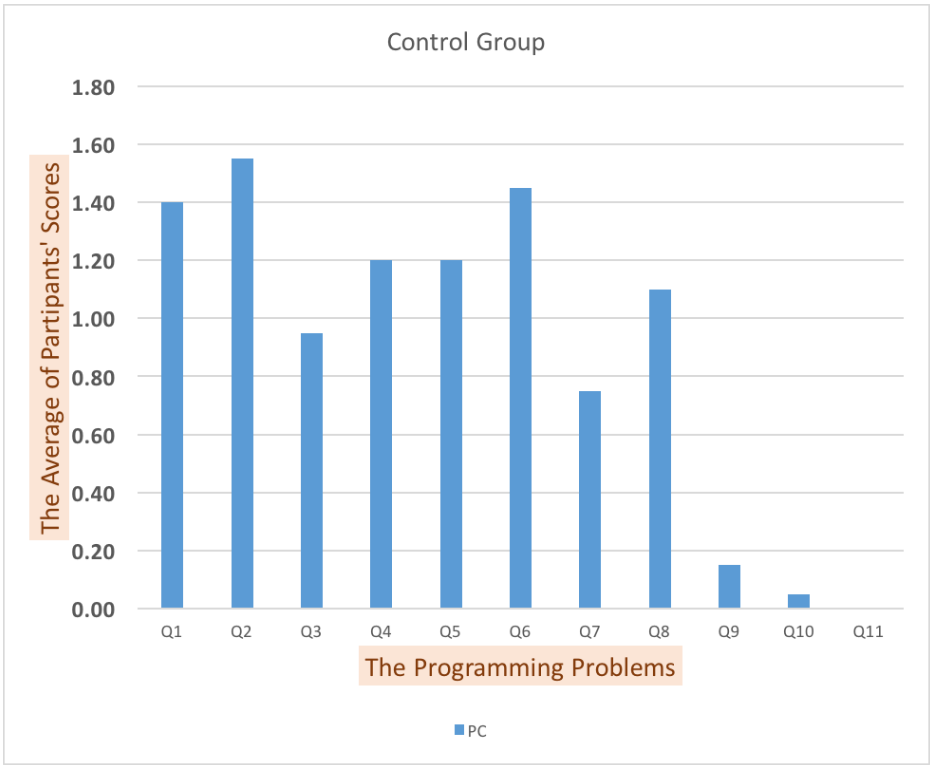


Figure 6.9: The average for each problem in log files for Control group

By looking at Figure 6.10, it shows the average of the performance for each problem of both groups (experimental and control groups). There is a little bit increase on the performance of the experimental group than control groups. That means participants from the experimental group have acquired some strategies such as looking back, monitoring and control for teaching Amy and solving the problem. Generally, the performance of both groups depended on the difficulty of the questions because the learning by teaching strategy can engage the metacognition in implicit way. The performance of control group was close to the experimental group.

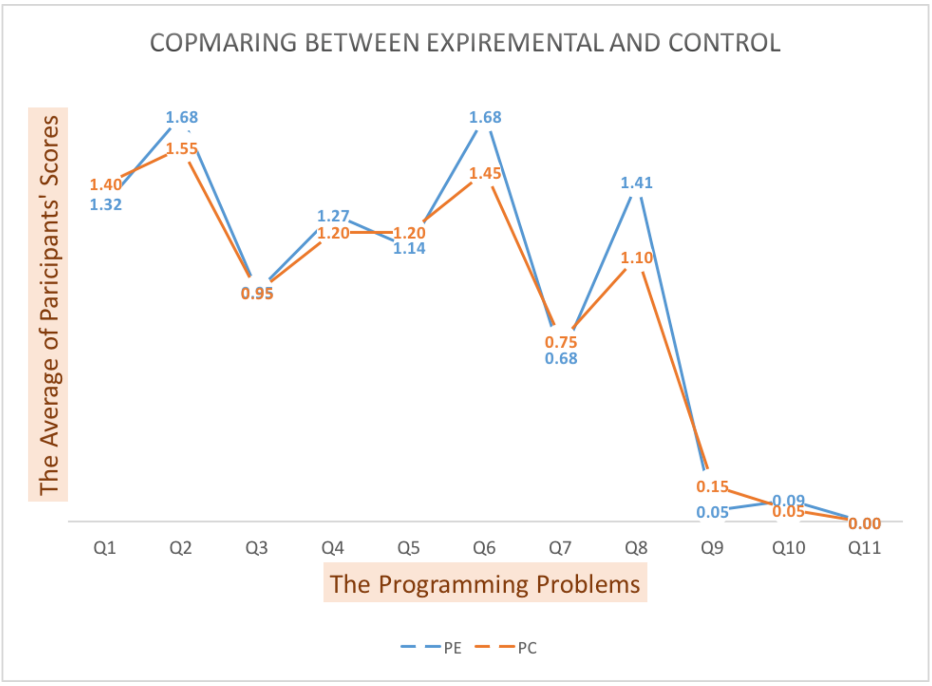


Figure 6.10: Comparing the average for eleven problems in log files for both groups

6.6 Analysis of Interaction With The System

This section gives overview about the participants opinion about the system. It presents the results of the usability and evaluation of the system.

6.6.1 Can Programming Problem Solving Be Learned By The Computer-based System?

We collected the participants' opinions if programming problem solving can be learned by the computer-based system before and after using the system for both experimental and control group. This question is 7-point Likert scale (7 = strongly agrees and 1= strongly disagree). We measure the average for both group as shown in 6.9.

Table 6.9: the participants' opinion if programming problem solving can be learned by the computer-based system

	The average before using the system	The average after using the system
Experimental group	5.82	6.46
Control group	5	5.7
The average of all participants	5.429	6.095

As we can seen, most the participants that agree and believe that programming problem solving can be learned by the computer-based system. Also, the average of of their opining have been increased after using the system. That means both groups agree and strongly agree that computer system can help the to learned the programming problem solving.

6.6.2 Usability of The System

At the end of the experiment, all participants filled their opinions of the usability of the system using the point Likert scale (7 = strongly agrees and 1= strongly disagree). We categorised our ten questions (part of usability as shown in Appendix F) into five attributes as describes by [79] as following:

- Affect that shows the degrees of the user's emotional reaction to the software .
- Efficiency that means the degree of user's feeling about the system assists them or not.
- Helpfulness that measures the extent to which the system is easy to use.
- Control that measures the extent to which the user feels in control of the software when use the system.
- Learnability that measures the speed and facility with which the user feels that they have been able to master the system, or to learn how to use new features when necessary.

Table 6.10 shows these attributes and the average of each question for both groups. In addition to that, the average of all participants for both groups for each question. We found their participants opinion is agree and strongly agree about each questions. Interestingly, we found the most participants would like to use this system frequently and recommend it to their friends. Unexpectedly, the average of control group (that is easy to make the system do exactly what I want) is slightly agree.

Table 6.10: Attributes of usability questions and the average of each question for both groups and the average of all participants

Attributes	Usability questions	The average of experimental group	The average of experimental group	The average of all participants for both groups for this question
Affect	I would recommend this system to my friends.	6.18	5.9	6.05
Affect	I would like to use this system frequently	6.55	6.2	6.38
Efficiency	I would find the system useful in my studies for problem solving	6.27	6.2	6.24
Helpfulness	It was easy to learn to use this system	6.27	6.1	6.19
Helpfulness	The interface of this system is pleasant.	5.73	6.2	5.95
Helpfulness	The organization of information on the system screens is clear.	6.36	6.7	6.52
Helpfulness	The way that Ms. Sarah's information is presented is clear and understandable	6.09	6.2	6.14
Helpfulness	The way that Amy's information is presented is clear and understandable	6.09	5.8	5.95
Learnability	I feel comfortable using this system	5.73	6	5.86
Control	It is easy to make the system do exactly what I want	5.00	4.9	4.95

6.6.3 Participants' Evaluation of The Benefits of The System for Developing Their Own Skills

In the evaluation part (as shown in Appendix F), the participants thought about the benefits of the system to develop their thinking process on solving problems. Also, all participants filled their evaluation of the system using point Likert scale from 1 to 20 points. In addition to that, the participants should explain their reasons for given this score.

