A FRAMEWORK TO SUPPORT THE TEACHING OF
INTRODUCTORY PROGRAMMING

RANDAVE SINGH GILL

SUBMITTED IN FULLFILMENT OF THE
REQUIREMENT FOR THE DEGREE OF MASTER OF
COMPUTER SCIENCE

FACULTY OF COMPUTER SCIENCE
UNIVERSITY OF MALAYA
KUALA LUMPUR

June 2006
Abstract

Teaching introductory programming is still seen to be a very challenging in today’s world of computing education. Many key factors such as high failure rates in first year programming courses, low student retention numbers in programming streams and the reluctance of students to pursue advance programming courses provide indication of the difficulties faced by educators and students. Educators are constantly finding ways to improve the various elements relating to the teaching and learning of the subject in order to reduce or eliminate the challenges being faced. One example of these initiatives which is currently being studied and applied in various institutions, is the need to understand the student’s cognitive characteristics in order to develop a more individualized and specific learning environment. Other forms of benefits resulting from the paradigm shift of traditional (objectivism) to newer and more effective teaching models (constructivist model) as well as the proliferation of various Internet based support tools are also being measured and piloted accordingly. It is with this tone that this study aims develop a support framework that will improve the process of teaching introductory programming. The development and implementation of this framework is expected to enhance the teaching process hence creating a more conducive learning environment. The formation of this framework will be based on the findings and conclusions gathered from various initiatives and research from all around the world as well as from the lecturers and students within FSKTM.
Acknowledgement

I would like to thank to my supervisor Associate Professor Salimah Mokhtar for her guidance and support throughout the entire course of this dissertation. I would also like to convey my gratitude to all staff of and students of FSKTM for their cooperation and participation in all relevant stages within this research. Last but not least my deepest appreciation to my family for their unconditional understanding and support.
# Table of Contents

Abstract ..................................................................................................................2  
Acknowledgement .................................................................................................3  
List of Tables ..........................................................................................................7  
List of Figures .........................................................................................................9  
Chapter 1 ..................................................................................................................9  
  1.0 Introduction .....................................................................................................10  
  1.1 Statement of Problem ....................................................................................12  
  1.2 Purpose of Study ............................................................................................13  
  1.3 Research Questions .........................................................................................15  
  1.4 Significance of Study ......................................................................................20  
  1.5 Definition of terms ........................................................................................21  
  1.6 Research Limitation .......................................................................................22  
Chapter 2 – Literature Review .............................................................................23  
  2.0 Literature Review ............................................................................................23  
    2.1 Summary of literature of review ....................................................................44  
Chapter 3 – Methodology ......................................................................................48  
  3.0 Introduction .....................................................................................................48  
  3.1 Population .......................................................................................................48  
  3.2 Sample .............................................................................................................48  
  3.3 Instrumentation .................................................................................................49  
  3.4 Procedures ......................................................................................................51  
    3.5 Data Analysis .................................................................................................52  
Chapter 4 – Findings ..............................................................................................56  
  4.0 Introduction .....................................................................................................56  
  4.1 Response Rate .................................................................................................56
List of Tables

Table 2.1.1 – Summary of Literature Review

Table 3.1.1 – Instrumentation Data Sheet (Interview Schedule)

Table 3.1.2 – Instrumentation Data Sheet (Questionnaires)

Table 3.5.1 – Response from Lecturers on Questions on Preferred Learning Styles

Table 3.5.2 – Response from Lecturer on Questions on Motivation Type

Table 3.5.3 – Response from Lecturer on Questions on Prior Experience

Table 3.5.3 – Response from Lecturer on other Techniques Used

Table 3.5.4 – Impact of APS Table Layout

Table 3.5.5 – Advantages/Disadvantages of Tutors Table Layout

Table 4.1.2 – Response rate for sample SMP001

Table 4.1.2 – Response rate for sample SMP002

Table 4.1.3 – Demographic data for SMP001

Table 4.3.1 - Summary of Responses from Teachers (a) Who Determine The Preferred Learning Style Of A Student, (b) How Do They Determine The Type Of Learner A Student Is, (c) Why is it Important/Not Important To Determine The Type of Learner A Student Is.

Table 4.3.2 - Summary of Responses from Teachers (a) Who Determine The Type Of MOTIVATION A Student Has, (b) How Do They Determine The Type Of MOTIVATION A Student has, (c) Why is it Important/Not Important To Determine The Type of MOTIVATION A Student Has.

Table 4.3.3 – How lecturers would use the above information (type of motivation and learner) to improve the process of teaching introductory programming if it were to be made available to them

Table 4.3.4 – Responses from lecturer (a) if student with prior experience perform better and (b) the importance of having prior experience in programming
Table 4.3.5 - Comments from teacher on each constructivist strategies.........................68
Table 4.3.6 - Comments on any other techniques that are used to improve the process of teaching introductory programming.............................................................................................................69
Table 4.3.7 - Lecturers responses on the impact of having an “Alternate Program Sequence” as part of the teaching environment.................................................................85
Table 4.3.8 - Lecturers responses on the impact of having an “Introduction of tutors” as part of the teaching environment. ...................................................................................87
Table 5.1.1 – Adopted Strategies........................................................................................94
Table 5.1.2 – Poor Learning Tendencies (PTL) follow up action ...................................96
Table 5.1.3 – Motivation type questionnaires........................................................................97
Table 5.1.4 – APS guidelines...............................................................................................100
List of Figures

Figure 3.5.2 – Type of constructivist strategies / experience.................................53

Figure 3.5.3 – Type of constructivist strategies/ impact on academic / motivation level
........................................................................................................................................53

Figure 3.5.5 – Type of constructivist strategies / academic / involvement rate........54

Figure 4.3.5 - Student responses on the impact of each constructivist technique (has or
will have) on the MOTIVATION level in learning the subject.................................76

Figure 4.3.6 - Responses of student who have experienced the different types of support
tools applied in their introductory programming subjects.....................................80

Figure 4.3.7 - Student responses on the impact of each support tool (has or will have) on
the RESULTS of test, assignment, tutorial, quiz or examination.............................81

Figure 4.3.8 - Student responses on the impact of each support tool (has or will have) on
the INVOLVEMENT level in learning the subject.........................................................82

Figure 5.1.1 – Teaching Support Framework Diagram.............................................103
Chapter 1

1.0 Introduction

Teaching and learning programming at an introductory level often forms a fundamental section of a tertiary education in computing. Many computing undergraduates begin to embark in programming as early as in the first year of education. This presents an exciting time ahead for students (learners) as the subject introduces various concepts and methods which are new and key to the discipline of Computer Science. The way to teach introductory programming is also well documented and understood. Material can be organized neatly into the syllabus of a lecture course. A typical course would start by a lecturer teaching the mechanics of creating running program, followed by introducing concepts of programming one at a time and concluding with some consideration of issues of style and debugging. No experience academic would be daunted by the prospect of lecturing this material (Jenkins and Davy, 2000).

However in contrary to the common perception much of the literature on computer programming education shows teaching and learning computer programming at an introductory level is still a challenging task. This is seen in high failure rates in first year programming courses, low student retention numbers in programming streams and, for many students who successfully proceed to higher level programming courses, a lack of understanding of basic programming concepts (Carbone et al. 2001, Jenkins, 2001). It is also common for students to approach their final year project work determined to avoid programming at all costs, presumably because they either cannot program or believe that they can not (Carter and Jenkins, 1999).
This state of affairs are concerning as computer programming should not be a subject which student shy away from especially in today’s day and age where it is common to find household appliances that are programmable and computerized searches and office applications that requires the used of logical operations (Prasad and Fielden, 2002). The lack of understanding of programming concepts, poor grades scores in examination and the fear of pursuing subjects or carrier paths that require basic programming knowledge creates perennial problem that continues to challenge educators of this discipline.

It is with this essence that this study aims to develop a support framework that will improve the process of teaching introductory programming thus indirectly creating a more conducive learning environment. This framework will be developed by first summarizing the contributing factors that lead to the challenges faced by educators and undergraduates in teaching and learning introductory programming as well as the best practices and methods applied by educators to which positive and encouraging results in delivering the required knowledge has been achieved. All this valuable information will be obtained based on an in-depth analysis of the current literature review and related references. This will be followed by administrating a series of interviews and questionnaires to the educators and undergraduates of the introductory programming at the Faculty of Computer Science (FSKTM) in University Malaya (UM) with the objective of measuring the effectiveness and acceptance of the best practices and methods to improve the teaching of introductory programming in FSKTM. Finally a system prototype will be developed to better facilitate the support framework.
1.1 Statement of Problem

In this section the problem statement is further expanded by first reiterating the challenges faced by educators of introductory programming and then followed by an explanation of the contributing factors that influences these problems.

As described in the previous section the factors below reinforce the theory that teaching and learning introductory programming is still a challenging task:-

a. High failure rates in first year programming courses.

b. Low student retention numbers in programming streams.

c. For many students who successfully proceed to higher level programming courses, a lack of understanding of basic programming concepts.

d. It is also common for students to approach their final year project work determined to avoid programming at all costs, presumably because they either cannot program or believe that they can not.
1.2 Purpose of Study

The previous sections detailed the background and the problem statement of this study. With various elements and challenges evident it seems rather impossible to find the one single “silver bullet” to this problem. Maybe there isn’t a single solution to this problem, but rather only a set of practical steps and techniques coupled with the used of innovative technological implementation to ensure that the process of delivering the required knowledge from the educator to student is done with a higher and improved success rate. In more specific terms, this study is aimed to develop support framework to aid educators in constructing a learning environment that will allow them to better deliver the required knowledge by providing a suite of resources that bundles traditional pedagogical approaches such as lectures and tutorials with the innovations of new teaching approaches together and with the support of Internet based technology.

The construction of the framework will be based on the following steps that are explained below:-

a. Based on the literature review summarize the various initiatives that includes the problem areas and corresponding solutions made to improve the process of teaching introductory programming conducted by various educators and faculties of learning institution.

b. This will be followed by a study on the lecturers and undergraduates involved in introductory programming at the Faculty of Computer Science (FSKTM) in University Malaya (UM). This study is aimed to investigate effectiveness and acceptance of these methods and will be measured using a series of interviews and questionnaires.
c. Based on the findings from lecturers and student in the FSKTM, a support framework will be developed to improve the process of teaching introductory programming at the Faculty of Computer Science (FSKTM) in University Malaya (UM). This will be followed by a system prototype to better facilitate the support framework.
1.3 Research Questions

a. **Research Question 1:**

   (i) What are the benefits of understanding student cognitive characteristics such as learning styles and motivation?

   (ii) How can this understanding be applied by an educator to improve the teaching of introductory programming?

**Key Terms:** cognitive characteristic, learning style, motivation

**Justification:**

   (i) The first question aims to obtain the views from FSKTM lectures on the benefits of understanding a student’s cognitive characteristics such as preferred learning styles and motivation. The data from this finding will reinforce the need to have assessment exercises in place to enable an understanding of student cognitive characteristic be obtained.

   (ii) The aims as well as the data form the finding from the second question will determine how the information on the student cognitive characteristics be used to improve the teaching of introductory programming.

**Sample:** FSKTM lecturers with more that 1 year experience teaching introductory programming as part of UM curriculum.

**Technique:** Questionnaire, Interview Schedule

**Method:** Survey Research
b. **Research Question 2:**

(i) What are the constructivist strategies that have been implemented by the teaching staff for the students of introductory programming in FSKTM?

(ii) What are the perceptions from the teaching staff on the effectiveness of implementing constructivist strategies to enhance the process of teaching introductory programming?

(iii) How do students of introductory programming in FSKTM feel about the use of constructivist strategies as a teaching method and its impact to their motivational level and academic performance?

**Key Terms:** constructivist strategies, introductory programming, teaching staff

**Justification:**

(i) This first question aims to investigate the current constructivist strategies that have been implemented in FSKTM. The data from these finding will enable an understanding on the current constructivist strategies that have been applied as well as its success in improving the teaching of introductory programming.

(ii) The second question aims to investigate the perception of the teaching staff on the effectiveness of the constructivist strategies. The data from this finding will measure the effectiveness of different types of constructivist strategies and help plan and device the implementation for the students of introductory programming in FSKTM.

(iii) The third questions aims understand the student perceptions of constructivist strategies as a teaching method and its impact to their motivational level and academic performance. The data from this finding will enable the better planning on the selection of constructivist strategies that are student are more inclined to participate in.
Sample:
(i) FSKTM lecturers with more than 1 year experience teaching introductory programming as part of UM curriculum.
(ii) FSKTM undergraduates who are currently or have prior experience learning introductory programming as part of UM curriculum.

Technique:
(i) Interview Schedule and Questionnaire
(ii) Interview Schedule and Questionnaire
(iii) Interview Schedule Questionnaire

Method:
(i) Survey Research
(ii) Survey Research
(iii) Survey Research
c. **Research Question 3:**

(i) What are the perceptions from students on the impact of introducing application support tools to their learning environment in order to improve the academic grade scores and the level of student involvement?

**Key Terms:** application support tools, student involvement, student achievement

**Justification:** The question aims to investigate the opinions of students on the impact of introducing application support tools to improve the level of student academic scores and involvement. The data from this finding will help create an environment where students are more comfortable to participate in the learning process outside the classroom hence indirectly creating a positive impact to their overall achievement.

**Sample:**

(i) FSKTM undergraduates who are currently or have prior experience learning introductory programming as part of UM curriculum.

**Technique:**

(i) Questionnaire, Interview Schedule

(ii) Questionnaire, Interview Schedule

**Method:** Survey Research
d. **Research Question 4:**

   (i) What are the opinions from the lecturers in regards to the introduction of tutors and alternative program sequence to improve the process of learning teaching introductory programming?

**Justification:** This question aims to investigate the opinions from FSKTM lecturers on the changes teaching environment currently used in introductory programming. Data collected from the findings will help to create a better understanding on how the teaching environment should be organized and supplemented.