As we can see in Table 6.11 , the average of experimental group of the each questions is high than control group. Furthermore, The participants found the guided meta-cognitive feedback is more useful than provided ideal answer.

Table 6.11: The evaluation the benefits of the system and the average of both groups

The average of	Experimental group	Control group
Motivating their own thinking process on solving problem	15.82	14
Assisting to find their own strengths and weaknesses on problem solving skills	13.91	14
Improving your problems solving skills	16.18	13.9
How much Ms. Sarah feedback assist you to develop your skills in the future	17.27	15.1

6.6.4 Participants' Opinion About The Most Positive and Negative Aspects in The System

There are two open questions to explain what the most positive and negative aspects based on the participants' perspectives. Most participants in both groups answered that there is no negative aspect of the system, but some participants found that the system has limited number of questions and there are some deficiencies in the system without explaining more.

In terms of positive aspects, these are some answers of some participants such as assisting to improve my programming problem-solving skills, helping to find their own mistakes, easy using the system, and the user interface is friendly.

Chapter 7

Conclusions and Future work

The main goal of this chapter is to present the conclusion of our work. Also, it presents the main contributions of this work and the limitations of it. The last section displays some further suggestions for future research.

7.1 Conclusion

The main goal of our research is to enhance the meta-cognitive skills of beginner programmers in order to improve problem-solving skills. We build a computer-based learning environment. It combines learning by teaching technique and guided meta-cognitive support. A novice programmer has the opportunity to teach Amy which is a virtual agent that plays the tutee role. There is another virtual agent (Ms. Sarah) who plays a mentor agent that can evaluate Amy's solution and provide guided meta-cognitive feedback to the novice programmer in order to re-teach Amy correctly. As it is known, the ultimate goal of all the educational systems is to facilitate learning of students and to stimulate their learning interest. For that, we developed a sequence of activities in the system whose emphasis is on problem-solving skills and meta-cognitive support.

Furthermore, our system covers the limitation of all of these existing systems that focus on various domains and none of them have considered programming problem-solving. In addition to that, the majority of the existing systems did not provide meta-cognitive support. They only focus on providing feedback about the content such as providing correct answer whereas our system provides meta-cognitive feedback to novice programmers.

For that, we conducted an experiment in order to investigate the effect of these two types of feedback on novice programmers' problem-solving skills and their approach to teach the teachable agent (Amy). We conducted a study to evaluate the effectiveness of the meta-cognitive support provided by the system.This study consisted of three sessions. They were not on consecutive days to measure the long-term retention of the acquired skills especially metacognition. So, the study took a place at the College of Computer at Al-lieth in Umm Al-Qura University. We analysed the data gathered from novice programmers, 18-25 years old, studied and who have passed at least one of the courses of programming.

The results show the meta-cognitive feedback had a positive effect on the novice programmers' skills comparing when the performance on the pre-test, post-test and delayed test are analysed. At the end of the post-test, 82% of the experimental group participants improved whereas it was limited to 30% for the control group. At the end of the delayed post-test, 70% in experimental group improved whereas only 50% of the control group improved. Furthermore, the results revealed that the average of KMA score for both groups has increased gradually. That means they have ability to monitor their knowledge. In other words, most of the time they make correct judgements about their have ability to provide correct solutions. In terms of improvements related to adjusting the novice programmers' approach for teaching Amy, the performance of the experimental group is slightly higher than that of the control group. The analysis revealed that experimental group participants acquired some strategies such as looking back, monitoring and control their teaching process, and their ability to solve the problem.

7.2 Research Contributions

The results of this thesis contributed to an intersection of different disciplines: education, psychology and computer science particularly novice programmers in solving problems. This research benefits novice programmers to improve their problem solving before starting to write the source code in any programming language.

The major contributions of this work can be obtained from four perspectives:

1. Identification of the benefits of learning by teaching technique for novice programmers.
2. Development of instructional design system that includes the activities that support the learning by teaching approach and guided meta-cognitive feedback.
3. Comparing the two types of feedback. The first is to focus on the content of solving the problem. The second type of feedback is to enhance the meta-cognitive skills of those programmes.
4. Knowing the effect of these two types of feedback on the performance of those programmes.

7.3 Limitations of the Work

This project was an interdisciplinary project encompassing computer science, education and psychology; i.e. we as computer scientists needed to work closely with education and psychology experts in order to make sure that we meet the educational requirements of the novice programmers in designing the system. Working in between these two disciplines resulted in some conflicts, especially in the different ways of reporting the results of the study.

Furthermore, the project was done at the University of Adelaide in Australia whereas the experiment was conducted in Saudi Arabia. Keeping the two teams together in two different countries as well as two different disciplines required a lot of time and energy.

It was difficult to execute the experiment in Australia as all participants studies English as the second language. We recognize that our experiment has some limitations concerning its design. Firstly, the number of participants in the experiment was small. It was very difficult to gather volunteers for the experiment due to the issue of having more than one session and these sessions were not conducted on consecutive days.

An additional factor that reduced the power of the experiment was the limited number of problems in the system. This was not possible due to time constraints.

Another limitation which is not related to the experiment, but to the design of the system is that the types of problems presented in the system were not tailored to each student's skills of programming problem-solving. Thus, some participants found these problems in the system are more challenging during the experiment. However, the majority of participants tried to solve them even if they did not provide complete solutions. Thus, we believe that some of the results may have been influenced by these limitations.

7.4 Future Work

Even though the results of this study revealed many interesting findings regarding the effectiveness of applying the learning by teaching approach, there is more space for improving this work in the future in terms of improving AmyBrain or Ms.SarahBrain.

We can develop Ms. Sarah's Brain to understand the behavior of the human students and the style of learning. That could help to provide more suitable feedback to the human student. In this case, we need to read more about psychology field.

Also, we can develop Ms. Sarah to communicate and interact more with Amy and human students to enrich the way of learning. Also, we can let human students to communicate with her. In this case, we need natural language techniques.

Also, we can improve Amy's Brain, so that can make the relationship between all the previous problems in order to improve her knowledge. However, we need be careful in this case because Amy could be excel her teacher who is supposed to be a novice programmer. Especially, our goal is to to improve the human students skills not Amy because we consider Amy as a reflection of the students to show their skills level.

Another suggestion which is not related to the Ms. Sarah's Brain and Amy's Brain is that we can develop the system to have more challenging programming problems. Furthermore, we can allow human students to write their own problems to teach Amy. Furthermore, we can develop the system, so that it can provide suitable resources to assist the human students to improve their programming skills and teaching process.

Another venue for further research is to make the system adaptive to support each individual student. Currently, each student receives the problems from our system. In other words, the students cannot provide their own programming problems to teach Amy.

Bibliography

[1] Douglas J Hacker, John Dunlosky, and Arthur C Graesser. *Metacognition in educational theory and practice*. Routledge, 1998.

[2] Sandra Y Okita and Daniel L Schwartz. Learning by teaching human pupils and teachable agents: The importance of recursive feedback. *Journal of the Learning Sciences*, 22(3):375–412, 2013.

[3] Michael de Raadt, Mark Toleman, and Richard Watson. Training strategic problem solvers. *ACM SIGCSE Bulletin*, 36(2):48–51, 2004.

[4] Jason Tan, Gautam Biswas, and D Schwartz. Feedback for metacognitive support in learning by teaching environments. In *Proceedings of the 28th Annual Meeting of the Cognitive Science Society, Vancouver, Canada*, pages 828–833. Citeseer, 2006.

[5] Camilla Kirkegaard. *Adding Challenge to a Teachable Agent in a Virtual Learning Environment Adding Challenge to a Teachable Agent in a Virtual Learning Environment*. PhD thesis, Linköping University Electronic Press, 2016.

[6] Lena Pareto, Magnus Haake, Paulina Lindström, Björn Sjödén, and Agneta Gulz. A teachable-agent-based game affording collaboration and competition: Evaluating math comprehension and motivation. *Educational Technology Research and Development*, 60(5):723–751, 2012.

[7] Noboru Matsuda, Evelyn Yarzebinski, Victoria Keiser, Rohan Raizada, William W Cohen, Gabriel J Stylianides, and Kenneth R Koedinger. Cognitive anatomy of tutor learning: Lessons learned with simstudent. *Journal of Educational Psychology*, 105(4):1152, 2013.

[8] Chih-Yueh Chou and Tak-Wai Chan. Reciprocal tutoring: design with cognitive load sharing. *International Journal of Artificial Intelligence in Education*, 26(1):512–535, 2016.

[9] Gautam Biswas, Thomas Katzlberger, John Bransford, Daniel Schwartz, et al. Extending intelligent learning environments with teachable agents to enhance learning. In *Artificial Intelligence in Education*, pages 389–397, 2001.

[10] Michael J Hogan, Christopher P Dwyer, Owen M Harney, Chris Noone, and Ronan J Conway. Metacognitive skill development and applied systems science: A framework of metacognitive skills, self-regulatory functions and real-world applications. In *Metacognition: Fundaments, applications, and trends*, pages 75–106. Springer, 2015.

[11] Yun-Jo An and Li Cao. Examining the effects of metacognitive scaffolding on students' design problem solving and metacognitive skills in an online environment. *Journal of Online Learning and Teaching*, 10(4):552, 2014.

[12] John H Flavell. Metacognition and cognitive monitoring: A new area of cognitive–developmental inquiry. *American psychologist*, 34(10):906, 1979.

[13] Gregory Schraw and Rayne Sperling Dennison. Assessing metacognitive awareness. *Contemporary educational psychology*, 19(4):460–475, 1994.

[14] Barry J Zimmerman and Dale H Schunk. Reflections on theories of self-regulated learning and academic achievement. *Self-regulated learning and academic achievement: Theoretical perspectives*, 2:289–307, 2001.

[15] John G Borkowski, Martha Carr, and Michael Pressley. "spontaneous" strategy use: Perspectives from metacognitive theory. *Intelligence*, 11(1):61–75, 1987.

[16] Martha Carr, Beth E Kurtz, Wolfgang Schneider, Lisa A Turner, and John G Borkowski. Strategy acquisition and transfer among american and german children: Environmental influences on metacognitive development. *Developmental Psychology*, 25(5):765, 1989.

[17] Chris D Frith. The role of metacognition in human social interactions. *Phil. Trans. R. Soc. B*, 367(1599):2213–2223, 2012.

[18] Reza Pishghadam and Gholam Hassan Khajavy. Intelligence and metacognition as predictors of foreign language achievement: A structural equation modeling approach. *Learning and Individual Differences*, 24:176–181, 2013.

[19] R Swartz and C McGuinness. Developing and assessing thinking skills. final report part 1, 2014.

[20] Claudia Amado Gama. *Integrating metacognition instruction in interactive learning environments*. University of Sussex, 2005.

[21] Diane F Halpern. *Critical thinking across the curriculum: A brief edition of thought & knowledge*. Routledge, 2014.

[22] Eugene Bardach and Eric M Patashnik. *A practical guide for policy analysis: The eightfold path to more effective problem solving*. CQ press, 2015.

[23] Elizabeth C McNie. Reconciling the supply of scientific information with user demands: an analysis of the problem and review of the literature. *Environmental science & policy*, 10(1):17–38, 2007.

[24] H Lee Swanson. Influence of metacognitive knowledge and aptitude on problem solving. *Journal of educational psychology*, 82(2):306, 1990.

[25] Annemieke E Jacobse and Egbert G Harskamp. Towards efficient measurement of metacognition in mathematical problem solving. *Metacognition and Learning*, 7(2):133–149, 2012.

[26] Robert J Sternberg and Peter A Frensch. *Complex problem solving: Principles and mechanisms*. Psychology Press, 2014.

[27] Fadi P Deek and James A McHugh. Problem solving and cognitive foundations for program development: an integrated model. *Information and communication technology*, 2003.

[28] Susan Bergin, Ronan Reilly, and Desmond Traynor. Examining the role of self-regulated learning on introductory programming performance. In *Proceedings of the first international workshop on Computing education research*, pages 81–86. ACM, 2005.

[29] M Havenga. Problem-solving processes in computer programming: a case study. *SACLA, Ballito, KwaZulu–Natal, SA (July 6-8)*, 2011.

[30] Margaret Bernard and Eshwar Bachu. Enhancing the metacognitive skill of novice programmers through collaborative learning. In *Metacognition: Fundaments, Applications, and Trends*, pages 277–298. Springer, 2015.

[31] Siti Nurulain and Mohd Rum. *A metacognitive support environment for novice programmer using semantic web*. PhD thesis, University of Malaya, 2016.

[32] Paul Vickers. *How to think like a programmer: problem solving for the bewildered*. Cengage Learning EMEA, 2008.

[33] Elliot Soloway and James C Spohrer. *Studying the novice programmer*. Psychology Press, 2013.

[34] Essi Lahtinen, Kirsti Ala-Mutka, and Hannu-Matti Järvinen. A study of the difficulties of novice programmers. In *Acm Sigcse Bulletin*, volume 37, pages 14–18. ACM, 2005.

[35] Minjie Hu, Michael Winikoff, and Stephen Cranefield. Teaching novice programming using goals and plans in a visual notation. In *Proceedings of the Fourteenth Australasian Computing Education Conference-Volume 123*, pages 43–52. Australian Computer Society, Inc., 2012.

[36] Richard E Mayer. Cognitive, metacognitive, and motivational aspects of problem solving. *Instructional science*, 26(1):49–63, 1998.

[37] Anneli Eteläpelto. Metacognition and the expertise of computer program comprehension. *Scandinavian Journal of Educational Research*, 37(3):243–254, 1993.

[38] G Polya. How to solve it princeton univ. *Press, Princeton, NJ*, 1945.

[39] PH Winne and DL Butler. Student cognition in learning from teaching. *International encyclopedia of education*, 2:5738–5775, 1994.

[40] Lev Semenovich Vygotsky. *Mind in society: The development of higher psychological processes*. Harvard university press, 1980.

[41] George A Brown, Joanna Bull, and Malcolm Pendlebury. *Assessing student learning in higher education*. Routledge, 2013.

[42] Fred Paas, Alexander Renkl, and John Sweller. Cognitive load theory and instructional design: Recent developments. *Educational psychologist*, 38(1):1–4, 2003.

[43] Valerie J Shute. Focus on formative feedback. *ETS Research Report Series*, 2007(1), 2007.

[44] Susanne Narciss. Designing and evaluating tutoring feedback strategies for digital learning environments on the basis of the interactive tutoring feedback model. *Digital Education Review*, (23):7–26, 2013.

[45] Sally Brown and Peter Knight. *Assessing learners in higher education*. Psychology Press, 1994.

[46] Janet G Donald. Intellectual skills in higher education. *Canadian Journal of Higher Education*, 15(1):53–68, 1985.

[47] Hyeon Woo Lee. *The effects of generative learning strategy prompts and metacognitive feedback on learners' self-regulation, generation process, and achievement*. The Pennsylvania State University, 2008.

[48] Deborah L Butler and Philip H Winne. Feedback and self-regulated learning: A theoretical synthesis. *Review of educational research*, 65(3):245–281, 1995.

[49] John W Jacobs and John V Dempsey. Simulation and gaming: Fidelity, feedback, and motivation. *Interactive instruction and feedback*, pages 197–227, 1993.

[50] David Dunning, Kerri Johnson, Joyce Ehrlinger, and Justin Kruger. Why people fail to recognize their own incompetence. *Current directions in psychological science*, 12(3):83–87, 2003.

[51] Krittaya Leelawong and Gautam Biswas. Designing learning by teaching agents: The betty's brain system. *International Journal of Artificial Intelligence in Education*, 18(3):181–208, 2008.