**Key Terms:** teaching environment

**Sample:** FSKTM lecturers with more than 1 year experience teaching introductory programming as part of UM curriculum.

**Technique:** Questionnaire, Interview Schedule

**Method:** Survey Research
1.4 Significance of Study

As the title depicts the main objective of this study is to develop a support framework which will improve the process of teaching introductory programming. This framework will consist of a suite of resources that bundles traditional pedagogical approaches such as lectures and tutorials with the innovations of new teaching approaches together and with the support of Internet based technology. The significance of this study is divided into four key areas. Firstly, the awareness and understanding of student cognitive characteristics will provide invaluable knowledge for educators to better engage, adapt and respond to the various levels of a student’s learning needs in introductory programming. Educators will be more responsive and alert to the difficulties faced by students while learning the subject and to also proactively react in the event if the teaching methods that are employed are either working or will not work. Secondly, the perception of the teaching staff and students on the types and impact of various constructivist strategies that have been practiced and applied in introductory programming will be understood. This will enable a validation to be made on the current as well as the future practices of these strategies in FSKTM. Thirdly, these strategies can be extended with a clear identification on the type and implementation methods of application support tools that improve student involvement and motivational level. This data will provide valuable information into the process of planning, selection and development of application support tools that will inherently improve the teaching of introductory programming. Finally, further information on the changes needed as well as the enabling improvement in the teaching environment will be obtained. Areas such as format of the teaching deliverables, organization of student in the classroom, pacing of subject contents will be investigated among the relevant teaching staff. Data provided for this section will enable a justification on how the teaching environment should be organized and supplemented accordingly.
1.5 Definition of terms

a. **Application support tools:** ICT tools that can be used to improve the process of learning introductory programming. These tools are functional in either an online or offline basis.

b. **Constructivist strategies:** A theory of learning where knowledge is actively constructed by the students through various meaningful activities and collaboration with peers and not passively absorbed from textbooks and lecture. Examples of these strategies are discussion, pair programming, group based classes, visualization tool and problem based learning.

c. **Feel:** The opinion of students.

d. **Introductory programming:** Subjects that are foundational to understanding and learning computer programming.

e. **Learning process:** The acquisition and development of knowledge in introductory programming.

f. **Students:** FSKTM undergraduates who are currently or have prior experience learning introductory programming as part of UM curriculum.

g. **Student achievement:** An attained level of expertise visible from higher grades scores and student performance on a task.

h. **Student involvement:** Participation rate either individually or in groups within and out of the classroom.

i. **Teaching staff:** FSKTM lecturers with more than 1 year experience teaching introductory programming as part of UM curriculum.

j. **Teaching style:** The methods, techniques and activities used to transfer knowledge from teachers to student.

k. **Teaching environment:** The environment which consist of the approach and format where the curriculum is taught.
1.6 Research Limitation

a. Due to the small/unique sample available for the study, results may not be generalized beyond the specific population from which the sample was drawn.

b. Due to lack of empirical data on suitable choice of programming language to teach introductory programming, this study will not consider the selection of the programming language to be used in the overall framework. This decision will be left at the discretion of the faculty and respective teaching staff.

c. Due to the failure of sample respondents to answer with candor, results might not accurately reflect the opinions of all members of the included population.
Chapter 2 – Literature Review

2.0 Literature Review

Extensive literature has been formulated in an approach to improve the teaching and learning of introductory programming. This literature review starts by re-examining the problems that are associated with the teaching and learning of introductory programming and followed by a critical review on the existing or previous work attempted to solve the problem.

As described in the introduction and statement of problem section of this study, the theory that teaching and learning introductory programming is still a challenging task and this is seen in high failure rates in first year programming courses, low student retention numbers in programming streams and, for many students who successfully proceed to higher level programming courses, a lack of understanding of basic programming concepts (Carbone et al. 2001, Jenkins, 2001). It is also common for students to approach their final year project work determined to avoid programming at all costs, presumably because they either cannot program or believe that they can not (Carter and Jenkins, 1999). While it is true that computer programming teaching has evolved drastically from the early days with theory now being heavily supported with practice, and innovative teaching methods and delivery modes being put into the curriculum, the concept that programming is for a ‘certain type’ of people is still paramount. There is still a common belief that programming is a different way of thinking (Turkle and Papert, 1990), and that only people who are analytical, logical thinkers and planners have a real chance of learning how to program successfully.
When analyzed further the contributing factors that results in this challenges being apparent are explained in the following section.

Firstly, educators often overlook the need to understand the cognitive characteristics of students that consist of various psychological aspects such as personality type, preferred learning styles, poor learning tendencies and student motivation in learning the subject. This oversight often results in many students having difficulties in keeping up with the material that is being presented as the teaching methods and resources used are not individualized to each learning environment. An understanding of student’s cognitive characteristics is vital to enable educators to better pace, structure and deliver the required knowledge. (Prasad and Fielden, 2002).

Secondly significant research has been conducted on the relevant teaching models used to teach introductory programming. The question which often arises is should the teaching be based on an objectivist or a constructivist model (Van Gorp and Grissom, 2001). Objectivism often manifest itself in teacher-centered and teacher-controlled classroom, while constructivist focuses more on the learning experience of the student. The traditional teaching style in most tertiary institutions has been wholly objectivist, with the teacher appearing in front of the class to deliver lectures either verbally, and/or by the aid of presentation tools. However, as (Carter and Boyle, 2002) quite correctly pointed out, in a practical subject like IT, students need to be active in the learning process by “interacting with the concepts, testing out theories and being allowed the space to make mistakes”. The constructivist model on the other hand is enforced by combining certain elements of the objectivism model while requiring students to construct knowledge based upon meaningful activities and collaboration with their peers. This paradigm shift from traditional objectivist strategies to a more new and
practical implementation of the constructivist model is seen by many educators to be daunting as it will result in radical changes to the way the subject is taught.

The third key influence is that introductory programming is a subject that encompasses of multiple disciplines such as mathematics, language learning, arts and psychology (Prasad and Fielden, 2002). Its roots in mathematical sciences are well noted where programming requires the same mathematical skills such as problem solving, abstraction and deduction. (Feliesien et al. 2000). The expertise required in learning a new language, i.e. comprehension, reading and writing coupled with the creativity and composition skills apparent in an art discipline creates a challenging path in order to develop a strong understanding in introductory programming. Also programming relates to the field of psychology and communication given the ‘user’ end of the discipline. (Prasad and Fielden, 2002). The multi discipline nature of introductory programming does not do any favors for students who are experiencing a subject of this nature so early into their tertiary education. Most often than not, students are unable to cope with this demand thus resulting in negatives influences such as fear of perusing advance programming subject and the inbuilt mindset of avoiding programming related subject in the later stages of their education.

The understanding of human cognitive characteristics has always had a profound effect when it comes to teaching and learning introductory programming. One key characteristic that often reveals itself and is influential in producing the required results is a student’s motivation to learn and excel in the subject. “To succeed in any academic task, student must be motivated. They must want to succeed” (Jenkins,2001).This is especially true in a practical discipline such as programming where student must be motivated to spend time practicing, even when there is no explicit assessment credit available. Motivation in students can be generally categorized into five types, namely;
extrinsic, intrinsic, social, achievement and null. Students normally have at least one of the five types of motivation which can have a direct impact to their academic performance (Jenkins, 2001). The five different type of motivation are explained in more details below:-

a. **Extrinsic Motivation**

Is where the primary motivator is the career and associated rewards that will follow from the successful completion of the course. Students with this type of motivation often are driven with the lucrative benefits programming can bring.

b. **Intrinsic Motivation**

Is driven by a deep interest in computing (or specially programming) for its own purpose. Student with intrinsic motivation often have a genuine interest and passion for the subject.

c. **Social Motivation**

Is the desire to please some third party whose opinion is valued; one good example would be a student wanting to please their parent by doing well in the subject.

d. **Achievement Motivation**

Is driven with the need to “do well” for personal satisfaction. Students in this category would excel to succeed solely to increase the grade point average.

e. **Null Motivation**

The final type of motivation is “null motivation” which is basically used to accommodate cases falling outside these categories. Such “motivation” would be characterized by statements such as “I just want to pass” (Jenkins,2001).

In coherent views, Robins et al (2002) conducted a study where empirical data on the factors influencing student grades in introductory programming were analyzed. The study revolved around students of first year programming at the University of Otago.
Data were collected during the second semester of 2000 and again in 2001. Students were asked to complete an online survey that mainly consisted of multi-choice questions. The conclusion of the study reveals that achievement motivation – “students who expect to get an A and are willing to say so” was the key predictor in being successful in introductory programming.

Other than motivation being a cognitive characteristic within a student, this drive can be nurtured and developed with the involvement and encouragement of teachers. Today’s changing world of education demands a teacher role to evolve from a person who transmits instructional material set by the curriculum to a motivator who encourages students to learn and engage their curiosity and interest (Jenkins, 2001). Two fundamental factors that influence this process are trust and expectation. An improved level of trust between teacher and student eliminate the invisible barrier that exist and allows room for better flow of communication, “Teachers are figures of authority. Motivators are rather different, and are more like trusted friends. If I am to successfully motivate students, they must, in some sense at least, trust me and I must trust them” (Jenkins, 2001). Expectation on the other hand sets the bar of the required level of deliverables of the subject, “Students and teachers both have expectations at the start of a programming course. Teachers need to be sure of what they expect from the class – they need to understand why the class are there, and what they are hoping for from the course” (Jenkins, 2001).

In addition to motivation, a comprehensive understanding of a student learning style can provide invaluable insight into the way a teacher should structure and transmit the required knowledge. This learning style can viewed from two angles, one is the type of
preferred learning style that a student is more accustomed to and poor learning strategies that student commonly employs.

Carbone et al. (2001) described that students often fall prey to three poor learning strategies namely; superficial attention, impulsive attention and staying stuck. These different types of learning deficiencies are framed by a prior study by Baird and Northfield. Baird identified a series of processing habits in tertiary learners that he called Poor Learning Tendencies (PLT’s). He pointed out that PLT were all characterized by a passive, dependent, uniform approach to learning. Of the processing habits found by Baird that act as a barrier of quality learning, three were considered in the study by (Carbone et al. 2001). An explanation of these poor learning strategies are described below:-

a. **Superficial Attention**

This involves skimming over a communication, with no attempt to actively process the information to generate personal meaning.

b. **Impulsive Attention**

Some parts of communication are attended to but others are overlooked. For example, the learner may focus on an interesting example and ignore the major point.

c. **Staying Stuck**

Lack of any strategy to cope with getting stuck except to call for help. No attempt to return to the instruction, reflect on the strategy selected, analyze what has been done so far or consider alternative.

Carbone et al. (2001) study involved students being given several tasks that will give evidence to all three difference types of PTL. The first task revolved around the Data Structure and Algorithm component of the course. The intention of this task was to give
students the opportunity to familiarize themselves with stacks and experience in implementing these. The task required students to be able to reproduce what has been specifically covered in the lecture with minor modifications. Evidence of superficial attention were found as the subsequent interviews with student revealed that student’s didn’t search for a deep understanding, they were happy to apply their attention superficially and success by copying slabs of code. Even though the task asked students to modify code, students copied the code without thought. Tutors noted that many of their students missed the main concepts and their program were replicas of those provided by the lecturer. Below are some of the comments made by students:

“\textit{I copied the program to create a stack from the lecture notes. I couldn’t understand “pop()” and I didn’t understand the difference between “st\textcolor{red}{top} and s\textcolor{red}{top}”}\n
“\textit{I had managed to get the first four questions working simply by copying sections of the code from my lecture notes. I didn’t really understand what was happening to the frame pointer, stack pointer, parameters and return values. I could just see that it worked, somehow.}”

Carbone et al. (2001) also concluded that ultimately lecturers teaching programming would expect their students to be able to design, implement and test a relatively complex piece of software. It is a common belief that the larger and more complex the code students write, the better programmer they are. Most programming courses are designed to include a laboratory component in which students spend 2-3 hours coding the tasks set. However, if students are asked to write complex pieces of code at too early a stage they can be pushed into adopting a poor learning tendency. It would be a complete change in culture for some academics teaching programming to design some tasks that do not require pure coding. Often smaller tasks can still be used to develop
the students' interest and understanding of the subject. Below is a list of suggested improvement that Carbone et al. (2001) has made to reduce superficial attention:

a. **Not always coding.**

Often students are required to write lines of code, and very rarely are students required to do alternative activities. Getting students to diagrammatically present material often highlights misconceptions that can be addressed immediately. Including tasks that require tracing code, or answering a series of questions, can be used as alternatives to purely writing code.

b. **Rewards for understanding not completing.**

If students were aware that understanding was rewarded, and not copious amount of code, they might be less likely to take a crude approach to completing the task.

c. **Outline a method of attack.**

Without a design students can wander from the intended pathway and ultimately reach a point where the only source of help is seen to be copying extracts of code provided.

d. **Smaller coding questions.**

Introduce questions that don't consume too much of the students' time, so they don’t feel pressured into copying straight from the notes.

The second task resolves around the understanding of functions. The task required students to design a solution to a mathematical problem using functions. In this case coding the solution was not as simple as reproducing the code given in lecturers. The outcome of this study revealed that student focused more on understanding how to perform the mathematics behind the calculation rather than concentrating on invoking the function, parameter passing and returning values. This was validated by comments made by the observing tutors:-
“Finding the powers of numbers is not a common thing student do very often. So they concentrate on getting the calculation right rather than how to use functions.”

“They definitely concentrated on the mathematics rather than programming. Therefore they are not learning what they should be, just spending a lot of time on mathematics”

In an approach to minimize impulsive attention Carbone et al. (2001) proposed the following suggestions:-

a. **Emphasize the key point.**

Usually there are many ways to code a solution to a problem. If the important points are emphasized in the tasks, through the task’s aim and the type of question, impulsive attention might be minimized

b. **Provide adequate resources for the introduction of unfamiliar material.**

Sometimes subjects are so tightly structured that it's not possible to cover everything in lectures. As a consequence, many lecturers introduce new material in the laboratory and tutorial exercises. Often it is not possible to remove material from the subject, leaving some of the material introduced for the first time outside of that environment. While the idea of introducing new material is fine, the resources needed to help students understanding are usually inadequate, rather than carefully planned.

The final task of identifying the evidence of staying stuck were seen with tutor expressing disappointment in their students who they felt progressively become too dependent on the tutor when they were “stuck”. Majority of students were unwilling to solve their problems without constant demonstrator guidance from their tutors and not wanting to “think hard” in solving their problems.
“I was stuck. Pure and simply. I dared not ask the lecturer for help and no one I knew was working on the same part of the program as I was - even if they were, the general impression I received was that they were hiding their secrets with ferocity. I brought up the subject with other students. I was expecting a rush of complaints on how hard it all was- especially the shoot module, but I got nothing. 'Why is no one saying anything?’ I thought, 'Is no one else finding this hard? Did we get the same project?’ The other students lack of concern forced me to plod on. I came to the conclusion that I needed to seek help. There was no way I could think of solving the problem. It was a definite dead end. The only person who I could think of was an older friend who had quite a bit of experience with C programming. I described my problem to him and gave him the project sheet to look at.”

The case above reveals the hesitation by the student to admit that they did not know how to proceed. This appeared to be a common problem amongst many of the weaker students. In this case, the student describes a group assignment given towards the end of semester 1. The intention of the project was to help the students learn to develop software in a team, and to integrate the various aspects of C programming learnt earlier in the semester.

Below is a list of suggested improvements made by Carbone et al. (2001), that can be trialed to tasks to minimize staying stuck:-

a. **Tactics on how to start with graded helps.**

Challenge a student first, don’t explain everything.

b. **Provide useful references and resources.**

Often the only resource students axe aware of are their textbook, lecture notes and the tutor.

c. **Provide guide to writing and testing code in manageable chunks.**
Include debugging strategies. Guides to writing code should be provided rather than providing the code. Build a program in stages, for example, if part of the problem requires file input and output, students should write a small program to ensure they understand how to do it.

In relation to the identification of poor learning strategies, it is also vital for teachers to determine the preferred learning style of students (i.e. “the type of learner a student is”). By identifying the preferred learning style of a student, teachers can start thinking in terms of how to frame various programming tasks so that good learning patterns are adopted and the learning process is improved. (Carbone et al. 2001). To accomplish this (Thomas L et al 2002) conducted a study by using The Felder-Silverman Learning Style Model to determine the preferred learning style of a student and its correlations with academic achievement. Students learning styles were obtained by a multiple choice test that used the Felder-Silverman scale. This test tested students along four axes and grouped them into 16 separate categories (Soloman et al, 2000). The results of this test will enable students to be classified into different categories of learners namely; active learners (learn by trying things out, working with others) , reflective learners (learn by thinking things through, working on their own), sensing learners(concrete, practical, oriented towards facts and procedures), intuitive learners( conceptual, innovative, oriented towards theories and meaning), visual learners(prefer pictures diagram and flow charts), verbal learners (written and spoken explanation), sequential learners (learn in incremental , orderly steps) , global learners (holistic, learn in large gaps) , inductive learners(prefers explanation that move from specific to general) and deductive learners(prefer explanation that moves from general to specific). The key characteristic of this model is its emphasis on the preferred learning style, not ability of students. (Felder and Richard, 1996) who created this model made a profound explanation that notes "A student's learning style or type profile provides an indication
of probable strengths and possible tendencies or habits that might lead to difficulty in academic settings. The profile does not reflect a student's suitability or unsuitability for a particular subject, discipline, or profession."

The purpose here is to identify what may be easy for the students as opposed to the areas in which they may be more challenged. "Functioning effectively in any professional capacity, however, requires working well in all learning style modes. ... If professors teach exclusively in a manner that favors their students' less preferred learning style modes, the students' discomfort level may be great enough to interfere with their learning. On the other hand, if professors teach exclusively in their students' preferred modes, the students may not develop the mental dexterity they need to reach their potential for achievement in school and as professionals. An objective of education should thus be to help students build their skills in both their preferred and less preferred modes of learning."

The outcome of the study conducted by (Thomas L et al 2002) reveal that there is a high possibility that their current education system favors student with a certain learning style preference. It is important scaffolding to student with different learning styles preferences and encouraging students to be aware of their own individual needs and preferences. Another interesting result from the study revealed that active, sensing and visual learners may be particularly at a disadvantage at the current methods of teaching which heavily based on traditional pedagogical approach such as lectures and tutorials.

In a subsequent research, Bruce et al. (2004) reported that there exist five different ways in which students go about learning introductory programming. These are captured in the following categories:-
a. Following – where learning to program is experienced as ‘getting through’ the unit.

b. Coding – where learning to program is experienced as learning to code.

c. Understanding and integrating – where learning to program is experienced as learning to write a program through understanding and integrating concepts.

d. Problem solving – where learning to program is experienced as learning to do what it takes to solve a problem.

e. Participating or enculturation – where learning to program is experienced as discovering what it means to become a programmer.

Subsequent to the cognitive characteristics of students, the learning environment of introductory programming is extremely crucial to the development of the learning process. Jenkins and Davy (2000), reported that in today education system students come from an increasingly wide variety of backgrounds, and with an increasingly wide range of skills and expectations. “Often they encounter teaching programmes that were designed with an entirely different student population in mind. This is especially the case in introductory programming where the teaching and support methods in place often do not meet the needs of an increasingly diverse student body.” In an approach to deal with this diversity, the authors developed a learning support environment where student were classified according to the following categories: - Rocket Scientist, Average, Strugglers and Serious Struggler. These students were classified using a series of amplitude test and informal interviews. Each classification of student was given a separate route to achieve their required grade in the subject. The outcome of the study was a success where it showed fewer students failing than in previous sessions.
Whittington K et al. (2003) also agreed that with such diversity in student backgrounds putting all students through the same curricular experience is not a good solution.

In their study, an improved learning support environment was developed where a new two-course sequence was introduced in a traditional introductory programming subject. The new two-course sequence begins by revisiting topics from the first programming course with which the majority of students had problems, and then covers the same material as our original second course. Students taking this alternative introductory programming sequence end up with one additional course that can be used in their program as a professional elective. The study concluded with positively where both the faculty and students involved in the alternatively paced introductory programming sequence feel that its initial trial was a success. Students experienced less intimidation, and social interaction in the classroom, a key factor in retention, was increased. Student persistence increased as students continued in the programming sequence at a higher rate than students who were only offered the traditional sequence in the previous academic year. And overall performance, in terms of grades in second-step concepts, was higher.

An essential aspect in facilitating the learning support environment is by the introduction of tutors who assist teachers in disseminating the required knowledge. Both Roberts et al. (1995) and Sheard and Hagan (1999) reported positive effects on student participation when qualified and trained tutor were present during the course of the subject. The tutors came in extremely useful during tutorial and out of class activities. A significant amount of workload is also transferred from the teacher to tutor thus allowing better emphasis in executing ones responsibilities.
The learning support environment is further extended with the presence of improved teaching and delivery methods such as constructivist strategies. Even though presenting an exact definition of constructivism is difficult and likely impossible (Phillips, 1995), constructivist classrooms or strategies are often viewed as problem-solving environments manifested through three C’s: context, construction, and collaboration. First, students are to be given problems entrenched in authentic and perhaps simplified contexts. This will provide internal motivation for the student as opposed to external motivation that cognitive objectivists would emphasize. Second, students are to construct knowledge based upon meaningful activities: they cannot be given this knowledge. Lastly, students often collaborate with their peers. This aids the knowledge construction process, as students must seek to examine alternative perspectives on problem solutions and thus possibly re-construct their own perspectives and solutions (Mark and Scott 2001). In their study, Mark and Scott (2001) measured the impact of various type constructivist strategies applied at different frequencies in an introductory programming subject. The types of strategies were namely:

a. **Code Walkthroughs**

In this activity, students step through existing code and predict the output. This helps students practice and better understand flow of control. Responsibilities can be assigned based on the code that is being reviewed. For example, consider a class session on parameter passing that distinguishes between passing by value and passing by reference. A number of methods that have four parameters are provided. Some are passed by reference and some are passed by value. Each group member is responsible for tracing through the code for one parameter. This division of labor helps to keep everyone involved instead of the strongest student taking control of the group.
b. Writing Code

Another activity is to have groups write code to solve a small problem. For example, "write pseudo code to simulate 500 coin tosses and print the number of heads." It is important that the teacher walks around the room and observe the groups working together. Guidance must be provided when appropriate but a preference must be given to remain quiet and allow group members to answer their own questions. Groups are motivated to write a complete and accurate solution since they may be called to share their solution with the rest of the class. A variation of writing code during class is to assign one or two small problems for students to solve before coming to class. They bring their typed solutions to class and break into their groups. Group members walk through each other's code to compare solutions and then collaborate on a group solution. Individual solutions as well as the group solution are then handed in. With this technique, all students must think about the problem on their own before coming to class and also understand the ways others have solved the problem.

c. Scaffolding

This educational concept recognizes that novices need additional support to solve a problem. The idea is simple but important to computer science pedagogy as learning language syntax, code flow, data representation, and appropriate design are daunting tasks for novices. Scaffolding is used in various ways. For example, students are split into groups of two and given code that solves a problem. The code in this example is the scaffolding that students will build upon. Students are then required to insert comments to describe the semantics of the code. Alternatively, teachers may give comments (the scaffolding) to the groups that describe an algorithm. Students then write code that corresponds to the comments. Of course, the ultimate goal is for students to generate all the code and think about efficient solutions, but novices need support in their initial
programming endeavors.

d. Code Debugging

A fifth activity is to give groups of students’ syntactically and logically buggy code. In this way all students can contribute to finding errors. For example, some students, who may be struggling with the course, will find the more obvious syntax and logic errors, while others are challenged to find more subtle errors that are not easily recognized. Sometimes students are asked to do this individually and then split up into groups and share results. At times, constructive thinking is promoted further when students disagree on what is or is not an error in the code.

e. Lecture Note Reconstruction

This technique in class sessions of 50 min as well as 75 min. Here, students are asked to not take notes during a mini lecture but instead to pay close attention. After 15 min of lecture they have a few minutes to reconstruct an outline of the lecture from memory. They then meet in their groups and refine their notes further. This is a great exercise to help students improve their listening skills (Bonwell, 1996).

The study consisted of 6 sections compromising of 32 CS1 students. Each section was taught by a different instructor that utilized the constructivist technique at different frequency. Neither student nor instructors were aware of this study until the end of the term. The initial outcome of the study revealed that two sections has no constructivist activities, one that rarely has activities, two sections that had activities about once per week and one section that had activities almost every day. Once the final exam scores were graded, it revealed a positive correlation between frequency and the mean exam scores. The average grades scores among student increased exponentially as the frequency of the constructivist activities increased.
In addition to the five constructivist techniques described above, another effective technique that has been experimented with is the use of pair programming. This technique is a practice where two programmers work collaboratively at one computer, on the same design, algorithm, or code (Wiebe et al., 2003). Pair programming at its best provides a framework in which students can optimize their efforts in designing and solving programming problems. Partners provide immediate feedback and motivation during the programming process. They also provide an immediate first avenue for answering questions and solving problems while working on the program. It also creates an overall lab environment that keeps attention focused on the problem at hand (Wiebe et al., 2003). In a study conducted in the University of Utah undergraduate Software Engineering course, students were paired together during the entire duration of the semester. Student continuously worked together in all aspects of the curriculum consisting of both theory and practical aspects of the subject. The outcome of the study revealed a higher success rate on the subject where more students were able to keep their overall grades at a C or above. The lab classes provided the most significant result where students show higher level of interaction as actively discussed, questioned and directed and guided each other throughout the lab session. Similarly in a study conducted by (Katira et al., 2004), the impact of pair programming is increased accordingly when students with similar skill levels are paired together.

In a further study to measure the impact of constructivist techniques in improving the teaching and learning of introductory programming, Hagan et al. (1998) introduced a weekly discussion class in addition to the normal lecture and lab work. These discussion classes were conducted in group of 16 or less without the present of any computers. Each class is lead by a trained tutor to provide the necessary guidance.
Several activities were designed to encourage participation from students namely grids, role plays and mimics. Each of these activities derived satisfactory results as students where actively participating in discussion hence encouraging active learning. The overall grades of student improved significantly. It is noted that “for many activities, students were formed into groups of about three, to encourage them to talk. Some students are extremely reluctant to speak in front of the whole class, but willing to contribute to a discussion among the people and ask questions of the tutor if he or she joins in their discussion casually. Student talk is important because it helps students to formulate their own links to prior knowledge, and to fit their understanding of a new topic into their framework of knowledge. The tutor’s role in the discussion classes was not to supply answers to questions but rather to assist students to find their own answers. One function of the discussion class was to create a “need to know” related to the following day’s lecture, in order to stimulate interest in that lecture” (Hagan et al., 1998).

In addition to the various constructivist strategies described in the literature, the next major area when it comes to supplementing the learning environment to improve the teaching of programming is the use and implementation of application support resources and tools. These resources can range from various internet based to standalone desktop applications. One of the most common implementation of these resources is the use of a subject website to structure and store information related to the subject. Some websites also include various courseware and interaction tools to enable offline and online communication between student and teachers. One such implementation of this approach is seen by an experiment conducted (Lockyer and
They developed an online courseware called FOLE which was experimented and used as part of their University learning environment. This courseware consist of features such as online and offline tutor facilities, progress monitoring and student facilities such as discussion boards, forum, email and chat engines among student, tutors and lectures themselves. The results of the experiment revealed the overall learning experience and focus on the subject of the student improved, "In general students seem to have had a broadly positive learning experience which although they found it challenging was nevertheless seen as rewarding and valuable in terms of skills acquisition. Students were also extremely complementary to the module tutors regarding the levels of support they provided. Given the problems with laboratory space and the concerns about coverage of some of the topics it might be worth considering reducing (or replacing?) the formal tutorial sessions with more flexible delivery using the FOLE (once the navigation has been improved). This would have the advantage of freeing up time to look at topics in more depth. All new modules have their teething troubles but this one seems to have managed remarkably well." (Lockyer and Harding, 2001).