[52] John A Bargh and Yaacov Schul. On the cognitive benefits of teaching. *Journal of Educational Psychology*, 72(5):593, 1980.

[53] Catherine Chase. Teachable agents and the protégé effect: Increasing the effort towards learning catherine chase, doris b. chin, marily oppezzo, & daniel l. schwartz stanford university.

[54] Krittaya Leelawong, Karun Viswanath, Joan M Davis, Gautam Biswas, Nancy Vye, Kadira Belynne, and John D Bransford. Teachable agents: Learning by teaching environments for science domains. In *IAAI*, pages 109–116, 2003.

[55] Gautam Biswas, Daniel Schwartz, and John Bransford. Technology support for complex problem solving: From sad environments to ai. In *Smart machines in education*, pages 71–97. MIT Press, 2001.

[56] Alice F Artzt and Eleanor Armour-Thomas. A cognitive model for examining teachers' instructional practice in mathematics: A guide for facilitating teacher reflection. *Educational Studies in Mathematics*, 40(3):211–235, 1999.

[57] Michelene TH Chi, Stephanie A Siler, Heisawn Jeong, Takashi Yamauchi, and Robert G Hausmann. Learning from human tutoring. *Cognitive Science*, 25(4):471–533, 2001.

[58] Christoph Rensing, Sara de Freitas, Tobias Ley, and Pedro J Muñoz-Merino. Open learning and teaching in educational communities. In *EC-TEL*. Springer, 2014.

[59] Byron Reeves and Clifford Nass. How people treat computers, television, and new media like real people and places. *CSLI Publications and Cambridge*, 1996.

[60] J Leon and M Fisher. The use of virtual characters to generate teachable moments. In *Museums and the Web*, 2006.

[61] Daniel L Schwartz, Catherine Chase, Doris B Chin, Marily Oppezzo, Henry Kwong, Sandra Okita, Gautam Biswas, RD Roscoe, Hogyeong Jeong, and JD Wagster. Interactive metacognition: Monitoring and regulating a teachable agent. *Handbook of metacognition in education*, pages 340–358, 2009.

[62] Gautam Biswas, James R Segedy, and Kritya Bunchongchit. From design to implementation to practice a learning by teaching system: Betty's brain. *International Journal of Artificial Intelligence in Education*, 26(1):350–364, 2016.

[63] Gautam Biswas, Krittaya Leelawong, Daniel Schwartz, Nancy Vye, and The Teachable Agents Group at Vanderbilt. Learning by teaching: A new agent paradigm for educational software. *Applied Artificial Intelligence*, 19(3-4):363–392, 2005.

[64] James R Segedy, John S Kinnebrew, and Gautam Biswas. The effect of contextualized conversational feedback in a complex open-ended learning environment. *Educational Technology Research and Development*, 61(1):71–89, 2013.

[65] Lena Pareto. A teachable agent game engaging primary school children to learn arithmetic concepts and reasoning. *International Journal of Artificial Intelligence in Education*, 24(3):251–283, 2014.

[66] Noboru Matsuda, Victoria Keiser, Rohan Raizada, Arthur Tu, Gabriel Stylianides, William Cohen, and Kenneth Koedinger. Learning by teaching simstudent: Technical accomplishments and an initial use with students. In *Intelligent tutoring systems*, pages 317–326. Springer, 2010.

[67] Ailiya Borjigin. *Affective teachable agent in virtual learning environment*. PhD thesis, 2014.

[68] Noboru Matsuda, Cassondra L Griger, Nikolaos Barbalios, Gabriel J Stylianides, William W Cohen, and Kenneth R Koedinger. Investigating the effect of meta-cognitive scaffolding for learning by teaching. In *International Conference on Intelligent Tutoring Systems*, pages 104–113. Springer, 2014.

[69] David Nichols. *Intelligent Student Systems: an application of viewpoints to intelligent learning environments*. PhD thesis, Lancaster University, 1993.

[70] Jean Hayes-Michie and Telephone Nos. Learning by teaching. In *Scandinavian Conference on Artificial Intelligence 89: Proceedings of the SCAI'89, Tampere, Finland, 13-15 June, 1989*, volume 4, page 307. IOS Press, 1989.

[71] Sung-il Kim, Sung-Hyun Yun, Mi-sun Yoon, Yeon-hee So, Won-sik Kim, Myung-jin Lee, Dong-seong Choi, and Hyung-Woo Lee. Design and implementation of the kori: Intelligent teachable agent and its application to education. *Computational Science and Its Applications–ICCSA 2005*, pages 191–197, 2005.

[72] Richard R Burton and John Seely Brown. An investigation of computer coaching for informal learning activities. *International Journal of Man-Machine Studies*, 11(1):5–24, 1979.

[73] Sigmund Tobias and Howard T Everson. Knowing what you know and what you don't: Further research on metacognitive knowledge monitoring. 2002.

[74] J Osborne. Measuring metacognition in the classroom: A review of currently-available measures. *Unpublished manuscript*, 1998.

[75] Kevin Downing, Richard Ho, Kristina Shin, Lilian Vrijmoed, and Eva Wong. Metacognitive development and moving away. *Educational Studies*, 33(1):1–13, 2007.

[76] Paul R Pintrich et al. A manual for the use of the motivated strategies for learning questionnaire (mslq). 1991.

[77] David A Cook, Warren G Thompson, and Kris G Thomas. The motivated strategies for learning questionnaire: score validity among medicine residents. *Medical education*, 45(12):1230–1240, 2011.

[78] James R Lewis. Ibm computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *International Journal of Human-Computer Interaction*, 7(1):57–78, 1995.

[79] Jurek Kirakowski and Mary Corbett. Sumi: The software usability measurement inventory. *British journal of educational technology*, 24(3):210–212, 1993.

Appendix A

Pre- test Part 1 and Part 2

Test (Part 1)

Participant ID: __________________

Note: your responses to this survey will be kept confidential and anonymous

Read each problem and provide your answer for each one (Yes or No) in next of the problem. Please do not solve the problems.

#	**Problem**	**Do you think you can solve the given problem correctly?**	
1	Write a pseudocode that reads two numbers and prints the sum of the integers up	[ ] Yes	[ ] No
2	Write pseudo code that tells a user that the number they entered is not a 5 or a 6	[ ] Yes	[ ] No
3	Write a pseudocode that performs the following: Read entered a number. If the number is between 0 and 10, print the word blue. If the number is between 10 and 20, print the word red. if the number is between 20 and 30, print the word green. If it is any other number, write that it is not a correct colour option.	[ ] Yes	[ ] No
4	Write pseudocode that reads five numbers and counts the total of them and then prints result.	[ ] Yes	[ ] No

Test (Part 2)

Participant ID: __________________

Instruction: Read and solve each given problem below of it. There are four problem and time limit is 30 minutes to solve all the given problems. The experimenter will tell you when the time's up. You can solve the given problems in any order.

The box below illustrates what is Pseudocode. Please read it before starting.

Pseudocode is an artificial and informal language that helps programmers develop algorithms. It is a "text-based" detail (algorithmic) design tool. The rules of Pseudocode are reasonably straightforward. All Statements showing "dependency" are to be indented. These include WHILE, DO, FOR, IF and SWITCH.

Problems:

1. Write a pseudocode that reads two numbers and prints the sum of the integers up

2. Write pseudo code that tells a user that the number they entered is not a 5 or a 6

3. Write a pseudocode that performs the following: Read entered a number. If the number is between 0 and 10, print the word blue. If the number is between 10 and 20, print the word red. if the number is between 20 and 30, print the word green. If it is any other number, write that it is not a correct colour option.

4. Write pseudocode that reads five numbers and counts the total of them and then prints result.

Appendix B

Post- test Part 1 and Part 2

Test (Part 1)

Participant ID: ___________________

Note: your responses to this survey will be kept confidential and anonymous

Read each problem and provide your answer for each one (Yes or No) in next of the problem. Please do not solve the problems.