Other than the internet based courseware described above, there exist various other support tools that has provided various benefits to teacher and students. One such tool is SOLVEIT that was developed by (Deek and McHugh, 2000). This tool is a prototype of an integrated environment to support student learning programming. Support is provided through all problem solving stages, including formulating a problem, planning and designing a solution, and testing and delivering that solution. The aim of this tool is to help students think more deeply about the problem in question. In addition to the project notebook, SOLVIT also provides a graphics editor that permits students to make
and save drawings / sketches concerning the problem. The information elicitation tool is used to extract relevant information from within the problem description.

This information includes goal, givens, unknowns, conditions and constraints and is stored within the multiple-view reference database. Other support tools such as the use of video clips – multimedia command centre (Gartner, 1997), micro worlds – Karel the Robot (Pattis, 1995), RoboPascal (Carey, 1996) have also seen some success in this domain (Gartner, 1997). Additional support tools to supplement the subsequent stages of learning programming which is design, development and implementation can be illustrated with tools such as FLINT (Crews and Ziegler, n.d), various type of IDE, conventional debuggers and program animator such BRADMAN (Smith and Webb, 2000) and VINCE (Rowe and Thorburn, 2000). These tools enable students to obtain visual view the interpretation of their code or algorithm in a real-time basis (Gartner, 2003).

In summary the key importance of application support tools and resources is that it enables guidance and feedback to be provided to learners in a way that is responsive and sensitive to learner individual needs (McLoughlin, 1998). This creates an environment that is both positive and encouraging to any educational institution.
2.1 Summary of literature of review

In order to facilitate better structure, organization and improved readability of this study, the above section (2.1 Summary of literature review) which is not common in other dissertation has been introduced. The introduction of this section is unique and vital to this study as its sets the stage and the remaining direction for this entire study. This summary will detail out the best practices and methods adopted and tested by educators to which positive and encouraging result in delivering the required knowledge has been achieved. This is further illustrated in the table below:-

<table>
<thead>
<tr>
<th>Teaching and learning programming is still a very challenging task because:</th>
<th>Key factor that influences these challenges:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. High failure rates in first year programming courses.</td>
<td>a. The lack of understanding and attention by educators to student cognitive styles.</td>
</tr>
<tr>
<td>b. Low student retention numbers in programming streams.</td>
<td>b. The reluctance of educators to adopt and adapt to the different types of teaching models being introduced due to the radical changes it will create in the way the subject is taught.</td>
</tr>
<tr>
<td>c. For many students who successfully proceed to higher level programming courses, a lack of understanding of basic programming concepts.</td>
<td>c. The multiple disciplines such as mathematics, language learning, arts and psychology that encompasses the subject.</td>
</tr>
<tr>
<td>d. It is also common for students to approach their final year project work determined to avoid programming at all costs, presumably because they either cannot program or believe that they can not.</td>
<td></td>
</tr>
</tbody>
</table>

Summary of the best practices and approaches adopted and tested by educators to which positive and encouraging result in delivering the required knowledge has been achieved:

a. Studies conducted by Robins et al (2002) and (Jenkins, 2001) where empirical data on the factors influencing student grades in introductory programming were analyzed. The conclusion of the study reveals that motivation – “students who expect to get an A and are willing to say so” was the key predictor in being successful in introductory programming.
b. Research by Carbone et al. (2001) describes that students often fall prey to three poor learning strategies namely; superficial attention, impulsive attention and staying stuck. These different types of learning deficiencies are framed by a prior study by Baird and Northfield. By understanding these tendencies lecturer can better prepare and adapt their teaching to the student needs.

c. Subsequent research by (Carbone et al. 2001) states that in relation to the identification of poor learning strategies, it is also vital for teachers to determine the preferred learning style of students (i.e. “the type of learner a student is”). By identifying the preferred learning style of a student, teachers can start thinking in terms of how to frame various programming tasks so that good learning patterns are adopted and the learning process is improved.

d. (Whittington K et al., 2003) provide empirical data to support that the learning environment of introductory programming is extremely crucial in the development of the learning process as student’s in today education system come with such diversity in backgrounds and with an increasingly wide range of skills and expectations that putting all students through the same curricular experience is not a very good solution. In their study, an improved learning support environment was developed where a new two-course sequence was introduced in a traditional introductory programming subject. The study concluded with positively where both the faculty and students involved in the alternatively paced introductory programming sequence feel that its initial trial was a success. Students experienced less intimidation, and social interaction in the classroom, a key factor in retention, was increased. Student persistence increased as students continued in the programming sequence at a higher rate than students who were only offered the traditional sequence in the previous academic year. And overall performance, in terms of grades in second-step concepts, was higher.

e. In an approach to deal with this diversity, Jenkins and Davy (2000) developed a learning support environment where student were classified according to the following categories: - Rocket Scientist, Average, Strugglers and Serious Struggler. These students were classified using a series of amplitude test and informal interviews. Each classification of student was given a separate route to achieve their
required grade in the subject. The outcome of the study was a success where it showed fewer students failing than in previous sessions.

f. Both Roberts et al. (1995) and Sheard and Hagan (1999) reported positive effects on student participation when qualified and trained tutor were present during the course of the subject. The tutors came in extremely useful during tutorial and out of class activities. A significant amount of workload is also transferred from the teacher to tutor thus allowing better emphasis in executing ones responsibilities.

g. The learning support environment is further extended with the presence of improved teaching and delivery methods such as constructivist strategies. In their study, Mark and Scott (2001) measured the impact of various type constructivist strategies applied at different frequencies in an introductory programming subject. Once the final exam scores were graded, it revealed a positive correlation between frequency and the mean exam scores. The average grades scores among student increased exponentially as the frequency of the constructivist activities increased.

h. In addition to the five constructivist techniques described above, another effective technique that has been experimented with is the used of pair programming. In a study conducted in the University of Utah undergraduate Software Engineering course, students were paired together during the entire duration of the semester. Student continuously worked together in all aspects of the curriculum consisting of both theory and practical aspects of the subject. The outcome of the study revealed a higher success rate on the subject where more students were able to keep their overall grades at a C or above. The lab classes provided the most significant result where students show higher level of interaction as actively discussed, questioned and directed and guided each other throughout the lab session. Similarly in a study conducted by (Katira et al., 2004), the impact of pair programming on student academic performance increased when students with similar skill levels are paired together.
i. In a further study to measure the impact of constructivist techniques in improving the teaching and learning of introductory programming, (Hagan et al., 1998) introduced a weekly discussion class in addition to the normal lecture and lab work. Several activities were designed to encourage participation from students namely grids, role plays and mimics. Each of these activities derived satisfactory results as student where actively participating in discussion hence encouraging active learning overall grades of student improved significantly.

j. In addition to the various constructivist strategies described in the literature, the next major area when it comes to supplementing the learning environment to improve the teaching of programming is the use and implementation of application support resources and tools. One such implementation of this approach is seen by a experiment conducted Lockyer, Harding, 2001. Other support tools such the use of video clips – multimedia command centre(Gartner, 1997), micro worlds – Karel the Robot(Pattis, 1995), RoboPascal(Carey,1996) have also seen some success in this domain(Gartner, 1997). Additional support tools to supplement the subsequent stages of learning programming which is design, development and implementation can be illustrated with tools such as FLINT(Crews and Zieglar , n.d), various type of IDE , conventional debuggers and program animator such BRADMAN (Smith and Webb, 2000) and VINCE (Rowe and Thorburn, 2000). These tools enable students to obtain visual view the interpretation of their code or algorithm in a real-time basis (Gartner, 2003). They developed an online courseware called FOLE which was experimented and used as part of their University learning environment. This courseware consist of features such as online and offline tutor facilities, progress monitoring and student facilities such as discussion boards, forum, email and chat engines among student, tutors and lectures themselves. The results of the experiment revealed the overall learning experience and focus on the subject of the student improved significantly. In summary the key importance of application support tools and resources is that it enables guidance and feedback to be provided to learners in a way that is responsive and sensitive to learner individual needs (McLoughlin, 1998). This creates an environment that is both positive and encouraging to any educational institution.

Table 2.1.1 – Summary of Literature Review
Chapter 3 – Methodology

3.0 Introduction

This chapter is subdivided into five sections namely population, sample, instrumentation, procedures and data analysis. Each section of this chapter will discuss the specific steps applied to collect data for this study.

3.1 Population

a. Undergraduates who are currently or have prior experience learning introductory programming.

b. Lecturers with more than 1 year experience teaching introductory programming.

3.2 Sample

Based from a preliminary investigation, the faculty (FSKTM) is offering the subject “WXSE1108 – Asas Pengaturcaraan Komputer” as part of first year curriculum to teach introductory programming during the semester year of (2005/2006). A total of 5 to 6 lecturers have been assigned to teach 450 registered students for this subject.

The first sample for this study namely SMP001 will consist of purposively selected lecturers from FSKTM who have more than 1 year experience teaching introductory as part of UM curriculum. These lecturer will be categorized based the number of years they have in teaching introductory programming. This sample will also consist of lecturers who are currently teaching the subject WXSE1108 for the semester year (2005/2006).
The second set of sample namely SMP002 will include randomly selected undergraduates from FSKTM who have registered and are currently pursuing an introductory programming subject. SMP002 will consist of students from two randomly selected classes that are each taught by different lecturers and currently pursuing the subject this (2005/2006) semester year. A minimum requirement for the selection of classes would that each class has a minimum of 40 registered students.

3.3 Instrumentation

There exist two instrumentations for this study namely INT001 and INT002. The first instrumentation (INT001) will be administered on sample SMP001. It consists of an interview schedule with a total of 24 questions that is then further subdivided into a combination of open and close-ended questions. Each of these questions these questions were derived based on references stated in the literature review section. A selected number of closed-ended questions are further extended into contingency questions which are then measured using a numerical rating scale.

The next instrumentation (INT002) will be administered on sample (SMP002) and consist of a set of questionnaires. There are a total of 13 close-ended questions in this questionnaire. Similar to the previous instrument (INT001) a selected numbers of close-ended questions are further extended into contingency questions which are then measured using a numerical rating scale. The following section describes the objective and organization of the questions in both instruments.
### INTRUMENTATION DATA SHEET

<table>
<thead>
<tr>
<th>Name</th>
<th>INT001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Interview Schedule</td>
</tr>
<tr>
<td>Sample</td>
<td>SMP001</td>
</tr>
<tr>
<td>Questions</td>
<td>Consist of 13 questions that is then further subdivided into a combination of open and close-ended questions</td>
</tr>
</tbody>
</table>

#### Questions

<table>
<thead>
<tr>
<th>Question</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 1-8</td>
<td>These set of questions are aimed to identify the type, frequency and the level of effectiveness of each constructivist strategy that has been (has or not) been applied by the teaching staff for student of introductory programming in FSKTM.</td>
</tr>
<tr>
<td>Question 9 – 17</td>
<td>These set of questions are aimed to obtain information to how the current subject is structured within the context of the curriculum specified by FSKTM.</td>
</tr>
<tr>
<td>Question 18 - 19</td>
<td>These question attempts to obtain the educators view on the introduction of the alternative introductory program sequence to the current teaching environment.</td>
</tr>
<tr>
<td>Question 20</td>
<td>These set of questions are aimed to obtain the feedback from the educator on their perception of introducing tutor to assist in the teaching of introductory programming.</td>
</tr>
<tr>
<td>Question 21 – 27</td>
<td>These set of questions are aimed to obtain feedback from the educator on the need and importance of identifying the cognitive characteristic of students to enhance the teaching of introductory programming.</td>
</tr>
</tbody>
</table>

Table 3.1.1 – Instrumentation Data Sheet (Interview Schedule)

### INTRUMENTATION DATA SHEET

<table>
<thead>
<tr>
<th>Name</th>
<th>INT002</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Questionnaires</td>
</tr>
<tr>
<td>Sample</td>
<td>SMP002</td>
</tr>
<tr>
<td>Questions</td>
<td>Consist of 13 close-ended questions</td>
</tr>
</tbody>
</table>

#### Questions

<table>
<thead>
<tr>
<th>Question</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 1-2</td>
<td>These set of questions are aimed to obtain the current, targeted of the academic grade scores and their motivation in pursuing introductory programming</td>
</tr>
<tr>
<td>Question 3 – 10</td>
<td>These set of questions are aimed to identify the type, frequency and the level of effectiveness (actual or perceived) depending on if the strategy (has or not) been applied by the teaching staff for students of introductory programming in FSKTM.</td>
</tr>
<tr>
<td>Question 11 – 13</td>
<td>These set of questions are aimed to identify the type, frequency and the level of effectiveness (actual or perceived) depending on if the application support tools (has or not) been for students of introductory</td>
</tr>
</tbody>
</table>

Table 3.1.2 – Instrumentation Data Sheet (Questionnaires)
3.4 Procedures

The details of sample (SMP001) such as name, email and phone number will be obtained via the faculty website or by referring to the student registrar department. Each lecturer from this sample will be emailed with a formal request to schedule for an interview session. Based on their email response an interview programme depicting the date, time, venue and lecturer name will be prepared. The interview scheduled for SMP001 will be administered directly by the researcher. A minimum of three lecturers will be interviewed within a period of two weeks. Each interview session will take approximately 60 minutes to complete.

The student questionnaires for sample (SNP002) will also be administered directly by the researcher. The questionnaires will be administered 15 minutes before a normal scheduled lecture. Approval will first be obtained from the class lecturer before proceeding with the study. Student will be informed of the objective and overview of this study as well as of their choice to either accept or decline to participate in the study.

In order to obtain a more accurate representation of the data, the questionnaires will be administered towards end of the semester the opinions and suggestion of students can be better validated with their experiences and knowledge gained during the course of the subject.
3.5 Data Analysis

The information obtained based on the research questions via both instruments (INT001) and (INT002) will be depicted and analyzed in either a bar chart or a tabulated form.

In research question 1(i), 1(ii) and 1(iii) the data collected from the respondents will be tabulated in the following tables 3.5.1 thru 3.5.3. These data will then be further analyzed by summarizing the information that is tabulated to enable a conclusion to be formulated.

<table>
<thead>
<tr>
<th>#</th>
<th>Determine the type of learner?</th>
<th>How is it currently done?</th>
<th>Why is it important/Not Important?</th>
</tr>
</thead>
</table>

Table 3.5.1 – Response from Lecturers on Questions on Preferred Learning Styles

<table>
<thead>
<tr>
<th>#</th>
<th>How can this information be applied</th>
</tr>
</thead>
</table>

Table 3.5.2 – Response from Lecturer on Questions on Motivation Type

<table>
<thead>
<tr>
<th>#</th>
<th>Prior Experience Perform better?</th>
<th>Further Comments</th>
<th>Why is it important/Not Important?</th>
</tr>
</thead>
</table>

Table 3.5.3 – Response from Lecturer on Questions on Prior Experience

In research question 2(i),(ii),(iii) the data obtained from the respondents will be tabulated in both a graphical and table form. There basically exist three formats for bar charts as depicted in figure 3.5.1 – “Type of Constructivist Strategies / Experience”, figure 3.5.2 – “Type of Constructivist Strategies / Experience”, figure 3.5.3 – “Type of Constructivist Strategies/ Impact on Academic Performance or Motivation Level”. Each of these figures will provide a depiction of data provided by the respondents. These data will then be averaged to enable a suitable conclusion be made.
Figure 3.5.1 – Type of constructivist strategies / experience

Figure 3.5.2 – Type of constructivist strategies / experience

Figure 3.5.3 – Type of constructivist strategies/ impact on academic / motivation level
Table 3.5.3 will summarize the comments by lecturers on each strategy. These comments will provide more added values in terms of understanding the information provided.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Comments on other techniques used to improve the process of teaching prog.</th>
</tr>
</thead>
</table>

Table 3.5.3 – Response from Lecturer on other Techniques Used

The data gathered from the respondents for question 3 will be depicted in the following figures below:-

Figure 3.5.4 – Type of constructivist’s strategies/experience

Figure 3.5.5 – Type of constructivist’s strategies/academic/involvement rate

Data from each of these figures will be averaged to enable a conclusion to be devised.
Data from the final question will be represented in a two table layouts each detailing the comments on the impact of alternate program sequence and the advantages/disadvantage of the introduction of tutors.

<table>
<thead>
<tr>
<th>#</th>
<th>Positive Impact?</th>
<th>Addition Comment</th>
<th>Challenges</th>
</tr>
</thead>
</table>

Table 3.5.4 – Impact of APS Table Layout

<table>
<thead>
<tr>
<th>#</th>
<th>Positive Impact?</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
</table>

Table 3.5.5 – Advantages/Disadvantages of Tutors Table Layout

In summary all data depicted in the bar chart and each table in this section will later be further analyzed in chapter 5 – Conclusion. This section will categorize, summarize and average the data from the figures gathered in research question 1 thru 4. The formation of the support framework will be heavily influence by the calculated averages in chapter five.
Chapter 4 – Findings

4.0 Introduction

This chapter is subdivided into response rate, demographic data and findings. Each section provides the results of data analyses and findings of this study.

4.1 Response Rate

The first sample which is SMP001 consists of randomly selected lecturers who have a minimum of 1 year experience teaching introductory programming subjects. The total list of participants was selected by referencing the faculties’ website and direct correspondence staff of the student registrar department. A total of 10 participants meet the necessary requirements. A formally drafted email approved by my supervisor was sent to each participant requesting them for a personnel interview. The table below depicts the response rate for each participant:

<table>
<thead>
<tr>
<th>Participant</th>
<th>Response</th>
<th>Remarks</th>
<th>Years of Experience</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>X</td>
<td>Responded but will not be able to conduct interview as the person is out of the country</td>
<td>&gt; 7 years</td>
</tr>
<tr>
<td>R2</td>
<td>✔️</td>
<td>Responded and accepted to be interviewed</td>
<td>&gt; 7 years</td>
</tr>
<tr>
<td>R3</td>
<td>X</td>
<td>Did not respond</td>
<td>&gt; 7 years</td>
</tr>
<tr>
<td>R4</td>
<td>✔️</td>
<td>Responded and accepted to be interviewed</td>
<td>4 – 6 years</td>
</tr>
<tr>
<td>R5</td>
<td>X</td>
<td>Responded but declined to be interviewed</td>
<td>4 – 6 years</td>
</tr>
<tr>
<td>R6</td>
<td>✔️</td>
<td>Responded and accepted to be interviewed</td>
<td>4 – 6 years</td>
</tr>
<tr>
<td>R7</td>
<td>X</td>
<td>Did not respond</td>
<td>4 – 6 years</td>
</tr>
<tr>
<td>R8</td>
<td>X</td>
<td>Did not respond</td>
<td>1 – 3 years</td>
</tr>
<tr>
<td>R9</td>
<td>✔️</td>
<td>Responded and accepted to be interviewed</td>
<td>1 – 3 years</td>
</tr>
<tr>
<td>R10</td>
<td>X</td>
<td>Did not respond</td>
<td>1 – 3 years</td>
</tr>
</tbody>
</table>

Table 4.1.2 – Response rate for sample SMP001

From the total of 10 participants who were invited to be interviewed 6 responded and from this total 4 agreed to be interview. A remaining of 4 participants did not respond even after few subsequent reminder emails.
The second set of sample SMP002, consist of randomly selected undergraduates from FSKTM who are currently pursuing the subject WXSE1108 – Asas Pengaturcaraan Komputer. Each class was interrupted for about 20 minutes where the administrator could administer the questionnaires directly to the students. This resulted in all students participating in the survey.

The table below depicts a summary of the response rate for SMP002 on each questionnaire question.

<table>
<thead>
<tr>
<th>Questions</th>
<th>Total Responded</th>
<th>Total Did Not Respond</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. CODE WALKTHROUGH</td>
<td>90</td>
<td>10</td>
</tr>
<tr>
<td>2. WRITING CODE</td>
<td>94</td>
<td>6</td>
</tr>
<tr>
<td>3. WRITING CODE (PRIOR TO CLASS)</td>
<td>96</td>
<td>4</td>
</tr>
<tr>
<td>4. SCAFFOLDING</td>
<td>96</td>
<td>4</td>
</tr>
<tr>
<td>5. CODE DEBUGGING</td>
<td>95</td>
<td>5</td>
</tr>
<tr>
<td>6. LECTURE NOTE RECONSTRUCTION</td>
<td>96</td>
<td>4</td>
</tr>
<tr>
<td>7. PAIR PROGRAMMING</td>
<td>96</td>
<td>4</td>
</tr>
<tr>
<td>8. DISCUSSION CLASSES</td>
<td>96</td>
<td>4</td>
</tr>
<tr>
<td>9. SUBJECT WEBSITE</td>
<td>96</td>
<td>4</td>
</tr>
<tr>
<td>10. SUBJECT WEBSITE</td>
<td>95</td>
<td>5</td>
</tr>
<tr>
<td>11. INTERACTIVE MEDIA</td>
<td>96</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 4.1.2 – Response rate for sample SMP002

4.2 Demographic Data

The demographic data for sample SNP001 should consist of the following:

| Lecturer 1                          | more than 7 years teaching experience |
| Lecturer 2                          | 4 – 6 years teaching experience      |
| Lecturer 3 & 4                      | 1 – 3 years teaching experience      |

Table 4.1.3 – Demographic data for SMP001

As for SMP002, there is no desired demographic data that will add additional value to survey.
4.3 Findings

A total of 100 students from two different classes were surveyed and the summary finding for each research question is listed below.

Research Question 1

(i) What are the benefits of understanding student cognitive characteristics such as learning styles and motivation?

<table>
<thead>
<tr>
<th>#</th>
<th>Determine student preferred learning style</th>
<th>How is it currently done?</th>
<th>Why is it important/Not Important?</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>YES</td>
<td>This is done while observing the student attitude and work output</td>
<td>It is important as we lecturer will be better equip to response to student needs.</td>
</tr>
<tr>
<td>R2</td>
<td>YES</td>
<td>Well through the common techniques like observation, daily interaction and informal discussions. I will also change my teaching style and explanation to suit the type of learner the student is.</td>
<td>It is important as we lecturer will be better equip to response to student needs.</td>
</tr>
<tr>
<td>R3</td>
<td>YES</td>
<td>The type of learner is determined based on observation that is done during the course of a lesson.</td>
<td>It will enable the pace of the class better managed and structured</td>
</tr>
<tr>
<td>R4</td>
<td>YES</td>
<td>Thru daily observation, but this takes weeks before a conclusion can be made</td>
<td>Enable lecturer to prepare the contents and exercises in accordance to the type of learner</td>
</tr>
</tbody>
</table>

Table 4.3.1 - Summary of Responses from Teachers (a)Who Determine The Preferred Learning Style Of A Student, (b) How Do They Determine The Type Of Learner A Student Is, (c) Why is it Important/Not Important To Determine The Type of Learner A Student Is.
Table 4.3.1 summarizes the responses from lectures in regards to the need to determine a student preferred learning style when pursuing an introductory programming subject, how this process is currently accomplished and the importance of obtaining this information. All four lecturers agree that it is important to determine the type preferred learning style of a student and this information is obtained by daily interaction and observation. However one lecture has noted that that determining by observation may some times be inaccurate and time consuming. The benefits that is seen by obtaining this information will enable lectures to better cater to student needs and enable the pace and structure of the class of the better managed.
<table>
<thead>
<tr>
<th>#</th>
<th>Determine the type of motivation?</th>
<th>How is it currently done?</th>
<th>Why is it important/Not Important?</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>NO</td>
<td>N/A</td>
<td>Motivation changes over time. I do not see much benefit in identifying the student motivation.</td>
</tr>
<tr>
<td>R2</td>
<td>YES</td>
<td>Yes again through the common techniques such as observation and daily interaction</td>
<td>Yes it is important. It allows us teachers to understand the student</td>
</tr>
<tr>
<td>R3</td>
<td>YES</td>
<td>Observation and interaction in class</td>
<td>This will enable us to know what is driving the student</td>
</tr>
<tr>
<td>R4</td>
<td>YES</td>
<td>Again thru observation. Many student come up to me asking me if this is an easy subject and will it enable my pointer to increase easily</td>
<td>Able to judge and pace the class accordingly</td>
</tr>
</tbody>
</table>

Table 4.3.2 - Summary of Responses from Teachers (a) Who Determine The Type Of MOTIVATION A Student Has, (b) How Do They Determine The Type Of MOTIVATION A Student has, (c) Why is it Important/Not Important To Determine The Type of MOTIVATION A Student Has.

Table 4.3.2 summarizes the responses from lectures in regards to the need to determine the type of motivation a students has in pursuing an introductory programming subject, how this is currently done and the importance of obtaining this information.

A total of 3 out of 4 lectures (75%) agree that they do determine the type of motivation a student in learning and pursuing the subject. All 75% of these lecturers concur that the motivation of students are determined thru daily interaction and observation of student behaviors and response in class. The benefit of determining the motivation are seen in two folds were by determining the motivation level of students it will enable lectures to understand what drives students to excel in the subject and also use this information to pace the momentum of the class accordingly. Only one lecturer responded to not determine the student motivation as this characteristic changes over the time and will not be beneficial in the teaching of introductory programming.
(ii) How can this understanding be applied by an educator to improve the teaching of introductory programming?

<table>
<thead>
<tr>
<th>#</th>
<th>How can this information be applied</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>The information will be useful, but it would not be practical to put into practice. I will not be able to cater for each individual learning style or motivation in a class of more than 60 students. But if the students are initiative I will be able to use this information during their one-to-one session with me.</td>
</tr>
<tr>
<td>R2</td>
<td>If this information is made available to me, I can better plan and structure my class accordingly. However there is a concern student motivation will change during the course of the subject. Hence this info cannot be conclusive but only use as a guide. Also the information should be anonymous to avoid any kind of biasness that may or may not arise.</td>
</tr>
<tr>
<td>R3</td>
<td>In general it will give teacher a better perspective of the capabilities and preferred learning style of student. Teachers will be better equipped to deal with the daily interaction with students.</td>
</tr>
<tr>
<td>R4</td>
<td>Yes, if I have this information on each student, I will specifically be able to cater to their strength. The faculty should provide more awareness to improve the motivation of students.</td>
</tr>
</tbody>
</table>

Table 4.3.3 – How lecturers would use the above information (type of motivation and learner) to improve the process of teaching introductory programming if it were to be made available to them

Table 4.3.3 summarizes the response of lecturers in regards to how they would use this information (type of motivation and learner) to improve the teaching process. All lecturers concur that this information will be extremely beneficial to them. They will be able to use this information to better structure and plan their proceeding of their lessons. It will enable lecturer to capitalize on student strengths and equip them better to deal with students needs. One key area mentioned by one lecturer this information should be anonymous to reduce the ‘biases’ that may arise. Also this information should also be used as a guide. The one constraint indicated by one lecturer was that it would not be practical to cater for each student learning and motivation style but rather it can be useful during one on one session.
<table>
<thead>
<tr>
<th>#</th>
<th>Perform better?</th>
<th>Further Comments</th>
<th>Why is it important/Not Important?</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>NO</td>
<td>-none-</td>
<td>It is not important as this course is catered for students with zero programming experience. However during the course of the subject, some experience such as problem solving skill can come into good use. Sometime prior experience can be detrimental as student may be too advance for the class and this may change the pace of teaching. The main issue is the attitude of the students. Knowledge can be transferred effectively if the student has a good attitude.</td>
</tr>
<tr>
<td>R2</td>
<td>YES</td>
<td>It doesn’t mean that they will perform better, but it will help to understand topic faster and better. However this does not translate to performing better in the end of the day. It just means that student may have a slightly easier time digesting the contents of the topic or subject.</td>
<td>Again I do not think that it is important or not important. It is good to have but it is not vital.</td>
</tr>
<tr>
<td>R3</td>
<td>YES</td>
<td>This was previously done before where student are broken into two separate classes (with/without experience). Student are not aware of this classification and the lecturer and tutor structure the teaching accordingly in both of these classes. There was no clear distinctive difference between the results of grade score towards the end of the semester. If this was not done, the student with prior exp may have excelled better compared to others students.</td>
<td>The curriculum is catered for student with minimal or no programming language. If the lecturer conduct their class accordingly, then prior experience is not important.</td>
</tr>
<tr>
<td>R4</td>
<td>YES</td>
<td>It can beneficial but does guarantee that they will perform better. The key thing is their motivation. A student who is motivated will perform and obtain the required knowledge</td>
<td>The main thing is motivation. The curriculum caters for zero based knowledge. Exp is not necessary</td>
</tr>
</tbody>
</table>

Table 4.3.4 – Responses from lecturer (a) if student with prior experience perform better and (b) the importance of having prior experience in programming
Table 4.3.4 depicts the responses from lectures to measure if student with prior experience in programming will perform better in introductory programming subjects and importance of student to having this experience before pursuing the subject. Three out of four lecturers generally agree that student will likely to perform better if they have had prior experience in programming. However this more of an exception rather than a rule, as the content and curriculum cater for student with zero or minimal knowledge. One lecture agree that it will be easier for student with prior experience to digest contents of what is being taught but this does not translate to them performing better. Another lecturer managed to experiment with this method by separating student with and without prior programming experience into separate classes. Students were not aware of this separation and lecturers used different pace and teaching method to cater for each class. The outcome of this experiment was both group of student performed similarly in the final examination. This would not been possible if these groups of student were in the same class, as the risk of loosing out students by pacing the class either too quick or slow for the liking of the students. Overall if the lecturer conducts the class in a proper manner prior experience should not count. One lecturer pointed out that it is more of the student attitude that matters, it is sometime more detrimental to have with prior experience as they may start to dictate the pace of the class.
Research Question 2

(i) What are the constructivist strategies that have been implemented by the teaching staff for the students of introductory programming in FSKTM?