#	**Problem**	**Do you think you can solve the given problem correctly?**	
1	Write a pseudocode that reads two numbers and divide them together and prints the result.	[ ] Yes	[ ] No
2	Write a pseudocode that calculate absolute value and print the result	[ ] Yes	[ ] No
3	Write a pseudocode that reads three integers from the user and determines the largest value.	[ ] Yes	[ ] No
4	Write a pseudocode that rearranges the order of numbers: so that all even number on the right hand and all the odd numbers on the left hand and print how many even numbers and odd numbers.	[ ] Yes	[ ] No

Test (Part 2)

Participant ID: ___________________

Instruction: Read and solve each given problem below of it. There are four problem and time limit is 40 minutes to solve all the given problems. The experimenter will tell you when the time's up. You can solve the given problems in any order.

The box below illustrates what is Pseudocode. Please read it before starting.

Pseudocode is an artificial and informal language that helps programmers develop algorithms. It is a "text-based" detail (algorithmic) design tool. The rules of Pseudocode are reasonably straightforward. All Statements showing "dependency" are to be indented. These include WHILE, DO, FOR, IF and SWITCH.

Problems:

1. Write a pseudocode that reads two numbers and divide them together and prints the result.

2. Write a pseudocode that calculate absolute value and print the result

3. Write a pseudocode that reads three integers from the user and determines the largest value

4. Write a pseudocode that rearranges the order of numbers: so that all even numbers on the right hand and all the odd numbers on the left hand and print how many even numbers and odd numbers.

Appendix C

<u>Delayed- test Part 1 and Part 2</u>

Test (Part 1)

Participant ID: __________________

Note: your responses to this survey will be kept confidential and anonymous

Read each problem and provide your answer for each one (Yes or No) in next of the problem. Please do not solve the problems.

#	**Problem**	**Do you think you can solve the given problem correctly?**	
1	Write pseudocode to convert distance from kilometre to meter. Hints: 1Kilometre = 1000 Meter	[ ] Yes	[ ] No
2	Write a pseudocode that reads the number of hours worded and calculate wages. Note: Standard hours in a work week is 40 hours and regular pay rate for one hour is 10.25$ and overtime pay rate for one hour is 14.5$.	[ ] Yes	[ ] No
3	Write a pseudocode that reverses the letters of a word. Hints-The reserved words of (abcd) is (dcba)	[ ] Yes	[ ] No
4	Read number and print the first numbers in Fibonacci sequence. The Fibonacci sequence, 1,1,2,3,5,8,13…. begins with two 1's, and each successive number is the sum of the preceding two numbers (e.g., 5+8=13)	[ ] Yes	[ ] No

Test (Part 2)

Participant ID: ___________________

Instruction: Read and solve each given problem below of it. There are four problem and time limit is 40 minutes to solve all the given problems. The experimenter will tell you when the time's up. You can solve the given problems in any order.

The box below illustrates what is Pseudocode. Please read it before starting.

Pseudocode is an artificial and informal language that helps programmers develop algorithms. It is a "text-based" detail (algorithmic) design tool. The rules of Pseudocode are reasonably straightforward. All Statements showing "dependency" are to be indented. These include WHILE, DO, FOR, IF and SWITCH.

Problems:

1. Write pseudocode to convert distance from kilometre to meter.

   Hints: 1Kilometre = 1000 Meter

2. Write a pseudocode that reads the number of hours worded and calculate wages. Note: Standard hours in a work week is 40 hours and regular pay rate for one hour is 10.25$ and overtime pay rate for one hour is 14.5$.

3. Write a pseudocode that reverses the letters of a word. Hints-The reserved words of (abcd) is (dcba)

4. Write a pseudocode that read number and print result in Fibonacci sequence. The Fibonacci sequence, (1,1,2,3,5,8,13….) begins with two 1's, and each successive number is the sum of the preceding two numbers (e.g., 5+8=13)

Appendix D

This Appendix shows all the ideal solutions for all problems in Pre-Test, Post-Test and Delayed-Test.

Problems in pre-Test:

1. **Write a pseudocode that reads two numbers and prints the sum of the integers up**

```
1. Read num1 , num2
2. Sum = num1 + num2
3. Print Sum
```

2. **Write pseudocode that tells a user that the number they entered is not a 5 or a 6**

```
1. Read num
2.  IF (num == 5)
    a. Print "your number is 5"
3. Else IF (num == 6)
    a. Print "your number is 6"
4. Else Print "your number is not 5 or 6"
5.
```

Other solution

```
1. Read num
2. IF( num == 5 || num = =6)
    a. Print "your number is a 5 or 6"
3. Else
    a. Print "your number is not 5 or 6"
```

3. **Write a pseudocode that performs the following: Read entered a number. If the number is between 0 and 9, print the word blue. If the number is between 10 and 20, print the word red. if the number is between 21 and 30, print the word green. If it is any other number, write that it is not a correct colour option.**

```
1. Read Num
2. If (Num >=0 and Num <= 9)
    a. Print “Blue”
3. else If (Num >=10 and Num <= 20)
    a. Print “Red”
4. else If (Num >=21 and Num <= 30)
    a. Print “Green”
5. else
    a. Print "not a correct colour option”
```

6. **Write pseudocode that reads five numbers and counts the total of them and then prints result.**

```
1. Read num1,num2,num3,num4, num5
2. Result = (num1+ num2+num3+num4,+num5)
3. Print
```

Other solution

```
1. Initialize result =0
2. For  i=1 to 5 with step=1
    a. Read num
    b. Result +=num
3. Print Result
```

Problems in Post-Test:

1. **Write a pseudocode that reads two numbers and divide them together and prints the result.**

```
1. Read num1 , num2
2. IF num2 ==0
    a. Print Error
3. Else
    a. Result = num1/num2
4. Print Result
```

2. **Write a pseudocode that calculate absolute value and print the result**

```
1. Read num
2. IF num < 0
    a. Result = num * -1
3. Else
    a. Result = num
4. Print Result
```

3. **Write a pseudocode that reads three integers from the user and determines the largest value**

```
1. Read num1, num2, num3
2. If (num1 >= num2  &&  num1 >= num3)
    a. Print num1
3. Else
    a. If (num2 >= num1  &&  num2 >= num3)
        i. Print num2
    b. Else
        i. If (num3 >= num1  &&  num3 >= num2)
       ii. Print num3
```

4. **Write a pseudocode that rearranges the order of numbers: so that all even numbers on the right hand and all the odd numbers on the left hand and print how many even numbers and odd numbers.**

```
1. Read Numbers
2. ListEvenNumbers={}
3. Initialize countEven =0
4. ListOddNumbers ={}
5. Initialize countOdd =0
6. ListResult={}
7. For i=0 to Number.length-1 with step=1
    a. If (Number[i] %2 ==0)
        i. ListEvenNumbers.add(Number[i])
        ii.
    b. Else
        i. ListOddNumbers.add(Number[i])


8. ListResult = ListOddNumbers + ListEvenNumbers
9. Print ListResult , countEven , countOdd
```

Problems in Delayed-Test:

1. **Write pseudocode to convert distance from kilometre to meter. Hints: 1Kilometre = 1000 Meter**

```
1. Read num1
2. result = num1 * 1000
3. Print result
```

2. **Write a pseudocode that reads the number of hours worded and calculate wages. Note: Standard hours in a work week is 40 hours and regular pay rate for one hour is 10.25$ and overtime pay rate for one hour is 14.5$.**

```
1. Read HoursWor
2. IF (HoursWor >=1 * HoursWor <=40)
    a. Result = HoursWor * 10.25
3. Else
    a. IF (HoursWor > 40)
        i. Result = ((HoursWor-40) * 14.5 )+ ( 40 * 10.25)
```

3. **Write a pseudocode that reverses the letters of a word. Hints-The reserved words of (abcd) is (dcba)**

```
1. Read word
2. Result ={}
```

```
3. For  i= word.length-1 downto 0 with step =1
    a. Result = word[i]
4. Print Result
```