(ii) What are the perceptions from the teaching staff on the effectiveness of implementing constructivist strategies to enhance the process of teaching introductory programming?

Figure 4.3.1 - Responses of teachers who have experienced applying the above constructivist strategies in their introductory programming subjects.

Figure 4.3.2 - Responses of teachers on the effectiveness of the above constructivist strategies in improving the student UNDERSTANDING in learning the subject
<table>
<thead>
<tr>
<th>Strategy</th>
<th>Respondent</th>
<th>Comments/Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>R1</td>
<td>The strategy will enable each student to be responsible for his/her work. However there will be too much of time and human resource required to implement this strategy.</td>
</tr>
<tr>
<td></td>
<td>R2</td>
<td>This strategy is effective and is currently done on an individual basis. The one disadvantage of this strategy is the amount of time required to review each individual answer of students.</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>The technique is beneficial but students prefer referencing actual code compared to pseudo-codes. This sometimes acts as deterrence toward archiving the day end objectives.</td>
</tr>
<tr>
<td></td>
<td>R4</td>
<td>This technique will encourage student to work on their own and construct knowledge independently as there is no one in the class working on the same problem. However it will require a large question bank to effective employ this technique.</td>
</tr>
<tr>
<td>C2</td>
<td>R1</td>
<td>This strategy is useful in allowing groups to work with one another, but the same time and resource constraints doesn’t allow a lecturer practice this approach.</td>
</tr>
<tr>
<td></td>
<td>R2</td>
<td>Through observation of student answers and understanding which progressively approve during the course of the subject.</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>If this activity is managed and conducted properly, it will enable students to engage in interactive communication and exchange of ideas. It is important for student to be comfortable in their environment as this activity often creates impromptu spur of the moment issues and questions that will require participation from all group members to enable a solution to be developed</td>
</tr>
<tr>
<td></td>
<td>R4</td>
<td>Thru daily observation and daily improvement to their task completed/submitted. However this is more evident in hardworking and keen to learn students</td>
</tr>
<tr>
<td>C3</td>
<td>R1</td>
<td>This strategy is applied in all tutorial classes but student solutions are not collaborated and discuss among group members. This method (working individually) is applied in all classes and is proven to be effective in improving the students understanding.</td>
</tr>
<tr>
<td>--------</td>
<td>----</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>R2</td>
<td></td>
<td>Through observation of student answers and understanding which progressively approve during the course of the subject</td>
</tr>
<tr>
<td>R3</td>
<td></td>
<td>This activity if done accordingly is effective, but often students merely copy the answers from their friends when their do not know how to answer the question. However a teacher will be able to identify to which is original and not.</td>
</tr>
<tr>
<td>R4</td>
<td></td>
<td>This is similar to the previous where student require to complete the question independently. However the constraints to the lecturer in obtaining a large enough question bank to perform this task</td>
</tr>
<tr>
<td>C4</td>
<td>R1</td>
<td>This strategy looks promising but again the time and resources required is to extensive.</td>
</tr>
<tr>
<td></td>
<td>R2</td>
<td>Students are general find it troublesome to insert comments in a program. The rather dive to the deep end using a compiler rather then work on a sheet of paper and writing comments for a program code. Through this strategy look to be beneficial (if executed properly) as it will allows the student to think about what the code is actually doing (semantics), I believe that this activity will decrease the student interest toward the subject.</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>Student performance increase over a period of time when conducting this activity. Students are forced to make an attempt to understand the line a code in order to provide comments to it.</td>
</tr>
<tr>
<td></td>
<td>R4</td>
<td>Student can only use this technique during the first few lessons. After that this technique is not used anymore. However when it is used, it provide a better understanding to students.</td>
</tr>
<tr>
<td></td>
<td>R1</td>
<td>R2</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td><strong>C5</strong></td>
<td>This method is good to identify syntactical errors, but it does not assist much in translating the problem into its logical representation.</td>
<td>Progressing thru each tutorial and lesson, the error more easily identified and the quality of code which the student then produce show improvements. This technique is used often especially during the completion of a topic and section.</td>
</tr>
<tr>
<td><strong>C6</strong></td>
<td>This strategy is not practical as will cater only for student who are enthusiastic in learning the subject and risk less motivated students being left behind.</td>
<td>This technique is more suited for more theoretical subject which are heavily lecture based. Programming is like math, where the more you program the better you get. Hence this technique is good but not suited for programming.</td>
</tr>
<tr>
<td><strong>C7</strong></td>
<td>This can be detrimental as there are high chances that one student will do all the work while the other copies from his partner.</td>
<td>This technique look promising as students have various different ways in attempting to solve a problem. By pair programming student, they will be able to exchange ideas proactively and the weaker student can benefit from the stronger students.</td>
</tr>
<tr>
<td>R3</td>
<td>This technique enables student to discuss and exchange ideas proactively. It its administered accordingly it can be very effective. However some student do take advantage of this time and slag off instead of putting quality hours in solving and resolving the issue. There is also good student who prefer working best when there are alone.</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>R4</td>
<td>It will be beneficial as we are able to pair poor students with good ones, hence the exchange of ideas and learning would be created.</td>
<td></td>
</tr>
<tr>
<td>C8</td>
<td>This strategy is important and done during each tutorial class. It is some difficult to quantify effectiveness for the student who are really keen this process give them a chance to think out of the box and resolve problem collectively.</td>
<td></td>
</tr>
<tr>
<td>R1</td>
<td>Programming is a subject that is 80% computer based (where a student sits in front of the computer and writing code to solve a problem). A discussion class will be beneficial for student who are proactive and have the desire to learn. There are not many students who are proactive and many will take this class as an escape route to study. This technique can be beneficial if discussion about the flow of the program (design).</td>
<td></td>
</tr>
<tr>
<td>R2</td>
<td>Even when encouraged by the teacher the students do not attempt to discuss and remain passive during the entire period.</td>
<td></td>
</tr>
<tr>
<td>R3</td>
<td>Those students who are proactive this technique is beneficial. However there are a majority of students who are very passive, hence not really reaping benefits of a discussion class.</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.3.5 - Comments from teacher on each constructivist strategies
<table>
<thead>
<tr>
<th>Description</th>
<th>Respondent</th>
<th>Comments/Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Other strategies used to improve the process of teaching introductory programming</td>
<td>R1</td>
<td>No other techniques</td>
</tr>
<tr>
<td></td>
<td>R2</td>
<td>There are no specific other technique that are used. But if a student comes and see me personally then I may employ technique such as one-to-one mentoring or coaching</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>VIVA - Teacher will ask student to retrieve any exercise that they have previous completed and are require to explain their solution once again to the teacher. This technique which was applied in the previous semesters created positive changes where student are more aware of the daily task and follow up work</td>
</tr>
<tr>
<td></td>
<td>R4</td>
<td>VIVA - This session is conducted by the tutor where there are 5 lectures and 2 students during each VIVA. A little improvement is seen.</td>
</tr>
</tbody>
</table>

Table 4.3.6 - Comments on any other techniques that are used to improve the process of teaching introductory programming

Figure 4.3.1, 4.3.2 and table 4.3.5, 4.3.6 depicts the overall findings for research question 2(i) and 2(ii). When reviewed collectively each of these figures and tables reveals valuable data which contributes to the study. Figure 4.3.1 and 4.3.2 describes the experience followed by the effectiveness of the each constructivist strategy (C1 thru C8). The subsequent table (4.3.5 and 4.3.6) extends these findings by listing the comments and remarks from the lecturers as well as the other technique used to improve the teaching of introductory programming.

The data from figure 4.3.1 reveals that a total of 3 out of 4 teachers (75%) do not have experience in applying “C1 – Code Walkthrough” in their introductory programming subjects. The remaining lecturer has had “some experience” using this strategy. However regardless of the level of experience, all four lecturers (100%) ascertain that this technique will be generally effective in improving the student understanding towards learning the subject.
This is seen in figure 4.3.2 where three lectures provided feedback that the technique (is or will be) “effective” and the remaining lecturer stated that it will be “satisfactory”. The level of effectiveness is slightly compromised where from the three lecturers who stated that the strategy is effective; two expresses that it will require a large amount of time, resources and effort to apply it and the remaining lecturer describing that a large question bank is required for this strategy. The final respondent provided a slightly different view in regards to the effectiveness of this technique by stating that student may be less inclined to adopt this strategy as they may be more accustomed in dealing with actual codes compared to pseudo-codes.

The following strategy which is “C2 – Writing Code” show an improved level of exposure where 3 out of 4 lecturers (75%) have “experienced” while the remaining lecturer has had “some experience” in applying this strategy. All four lecturers believe that this strategy is effective where the first two stated it to be “very effective” and the remaining described it to be “effective”. The comments expressed by the lecturers indicated that this strategy will enable groups of students to work together and engage in interactive communication and exchange of ideas. One key element that was highlighted by one lecturer was that students are required to be in a non-threatening and comfortable environment as it will encourage participation from all group members thus enabling the construction of knowledge independently. All lectures concurred that the effectiveness of this strategy is validated by some form of daily interaction and observations.
C3 – Writing Code (Prior to class) has a more distributed level of experience where one lecturer has had “experience”, two with “some experience” and the remaining with “no experience”. Similar to C2, all four lecturers (100%) indicate that this strategy is overall effective where one lecturer reported it to be “very effective”, two stating it to be “effective” and the remaining indicating it to be “satisfactory”. Similar comments of constraints were stated by lecturers such as:- this strategy will require a large amount of resources, time and effort plus a large question bank will be required to effectively implement this strategy. Also there is a risk that students will be tempted to merely copy the answers from their peers when they are unable to answer the question. The effectiveness of this strategy is similarly validated by various daily interaction and communication.

The subsequent constructivist strategy is “C4 – Scaffolding” where 2 out of 4 lecturers (50%) have “no experience” applying this technique and the remaining two is each divided into “some experience” and “experience”. The opinion from the lecturers on the effectiveness of this strategy is also equally divided, where 2 out 4 lectures stated that this strategy is “effective” and the balance indicating that it is “not effective”. Similarly the comments provided are in contrast with one another where one section of the group expressed that this strategy looks promising and it will contribute towards the student increase in academic performance while the remaining section believes that this technique is tedious and cumbersome and will only be applicable during the first few lesson. Those who indicated that this strategy is effective in improving the student understanding of the subject are validated thru observational technique on student performance over a time period.
Code debugging “C5” reveals the highest level of experience from lecture where all four lecturers (100%) have experience applying this strategy. The overall effectiveness of this technique is also seen to be extremely positive where one lecture stated that it is “very effective”, two indicating it to be “effective” and the remaining to be “satisfactory”. Similarly, the remark from the lecturer shows that this strategy is often used and is vital in improving the student understanding toward the topics that are being taught. Also, one lecturer did indicate that the student grades scores steadily improve the more this strategy is applied and is beneficial in identifying syntactical errors.

In contrary to “C5 – Code Debugging”, “C6 – Lecture Note Reconstruction” depicts an overwhelming response where all four lecturers (100%) have “no experience” applying this strategy. This is extended where all lecturers indicate and concur that this strategy is “not effective”. The remarks validating their opinion expresses that this strategy is not suitable for introductory programming subjects in which the practical elements are so much higher compared to the theoretical elements.

“C7 – Pair Programming” reverts to a more disperse view of experience where one lecturer indicated that this strategy is “effective” and another stating have “some experience. The remaining two lecturers were in consensus agreeing that they have “no experience” applying this strategy. The comment also provide some stark contrast where one lecturer commented that this strategy can be detrimental as there are high chances that only one student will do all the work while the other will merely copy the answered for their partners. However, there was more positive feedback in regards to this strategy from the remaining three lecturers where they agreed that by carefully pairing student with one another it will enable students to exchange ideas proactively and the weaker students can benefit from the stronger students.
The final strategy which is “C8 – Discussion Classes” shows that 3 out of 4 lecturers have experienced applying it in their introductory programming subjects where the remaining one lecture has “no experience”. The level of effectiveness is also dispersed where one lecturer indicated that this strategy is “effective”, one indicating it is “satisfactory” and the remaining two stating that this strategy is “not effective”.