4. Write a pseudocode that Read number and print the results in Fibonacci sequence. The Fibonacci sequence, (1,1,2,3,5,8,13….) begins with two 1's, and each successive number is the sum of the preceding two numbers (e.g., 5+8=13)

```
1. Read num
2. IF num <=1
    a. Return num
3. Initilize result =1;
4. Initilize pervious =1;
5. For i=2 to num with step =1
    a. Temp = result
    b. Result += pervious
    c. Pervious = temp
6. Print result
```

Appendix E

Motivated Strategies for Learning Questionnaire	
This survey will help us to understand metacognitive awareness with in programming problem course. I will be very grateful if you can spend some of your precious time completing the given questionnaire. Your responses to this survey will only be used for academic purposes.	
Participant ID:	
Section 1: Personal and Education Background	
Your Age	○ 18 – 25 years
	○ 26 – 30 years
	○ 31 – 35 years
Gender	○ Female
	○ Male
Year of Study in 2017	○ The first year
	○ The second year
	○ The third year
	○ The fourth or fifth year
	○ Graduate
Have you studied at least one of programming course?	
○ Yes	
○ No	
Which course have you studies? (you can choses more than one chose)	
○ Introduction to computer Science (3007101-3)	
○ Computer Programming (3007103-3)	
○ Structural Programming (3007204-3)	
○ Advanced Programming (3007205-3)	
○ Logical Programming (3007317-3)	
○ Other Programming Course	
Have you passed at least one of the programming course?	
○ Yes	
○ No	

Section 2: The motivation and learning strategy section

Please rate the following items based on your behaviour in **most recent programming course you have taken**. Your rating should be on a 7- point scale where **1= not at all true of me to 7=very true of me**.

To illustrate

Not at all true of me	1	2	3	4	5	6	7	Very tr ue of me

The Motivation part

1	In a class like this, I prefer course material that really challenges me so I can learn new things.	1	2	3	4	5	6	7
2	The most satisfying thing for me in this course is trying to understand the content as thoroughly as possible.	1	2	3	4	5	6	7
3	The most important thing for me right now is improving my overall grade point average, so my main concern in this class is getting a good grade.	1	2	3	4	5	6	7
4	I am very interested in the content area of this course.	1	2	3	4	5	6	7
5	If I don't understand the course material, it is because I didn't try hard enough.	1	2	3	4	5	6	7
6	I'm confident I can understand the most complex material presented by the instructor in this course.	1	2	3	4	5	6	7
7	I have an uneasy, upset feeling when I take an exam.	1	2	3	4	5	6	7

Learning strategies

8	When studying for this class, I read my class notes and the course readings over and over again.	1	2	3	4	5	6	7
9	I try to relate ideas in this subject to those in other courses whenever possible.	1	2	3	4	5	6	7
10	When I study for this class, I pull together information from different sources, such as lectures, readings, and discussions.	1	2	3	4	5	6	7
11	When I study the readings for this course, I outline the material to help me organize my thoughts.	1	2	3	4	5	6	7
12	I ask myself questions to make sure I understand the material I have been studying in this class.	1	2	3	4	5	6	7

	To illustrate 1 2 3 4 5 6 7							
	Not at all true of me Very tr ue of me							
13	When studying for this course I try to determine which concepts I don't understand well.	1	2	3	4	5	6	7
14	When I study for this class, I set goals for myself in order to direct my activities in each study period.	1	2	3	4	5	6	7
15	When I become confused about something I'm reading for this class, I go back and try to figure it out.	1	2	3	4	5	6	7
16	If course materials are difficult to understand, I change the way I read the material.	1	2	3	4	5	6	7
17	Even when course materials are dull and uninteresting, I manage to keep working until I finish.	1	2	3	4	5	6	7
18	When studying for this course, I often try to explain the material to a classmate or a friend.	1	2	3	4	5	6	7
19	Even if I have trouble learning the material in this class, I try to do the work on my own, without help from anyone.	1	2	3	4	5	6	7
20	I make good use of my study time for this course.	1	2	3	4	5	6	7
21	Whenever I read or hear an assertion or conclusion in this class, I think about possible alternatives.	1	2	3	4	5	6	7
22	I believe that programming problem solving can be learned by the computer based system.	1	2	3	4	5	6	7

Appendix F

Evaluation and Usability Questionnaire

Participant ID:

Note: your responses to this survey will be kept confidential and anonymous

This survey is part of a study to evaluate the usability of our system. By answering the next questions, you will help us to understand your perception on the usability of the system. Please answer all the questions.

		strongly agree	agree	Slightly agree	Non	Slightly disagree	disagree	strongly disagree
1	I believe that programming problem solving can be learned by the computer based system.	○	○	○	○	○	○	○
2	I would recommend this system to my friends.	○	○	○	○	○	○	○
3	I would like to use this system frequently.	○	○	○	○	○	○	○
4	I would find the system useful in my studies for problem solving.	○	○	○	○	○	○	○
5	It was easy to learn to use this system.	○	○	○	○	○	○	○
6	The interface of this system is pleasant.	○	○	○	○	○	○	○
7	The organization of information on the system screens is clear.	○	○	○	○	○	○	○
8	It is easy to make the system do exactly what I want.	○	○	○	○	○	○	○
9	I feel comfortable using this system.	○	○	○	○	○	○	○
10	The way that Ms. Sarah's information is presented is clear and understandable.	○	○	○	○	○	○	○
11	The way that Amy's information is presented is clear and understandable.	○	○	○	○	○	○	○

12 To what extent this system motivates your thinking process on solving programming problems?

Low | | | High

○○○○○○○○○○○○○○○○○○○○

Please explain your reasons

13 To what extent this system helps you to find your strengths and weaknesses on problem solving skills?

Low | | | High

○○○○○○○○○○○○○○○○○○○○

Please explain your reasons

14	To what extent working on this system improves your problems solving skills?
	Low | | | High
	○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○
	Please explain your reasons
15	Based on your use of the system Ms. Sarah provides feedback in order to re-teach Amy, How much this feedback assist you to develop your skills in the future?
	Low | | | High
	○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○
	Please explain your reasons
16	List the most **positive** aspect(s) about the system
17	List the most **negative** aspect(s) about the system

Appendix G

Number of Participants that used in the experiment

#	Number of Experimental group	Number of Control Group
1	E241700	C250001
2	E241702	C250003
3	E241704	C250005
4	E241706	C250007
5	E241708	C250009
6	E241710	C250011
7	E241712	C250013
8	E241714	C250015
9	E241716	C250017
10	E241718	C250019
11	E241720	C250021
12	E241722	C250023
13	E241724	C250025
14	E241726	C250027
15	E241728	C250029
16	E241730	C250031
17	E241732	C250033
18	E241734	C250035
19	E241736	C250037
20	E241738	C250039

Appendix H

Consent Form

Thank you for agreeing to take part in this experiment. You will be used our system and answer some questionaries. Also, we will give you information how to use the system. The aim of this paper is to get your approval to take part in the experiment.

Your names will be kept confidential and anonymous. Also, any other your response will be used for academic purpose.

Furthermore, you can withdraw at any time or refuse to answer any question without any consequences of any kind even if you agree to participate. Finally, we will welcome to answer any questions about the study.

I read and understood all the above information. I hereby consent to participate in this study.

Name:

Date:

Signature:

Appendix I

Experiment Materials

This appendix contains all materials that are used in the main experiment in our research for the three sessions

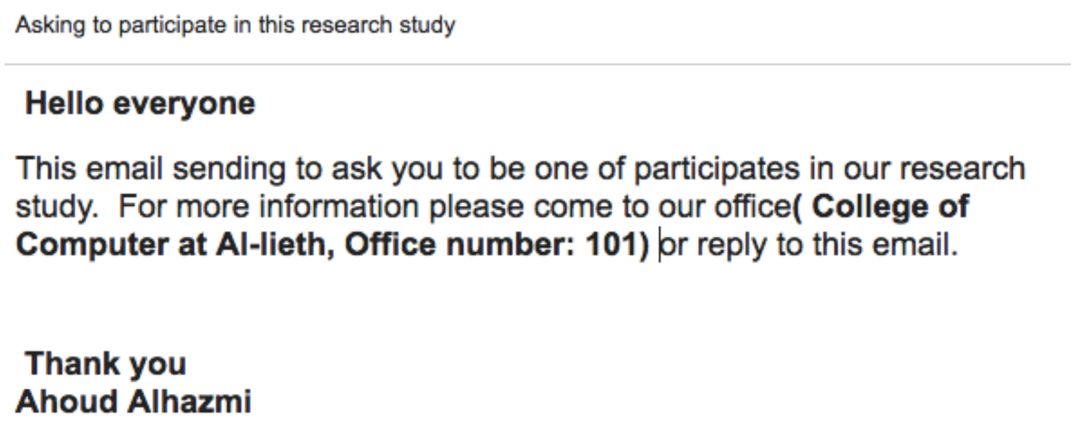
The materials are presented in the following order for all the three sessions:

1. Email that sending to inform the students
2. Consent Form.
3. Motivated Strategies for Learning Questionnaire
4. Video for how to use system
5. Pre-test: Part I and Part II.
6. Post-test: Part I and Part II.
13. Evaluation Questionnaire
14. Delayed-test: Part I and Part II.

Appendix J

Email that sending to inform the students

This appendix shows the email which sent to the group of computer sciense students


Asking to participate in this research study

Hello everyone

This email sending to ask you to be one of participates in our research study. For more information please come to our office**(College of Computer at Al-lieth, Office number: 101)** or reply to this email.

Thank you
Ahoud Alhazmi

Appendix K

Amy feedback and inquiry

This table shows Amy feedback and inquiry

	The tutor behaviour	Amy feedback
1	Asking for teaching her	Hi, teacher *tutorName*. my name is Amy and I'm your student. Could you please teach me how to solve this problem
2	No Understanding such user write not pseudocode	I am sorry. I could not understand. Please teach me before you send me to take the quiz"
3	Requesting Amy to take quiz with Teaching	Thank you for teaching me, you can send me to take quiz"
		Alright, what problem should I try !
		Thanks, to ensure I understand, you can send me to take quiz
		Thanks for teaching, OK. Let me know what I should do next
4	Requesting Amy to take quiz without Teaching	Excuse me. You have not taught me anything. Please teach me before you send me to take the quiz"
5	mysterious variable name such as x in line 3	What do you mean by 'x' variable in line 3?
		What is the purpose of' x' variable in line 3?
6	Ambiguous some actions in Pseudocode foe example in lines 3 ,5 and 10	Excuse me! I couldn't undertand what do you mean by actions in lines 3 ,5 and 10. Please teach me them before sending me to the quiz"
		I'm sorry. I couldn't understand what is the purpose of actions in lines3 ,5 and 10. Please teach me them before sending me to the quiz"

Appendix L

Amy explanations

This table shows some examples about the Amy explanations what she learns from the user after re-teaching correctly.

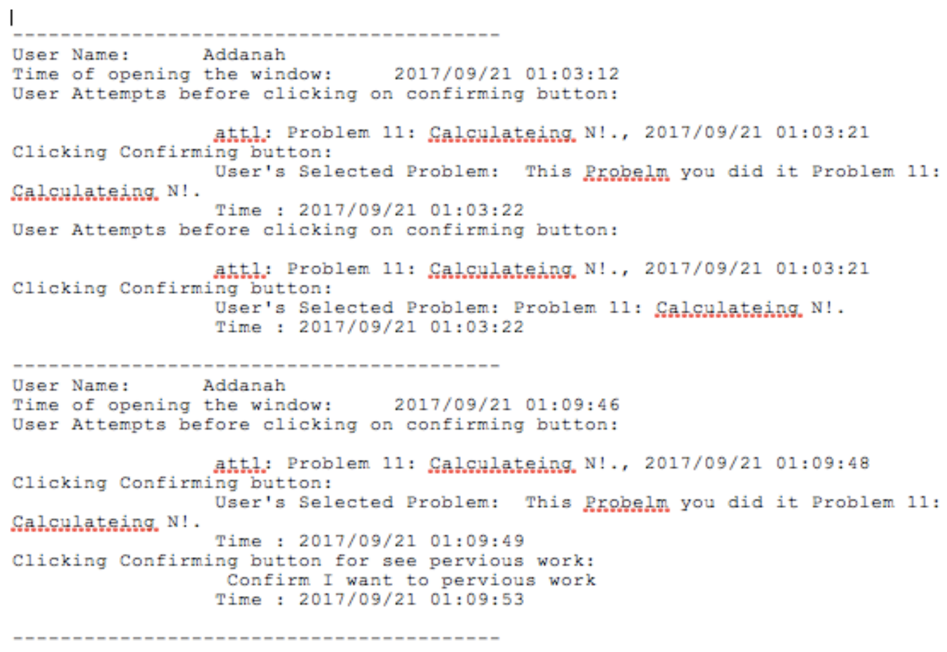
#	**The tutor's behaviour correctly in re-teaching stage**	**Amy explanation**
1	Identified input of the problem such as how to convert minutes to hour	Thank you for teaching me, I have to determine the input variable that means I have to read minutes' number.
2	Identified input of the problem such as finding the smallest number	I have to determine the output variable that means I have to print smallest value
3	Knew how to calculate the average temperature of seven days	I should divide the total of the temperature of seven days by 7
4	Put the solution steps in correct ordering of the problem such as checking the number is positive or negative	I should put the steps of the problem in correct ordering. First I have to read the inputs, then check the number is greater than zero then print positive or negative if the number less than zero.
5	Broke down the problem correctly for example, checking the number is positive or negative	I have to break down a problem into some steps. In this problem we have check about the number if the number greater than zero or less than zero.
6	Fix the logical error in condition	I have to verify about conditional expressions depending on the problem
7	Separated various part of conditions	I should separate the two conditions if the number is positive or negative
8	If she used switch incorrect place, then the tutor fix this problem by using If statement.	I should use IF statement because Using Switch must be a constant expression and this problem has rang for that it needs to be solve by using IF statement
9	Teaches her correctly about how to convert a binary number to a decimal number	Thank you for teaching me. Now I know about how to convert binary to decimal number. start with the binary numbers and multiply each binary digit by its weight. Then after multiplying all the digits, I will add the results. Is my explanation correct?
10	Fix the bugs start or end of in repetition	I have to check the bugs in the loop by looking back to the start and end value depending on the problem.

11	Fix the bugs in increment part of repetition	I should check increment value of the loop depending on the problem.
12	Fix the value of initialize	Next time i will be careful about the value of initialize
13	Re-teach her how to calculate x^y	I should knew how to calculate x power y, where "x" is the number I want to multiply and "y" is the power I want to multiply it by.