The general comments from the lecturers indicate that this strategy is useful for students who are proactive and have the desire to learn. However, these groups of students are extremely small and the majority of students will use this strategy as an escape route to slag off.

In addition, lecturers where also queried on the other strategies or technique used to teach introductory programming. One technique that often came up was VIVA (depicted in table 4.3.6) where a student lead by a panel of lecturers are required to retrieve any previously completed and are required to make some changes or provide explanation to it. This technique which has been previously applied reveals encouraging results where student were more aware of their work and understanding towards the topics that are being taught.
(iii) How do students of introductory programming in FSKTM feel about the use of constructivist strategies as a teaching method and its impact to their motivational level and academic performance?

![Figure 4.3.3](image-url)  
**Figure 4.3.3 - Responses of student who have experienced the different types of constructivist techniques applied in your introductory programming subjects**
Figure 4.3.4 - Student responses on the impact of each constructivist technique (has or will have) on the RESULTS of test, assignment, tutorial, quiz or examination.

<table>
<thead>
<tr>
<th></th>
<th>NC</th>
<th>NA</th>
<th>X</th>
<th>T_INC</th>
<th>T_DSC</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>56</td>
<td>14</td>
</tr>
<tr>
<td>C2</td>
<td>8</td>
<td>2</td>
<td>6</td>
<td>75</td>
<td>75</td>
</tr>
<tr>
<td>C3</td>
<td>13</td>
<td>5</td>
<td>4</td>
<td>72</td>
<td>75</td>
</tr>
<tr>
<td>C4</td>
<td>21</td>
<td>8</td>
<td>4</td>
<td>60</td>
<td>72</td>
</tr>
<tr>
<td>C5</td>
<td>9</td>
<td>5</td>
<td>4</td>
<td>81</td>
<td>60</td>
</tr>
<tr>
<td>C6</td>
<td>17</td>
<td>6</td>
<td>5</td>
<td>50</td>
<td>81</td>
</tr>
<tr>
<td>C7</td>
<td>16</td>
<td>11</td>
<td>4</td>
<td>59</td>
<td>50</td>
</tr>
<tr>
<td>C8</td>
<td>13</td>
<td>14</td>
<td>4</td>
<td>62</td>
<td>59</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th># of student who selected “No changes”</th>
<th># of student who did not answer the question</th>
<th># of student who answered “Increase”</th>
<th># of student who answered “Decrease”</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>10</td>
<td>8</td>
<td>13</td>
<td>21</td>
</tr>
<tr>
<td>NA</td>
<td>10</td>
<td>2</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>X</td>
<td>10</td>
<td>6</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>T_INC</td>
<td>56</td>
<td>75</td>
<td>72</td>
<td>60</td>
</tr>
<tr>
<td>T_DSC</td>
<td>14</td>
<td>9</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>
Figure 4.3.5 - Student responses on the impact of each constructivist technique (has or will have) on the MOTIVATION level in learning the subject.

Figure 4.3.3, 4.3.4 and 4.3.5 depicts the overall findings for research question 2(iii). When reviewed collectively each of these figures reveals valuable data which contributes to the study. Figure 4.3.3 describes the experiences of student with each constructivist strategy. This is followed with Figure 4.3.4 and Figure 4.3.5 which summarizes the impact of the each constructivist strategy (C1 thru C8) on the student academic performance and motivational level.

Data from table 4.3.3 reveals that a total of 42 students consisting of 18 “experience” and 24 having “some experience” of the “C1 – Code Walkthrough“strategy being applied in their introductory programming subject. The remaining 58 students have indicated that they have “no experience” in this strategy.

<table>
<thead>
<tr>
<th>NC</th>
<th># of student who selected “No changes”</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td># of student who did not answer the question</td>
</tr>
<tr>
<td>X</td>
<td># of students who provide an invalid answer</td>
</tr>
<tr>
<td>T_INC</td>
<td># of student who answered “Increase”</td>
</tr>
<tr>
<td>T_DSC</td>
<td># of student who answered “Decrease”</td>
</tr>
</tbody>
</table>
Figure 4.3.4 illustrates that a total of 56% of students have indicated that this strategy has improved their academic performance while only 14% of the same group of students indicated that “C1 – Code Walkthrough” has had a negative impact on their academic performance. A total of 20% of students either “Did not answer” or provided an “Invalid answer”. The remaining 10% of students stated that there was “No Change” in the academic scores. Figure 4.3.5 further illustrates the positive impact of this strategy where 74% of the students indicate that this strategy has increased the motivation level towards learning the subject and only a small fraction of the students (5%) indicate that this strategy has decreased their motivation level. The remaining group of students consists of (6%) indicating “No Changes”, 10% “Did not answer” and a final 5% provided “Invalid answers”.

“C2 – Writing Code” reveals a higher degree of exposure where 72% reported that they have had either “some experience” or “experience” in this strategy while the remaining 28% indicating “no experience”. The positive impact of this strategy is seen in both figures (4.3.4 and 4.3.5) where 75% denoted an increase in academic result and 73% on the motivational level respectively. Only a total 14% in both categories indicated a decreased in academic performance (9%) and motivational level (5%). The total invalids responses in these categories total up to 8% in both figure 4.3.4 and figure 4.3.5. The remaining reported figures are 8% indicating “no changes” in the academic performance and 10% in their motivational level.
In strategy “C3 – Writing Code (Prior to class)” shows a higher degree of inexperience from student where 59% denoted that they have “no experience” in this strategy. The remaining 41% are subdivided into 20% having “experience” and 21% “some experience”. Regardless of the level of experience the majority of students (72%) concur that “C3” has or will increase their academic performance and their motivational level. A small fraction of students (6%) and (3%) believe that this strategy will decrease their academic and motivation level in learning the subject. A total of 13% of students “No changes” in both the categories(academic and motivation).The remaining responses of students indicate 5% “did not answer”, 4% “invalid answers” for academic performances and 8% “did not answer”, 4% “invalid answers” for motivational level.

In “C4 – Scaffolding”, a large percentage of students reported “no experience” in learning introductory programming using this strategy. Only 16% reported to have “experience” and 13% with “no experience”. The impact of this strategy shows a higher degree of acceptance where 60% of student indicated that this strategy is effective in increasing their academic performance, 20% of students reported “No changes” and 6% reported a “Decrease”. A total of 12% of invalid responses were collected where from this total 8% were categorizes as “Not answered” and 4% as “Invalid”.

The following constructivist strategy which is “C5 – Code Debugging” shows a more distributed level of exposure where 34% of student responded “Yes”, 39% stated “Some Experience” and 27% indicated “No”. The impact this strategy indicates a high degree of acknowledgement where 81% of student responded that this strategy will or has increased their overall grade performance. 9% reported “No changes” and a total of 12% “did not answer” or “provided invalid answers”. 
Similarly the same percentages of students (81%) revealed that this strategy has also increased their motivational level in learning the subject. Only a small fraction of student stated that this strategy had “No changes” on their motivational level and the remaining 11% “provided invalid answers” or “did not answer the question”.

“C6 – Lecture note reconstruction” shows an overall lack of exposure among students where 61% indicated “No experience”, while only total “39% have either “experience” or “some experience” in this strategy. 50% of students indicate that this strategy will improve the academic performance and 57% their motivational level in learning the subject. A total students accumulating to 19% indicated a decrease in academic performance and 10% in their motivational level. A total of 10% of students “did not provide answer” or “provided an invalid answer” and the remaining 21% responded “no change” on their academic performance. As for the motivational level 16% responded “no change”, 13% “did not answer” and 4% were invalid.

“C7 – Pair programming” displays the lowest level of experience where 61% answered “no experience”, 27% “experience” and 11% “some experience”. The impact on the academic performance shows encouraging results where 59% indicated an increase, 16% “no change”, 11% “did not answer”, 10% “decrease “and 4% “invalid answers”. The motivational level that this strategy influences shows positive impact, with 59% indicating and “increase”, 14% each for both “no changes” and “not answered”, 4% on :invalid answer” and 4% of student stating a “decrease” in motivational level.

The final strategy which is “C8 – Discussion classes” show a total 62% of student having either “experience” or “some experience” in this strategy and the remaining 48% indicating “no experience”. 

79
62% of these students indicate that this strategy will or have increased their academic performance while only 7% stating a decrease. The remaining 31% were divided into 13% “no changes”, 14% “not answered” and 4% “invalid answers”. As for the motivational level, 60% of students indicated an “increase” and only 8% stating a “decrease”. The remaining 31% was accumulated with 14% each for “no changes” and “did not answer question” and 4% provided “invalid answers”.

**Research Question 3**

(i) What are the perceptions from students on the impact of introducing application support tools to their learning environment in order to improve the academic grade scores and the level of student involvement?

![Figure 4.3.6 - Responses of student who have experienced the different types of support tools applied in their introductory programming subjects](image-url)

Figure 4.3.6 - Responses of student who have experienced the different types of support tools applied in their introductory programming subjects
Figure 4.3.7 - Student responses on the impact of each support tool (has or will have) on the RESULTS of test, assignment, tutorial, quiz or examination.
Figure 4.3.8 - Student responses on the impact of each support tool (has or will have) on the INVOLVEMENT level in learning the subject.

Figures 4.3.6, 4.3.7 and 4.3.8 depicts the student responses on the impact of application support tools have on in either increasing or decreasing the involvement level in learning the subject as well as its impact to their academic performance. Students responses were analyzed based on three categories of support tools namely “S1 - subject website (static contents)”, “S2 - subject website (interactive contents)” and “S3 - interactive media”. These responses were tallied to measure if the introduction or the further development of application support tools will enables better improvements to the overall learning process.
From a total of 100 students which were surveyed, the highest level of experience level is seen in “S1 – subject website(static content) where 60% reported to have ‘experience’ followed by 21% indicating ‘some experience’ and 19% of student responded to have no experience in using the above application support tool. The impact on student academic performance and involvement rate show extremely positive results with 75% student responded an increase in their academic scores and similarly 72% on their participation rate. Only a fraction of students accumulating to 3% reported a decrease and 4% provided invalid answers in both categories. 5% of students reported “no changes” in their academic performance followed by 6% in their involvement rate. The remaining number of students (13% and 15% for each category) did not provide any answer to the survey questions.

The following application support tool which is ‘S2 – subject website (interactive contents) shows the lowest level of exposure among students where 53% responded to have “no experience” in using the above support tool followed with a remaining of 28% with “experience” and 19% with “some experience”. However regardless of the level of exposure 63% of students indicated and increase in the academic performance followed by 64% indicating a similar increase in their involvement rate. Only 2% stated a decrease in their academic performance and 3% in the involvement rate. A total of 9% of students from both categories provided and “invalid answered”. This is followed by 15% and 17% of students stating “no changes” in academic performance and involvement rate.

The final application support tool that was analyzed is “S3 – Interactive media” where 51% responded to have “no experience”, 29% with “experience” and 20% with “some experience”.
From this total 61% reported an increase, 11% decrease and 15% “no changes” in the academic performance. This is followed by 9% who did “not answer” the questions and 4% invalid answers. The next category reveals a total of 57% of student indicating an “increase”, 1% decrease and 16% responding “no changes” in their involvement level or rate. Only 4% responded with “invalid answers” and the remaining 22% “did not answer the question”.
Research Question 4

(i) What are the opinions from the lecturers in regards to the introduction of tutors and alternative program sequence to improve the process of learning teaching introductory programming?

**Alternative Program Sequence**

<table>
<thead>
<tr>
<th>#</th>
<th>Positive Impact?</th>
<th>Additional Comment</th>
<th>Challenges</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>YES</td>
<td>No comments</td>
<td><strong>Shortage of lecturers and insufficient time:</strong> The may be a large possibility that the required skill may still not be developed after this program</td>
</tr>
<tr>
<td>R2</td>
<td>YES</td>
<td>(this is currently being done, but a student is required to complete the whole subject and that may create an adverse effect on their interest and motivational skills)</td>
<td><strong>Other:</strong> The may be a large possibility that the required skill may still not be developed after this program.</td>
</tr>
<tr>
<td>R3</td>
<td>YES</td>
<td>This was done during previous semesters, the results were positive but somehow this program was not continued into the next semester</td>
<td><strong>Shortage of lecturers:</strong> If the faculty makes this mandatory the students will definitely benefit, especially if the number of student are small. However the faculty should provide incentives to lecturers to enable the program to take off</td>
</tr>
<tr>
<td>R4</td>
<td>YES</td>
<td>No comments</td>
<td><strong>Shortage of lecturers and insufficient time:</strong> There are any time slots. It would be beneficial if the faculty enforces it. A flavor of this is already taking place in UM.</td>
</tr>
</tbody>
</table>

Table 4.3.7 - Lecturers responses on the impact of having an “Alternate Program Sequence” as part of the teaching environment.
The above table analyzes the responses from the lecturers on the impact of having an “Alternate Program Sequence” as part of the teaching environment. All lectures agree that the above method will create a positive impact in improving the process of teaching introductory programming. However, there were some interesting views among the lecturers when asked to provide additional comments to validate their answer. One lecturer commented that this method has been applied and the results were seen, but somehow the faculty decided to discontinue with this program with no apparent reason.