Appendix M

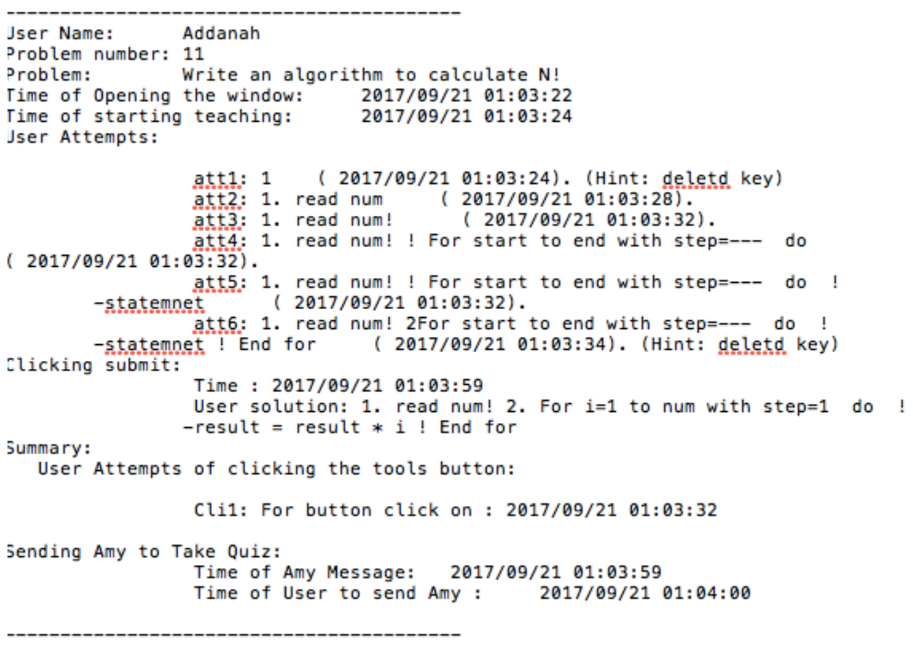
Some Screenshots from Log files

1. Log files for selection the problem from homepage



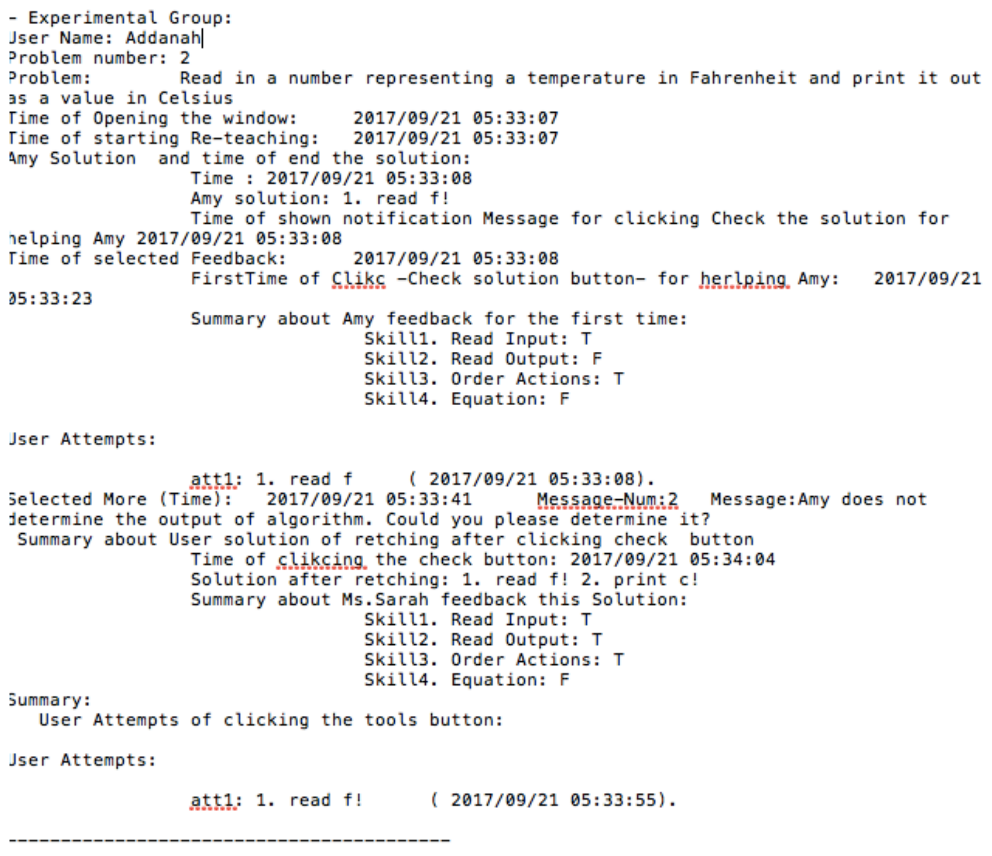
```
-----------------------------------------
User Name:      Addanah
Time of opening the window:     2017/09/21 01:03:12
User Attempts before clicking on confirming button:

                 att1: Problem 11: Calculateing N!., 2017/09/21 01:03:21
Clicking Confirming button:
                 User's Selected Problem:  This Probelm you did it Problem 11:
Calculateing N!.
                 Time : 2017/09/21 01:03:22
User Attempts before clicking on confirming button:

                 att1: Problem 11: Calculateing N!., 2017/09/21 01:03:21
Clicking Confirming button:
                 User's Selected Problem: Problem 11: Calculateing N!.
                 Time : 2017/09/21 01:03:22

-----------------------------------------
User Name:      Addanah
Time of opening the window:     2017/09/21 01:09:46
User Attempts before clicking on confirming button:

                 att1: Problem 11: Calculateing N!., 2017/09/21 01:09:48
Clicking Confirming button:
                 User's Selected Problem:  This Probelm you did it Problem 11:
Calculateing N!.
                 Time : 2017/09/21 01:09:49
Clicking Confirming button for see pervious work:
                  Confirm I want to pervious work
                 Time : 2017/09/21 01:09:53

-----------------------------------------
```

2. Log files for teaching Amy



```
-----------------------------------------
Jser Name:      Addanah
Problem number: 11
Problem:        Write an algorithm to calculate N!
Time of Opening the window:     2017/09/21 01:03:22
Time of starting teaching:      2017/09/21 01:03:24
Jser Attempts:

                att1: 1    ( 2017/09/21 01:03:24). (Hint: deletd key)
                att2: 1. read num     ( 2017/09/21 01:03:28).
                att3: 1. read num!      ( 2017/09/21 01:03:32).
                att4: 1. read num! ! For start to end with step=---  do
( 2017/09/21 01:03:32).
                att5: 1. read num! ! For start to end with step=---  do  !
        -statemnet      ( 2017/09/21 01:03:32).
                att6: 1. read num! 2For start to end with step=---  do  !
        -statemnet ! End for     ( 2017/09/21 01:03:34). (Hint: deletd key)
Clicking submit:
                Time : 2017/09/21 01:03:59
                User solution: 1. read num! 2. For i=1 to num with step=1  do  !
               -result = result * i ! End for
Summary:
   User Attempts of clicking the tools button:

                Cli1: For button click on : 2017/09/21 01:03:32

Sending Amy to Take Quiz:
                Time of Amy Message:   2017/09/21 01:03:59
                Time of User to send Amy :    2017/09/21 01:04:00

-----------------------------------------
```

3. Log files for Quiz stage



```
- Experimental Group:
User Name: Addanah
Problem number: 2
Problem:        Read in a number representing a temperature in Fahrenheit and print it out
as a value in Celsius
Time of Opening the window:     2017/09/21 05:33:07
Time of starting Re-teaching:   2017/09/21 05:33:07
Amy Solution  and time of end the solution:
                 Time : 2017/09/21 05:33:08
                 Amy solution: 1. read f!
                 Time of shown notification Message for clicking Check the solution for
helping Amy 2017/09/21 05:33:08
Time of selected Feedback:      2017/09/21 05:33:08
                 FirstTime of Clikc -Check solution button- for herlping Amy:   2017/09/21
05:33:23
                 Summary about Amy feedback for the first time:
                                 Skill1. Read Input: T
                                 Skill2. Read Output: F
                                 Skill3. Order Actions: T
                                 Skill4. Equation: F

User Attempts:

                 att1: 1. read f     ( 2017/09/21 05:33:08).
Selected More (Time):   2017/09/21 05:33:41      Message-Num:2   Message:Amy does not
determine the output of algorithm. Could you please determine it?
 Summary about User solution of retching after clicking check  button
                 Time of clikcing the check button: 2017/09/21 05:34:04
                 Solution after retching: 1. read f! 2. print c!
                 Summary about Ms.Sarah feedback this Solution:
                                 Skill1. Read Input: T
                                 Skill2. Read Output: T
                                 Skill3. Order Actions: T
                                 Skill4. Equation: F
Summary:
   User Attempts of clicking the tools button:

User Attempts:

                 att1: 1. read f!     ( 2017/09/21 05:33:55).

-----------------------------------------
```