Another lecturer stated that there may be some negative impact of this method as student motivational level is severely affected as they are required to complete the entire subject again. In summation, the overall challenges that were presented by the lecturers were shortage of lecturers and insufficient time to fully enable this program from becoming a success. Also, one lecturer noted that some incentives are required for lecturers as this will encourage them to participate in this program more proactively, as it requires involvement of time outside their normal working hours. The encouragement and support from the faculty is seen to be vital in ensuring the overall success of this program.
# Introduction of Tutors

<table>
<thead>
<tr>
<th>#</th>
<th>Positive Impact?</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>YES</td>
<td>It enables more personalization between student and tutor. More 1-1 communication can be established this way.</td>
<td>There is rarely any disadvantage as tutors must firstly be well trained before assuming the role.</td>
</tr>
<tr>
<td>R2</td>
<td>YES</td>
<td>Tutor will be able to personalize and have closer interaction with students. Also student see tutors as more senior student and not teachers. This perception enable student to be more comfortable when interacting with the tutor. Tutor also provides invaluable feedback on the performance and attitudes of students.</td>
<td>There may be some biases that exist as some tutor and student are friends. Tutor must be trained and educated the professional obligation to students.</td>
</tr>
<tr>
<td>R3</td>
<td>YES</td>
<td>Tutor reduces the heavy amount of workload that is burden by the lecturers.</td>
<td>There is a small sections of tutors who goes MIA at times and this can be very disruption to the class</td>
</tr>
<tr>
<td>R4</td>
<td>YES</td>
<td>Help lecturers cater to the large number of students, reduce workload and enable the lecturer to work better and identify strong and weak students</td>
<td>Some biases between tutor and students (who are friend can take place)</td>
</tr>
</tbody>
</table>

Table 4.3.8 - Lecturers responses on the impact of having an “Introduction of tutors” as part of the teaching environment.

Similar to table 4.3.7, all four lecturers concur that the above method will create a positive impact on the teaching of introductory programming. Their comments were validated by requesting the lecturers to provide the advantages or disadvantages of applying the above method as part of the teaching environment.
In general all lecturers agree that tutors will enable more one to one communication between them and students and also this introduction will reduce the tremendous amount of workload put on lecturer. Another benefit in relation to the reduced amount of workload is that it enables lecturers to structure their work better and identify strong and weak students by observing the interactions between tutor and student. One lecturer pointed out that students are more inclined to confide and communicate with tutor as they see tutor as either that peers or senior students. Thru daily communication and interaction between tutors and students it will enable invaluable feedback to be obtained in regards to the student performance and development while undergoing the subject. One clear disadvantage that may come from the introduction of tutors is the student “biasness” that may evolve by tutors who were previously or have become ‘friends’ with students. Tutor in this situation may have a tendency to pay closer attention to their ‘friends’ instead of the overall groups of student. However one lecturer pointed out that tutor who are trained accordingly, will not have this conflict as there are able to differentiate the professional and personal obligations.
Chapter 5 – Conclusion

5.0 Summary

The objective of this study is to build a support framework to aid educators in constructing a learning environment that will allow them to better deliver the required knowledge by providing a suite of resources that bundles traditional pedagogical approaches such as lectures and tutorials with the innovations of new teaching approaches together and with the support of Internet based technology. To accomplish this, it was important to list and understand the factors that reinforce the theory that teaching and learning programming is still a challenging task for both educators and students. This was followed by identifying the key influences that contributes to these factors. Based on these key influences extensive research was gathered to analyze and summarize the various initiatives by educators to which positive and encouraging results has been seen in improving the teaching of introductory programming. This was followed by an administration of a series of interview and questionnaires to the lecturers and student of introductory programming in FSKTM. Based on their feedback a support framework will be built which cater for the specific learning environment in FSKTM.
5.1 Conclusion

As described in the previous section, this subsequent area will conclude the findings reported in this study to which a support framework will be developed. The four key problem statement described in this study is reiterate below:-

i. High failure rates in first year programming courses.

ii. Low student retention numbers in programming streams.

iii. For many students who successfully proceed to higher level programming courses, a lack of understanding of basic programming concepts.

iv. It is also common for students to approach their final year project work determined to avoid programming at all costs, presumably because they either cannot program or believe that they can not.

It is also equally important to state the key influences which contribute to the problems described above as it will enable a better understanding of the overall conclusion be made:-

i. The lack of understanding and attention by educators to student cognitive styles.

ii. The reluctance of educators to adopt and adapt to the different types of teaching models being introduced due to the radical changes it will create in the way the subject is taught.

iii. The multiple disciplines such as mathematics, language learning, arts and psychology that encompasses the subject.

It is with this essence that following section is aimed to develop a support framework that will address each of these problems and their contributing factors.
The first area of research in this study was to gather the opinions of lecturers on the benefits of understanding a student cognitive characteristic such as preferred learning style and motivation and also to gather information on how this understanding can be applied to improve the process of teaching introductory programming. Based on the findings reported all lecturers agree that they do determine the type preferred learning style of a student and this normally done during daily observation and interaction. They also concurred that this process is important and critical as it will enable them to better pace and structure their teachings. In addition each lecturer believes that their teaching will be further improved if a formal assessment method to determine the type of learner a student is available. The currently assessment model of determining the type of learner relies a lot on the experience of a lecturer and students may be at a disadvantage if their teacher has limited knowledge in this areas. A formal method will be a very valuable asset to the learning environment.

In correspondence to determining a student’s preferred learning style, the next area of research explored the implementation and benefits of identifying the type of motivation a student has in learning the subject. All but one lecturer agree that knowing what motivates a student to learn the subject will provide invaluable data thus enabling a more improved student centric learning environment. Similar to determining the type of learner, a formal assessment method in identifying the motivation of a student would enable a more structured and accurate interpretation of the student motivational characteristics to be identified. In summary all lecturers believe that is vital to determine the preferred learning style and motivation a student has and the benefits to the learning environment will be extremely positive.
The next area of research is based on the findings reported in research question two. As illustrated in chapter four all strategies indicate various positive impacts on all categories which are being measured namely; the impact each constructivist strategy has in improving the teaching of introductory programming, student academic performance and motivation level. From a students perspective an average of 66.12% (529/8) indicated that the strategies will or has increased their academic scores. This corresponds similarly with their motivational level where an average of 67.25% (538/8) of student indicated that their motivation has or will increase by applying each constructivist strategy. On contrary, only an average of 8% and 5.75% of students indicated that these strategies has or will decrease the academic scores and motivation level respectively. This high degree of acceptance from student provides encouraging results towards the formation of framework. However one key aspect in this formation is the opinion and perception of each lecturer to formulate and validate the adoption of each strategy into the framework. Based on the findings reported by the lecturers, an average of 40.6% reported that the strategies were “effective” and 15.6% each indicated that the strategy was either “very effective” and “satisfactory”. This totals to an overall average of 71.8% stating that these strategies were above the “satisfactory” level which leaves only 28.2% reporting that these strategies were “not effective”. This 28.2% was largely due to the response on to strategy “C6 – Lecture Note Reconstruction” where all four lecturers indicated that the strategy was “not effective”. This is also seen from figure 4.3.4 and 4.3.5 showing the student indicated the lowest number of response for increase and the highest response for decrease in grade scores and motivation level. In summary the response from the students as well as the lecturers provides positive reinforcement to include C1,C2,C3,C4,C5,C7,C8 into the framework and leaving C6 out. As of such these categories will the adopted into the teaching framework to improve the process of teaching introductory programming in FSKTM.
The subsequent area of research was aimed in obtaining the students perception on the impact of introduction of application support tools to their academic performance and involvement level in learning the subject. Based on the findings depicted in figure 4.3.6 thru 4.3.8 an average of 66.3% of student reported an increase in academic scores and 64% on the involvement level. Only a fraction of students accumulating to an average of 16% and 14% for each category reported a decrease. This shows a positive and high degree of acceptance among students. As a result, all application support tools (S1 thru S3) strategies will be adopted into the framework.

The final area of research in this study was to obtain the opinions from the lecturer in regards to the introduction of tutor and alternative program sequence to improve the process of teaching introductory programming. Based on the response by the lecturers, all four agree that both the strategies mentioned above will create a positive impact in the overall learning environment. A sub section of the lecturers have even implemented the approaches and have reported to be satisfied with the results. Some of the constraints mentioned such as “shortage of lecturers and insufficient time” for the alternative program sequence strategy do not disqualify the potential benefits this strategy can bring but merely reinstate the changes that needs to be done with certain administrative policies in FSKTM. The following strategy by introducing tutor as part of the teaching environment provides various positive reinforcement to the lecturer to enable them to concentrate on the key task of teaching. Responsible tutors often become an extension to the teaching staff by providing invaluable insight to help lecturer better understand the effectiveness of their teaching. With such encouraging response from the lecturer both of these strategies will be adopted into the framework.
The ultimate objective on transferring all the required knowledge to a student during a lesson is slightly more possible by enforcing the strategies above.

Table 5.1.1 depicts all of the concluded findings and strategies that will be adopted into the framework. They are categorized into four quadrants to ensure that a simplified and efficient approach in practiced during the execution of these strategies within the framework. These four quadrants are listed below:-

(i) **Cognitive Characteristic of Students Quadrant**

(ii) **Constructivist Strategies Quadrant**

(iii) **Learning Support Environment Quadrant**

(iv) **Application Support Environment Quadrant**

<table>
<thead>
<tr>
<th>QUADRANT</th>
<th>STRATEGIES</th>
</tr>
</thead>
</table>
| **STUDENT COGNITIVE CHARACTERISTIC**          | • WHAT TYPE LEARNER  
Active, Reflective, Sensing, Intuitive, Visual, Verbal, Sequential, Global, Inductive and Deductive |
|                                               | • POOR LEARNING TENDENCIES (PLT)  
Superficial Attention, Impulsive Attention and Staying Stuck |
|                                               | • WHAT TYPE OF MOTIVATION  
Extinctive, Interactive, Social, Achievement, Null |
| **CONSTRUCTIVIST STRATEGIES**                 | • CODE WALKTHOUGH                                                            |
|                                               | • WRITING CODE                                                              |
|                                               | • WRITING CODE(PRIOR CLASS)                                                 |
|                                               | • SCAFFOLDING                                                                |
|                                               | • CODE DEBUGING                                                              |
|                                               | • PAIR PROGRAMMING                                                          |
|                                               | • DISCUSSION CLASSES                                                        |
| **LEARNING SUPPORT ENVIRONMENT**             | • ALTERNATE PROGRAM SEQUENCE                                                |
|                                               | • INTRODUCTION OF TUTORS                                                    |
| **APPLICATION SUPPORT ENVIRONMENT**          | • SUBJECT WEBSITE(STATIC)                                                  |
|                                               | • SUBJECT WEBSITE(DYNAMIC)                                                  |
|                                               | • INTERACTIVE MEDIA                                                         |

Table 5.1.1 – Adopted Strategies
Cognitive Characteristic of Students Quadrant

This quadrant places emphasis on the need to determine the cognitive characteristics of students who are pursuing the subject of introductory programming. There is no set interval to when this analysis should take place as the fundamental idea behind the framework is to provide a series of recommended guidelines or approaches that can be used by a lecturer to improve the process of teaching introductory programming. One may decide to evaluate the cognitive characteristic of student before the semester commence or periodically during the course of the subject. The set of guidelines and procedures recommended in this framework based on this study is described below.

- Determine the preferred learning style of student by using the Felder-Silverman Learning Style Model.

- Ensure that educators (lecturer and tutors) are equip with the knowledge of the Poor Learning Tendencies (PTL) that can fall prey too and the necessary follow up actions are executed.

<table>
<thead>
<tr>
<th>POOR LEARNING TENDENCIES</th>
<th>BRIEF DESCRIPTION</th>
<th>RECOMMEND FOLLOW UP ACTION</th>
</tr>
</thead>
</table>
| SUPERFICIAL ATTENTION    | This involves skimming over a communication, with no attempt to actively process the information to generate personal meaning. | • Reward for understanding and not completing  
• Outline a method of attack  
• Smaller coding questions  
• Not always coding |
| IMPLUSIVE ATTENTION      | Some parts of communication are attended to but others are overlooked. For example, the learner may focus on an interesting example and ignore the major point. | • Emphasize the key point  
• Provide adequate resource for introduction of unfamiliar material |
STAYING STUCK
Lack of any strategy to cope with getting stuck except to call for help. No attempt to return to the instruction reflects on the strategy selected, analyze what has been done so far or consider alternative.

- Tactics on how to start with graded helps
- Provide useful references and resources
- Provide guide to writing and testing code in manageable chunks

<table>
<thead>
<tr>
<th>Table 5.1.2 – Poor Learning Tendencies (PTL) follow up action</th>
</tr>
</thead>
</table>

The main highlighting point here is the lecturer must be well informed and trained to identify the type of PTL as each situation is different from one another and at times the recommended follow up action may not be appropriate.

- Determine the type of motivation the student has in learning the subject.

The next section in this framework is the need for lecturers to identify the type of motivation a student has in learning the subject. This identification process can take place at any time during the course of the subject and as frequently as the lecturer sees fit. The objective of this approach is to identify the type or traits of motivation as closely as possible. Lecturers can adopt their own style to determine the type of motivation or adopt approaches by previous researches described in the literature review. Based on the information gathered in the literature review there exist five key types of motivation traits which a student normally has and to which can be identified by a series of question in a questionnaire. The first section of the questionnaire probes for general information such as student status, background and anticipation in learning the subject. The second section consists of question that will help investigate the type of motivation a student has.
### SECTION ONE

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>SAMPLE QUESTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Gender, Age, Enrolment Status(PT/FT), Year of Study, Intended Major</td>
</tr>
<tr>
<td>Background information</td>
<td>What recent programming course have you taken? Do you know programming?</td>
</tr>
<tr>
<td>Anticipation</td>
<td>How difficult do you expect this subject to be? What is your expectation of the work load of this subject?</td>
</tr>
</tbody>
</table>

### SECTION TWO

<table>
<thead>
<tr>
<th>TYPE OF MOTIVATION</th>
<th>SAMPLE QUESTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extrinsic</td>
<td>What plans do you have when once you have completed your undergraduate studies (join a software house as a programmer, get a not IT related job, start a business)? Be as specific as you can?</td>
</tr>
<tr>
<td>Extrinsic</td>
<td>Being good in programming will provide me an invaluable skill set that will ensure a better job, salary and security?</td>
</tr>
<tr>
<td>Social</td>
<td>Do your parents or family want you to do well in University? How much support is given to you? Do they help you with your work? Do they ask you about your lesson and classes?</td>
</tr>
<tr>
<td>Social</td>
<td>Do your parents do anything special to encourage (or discourage) you in programming as compared to other subjects?</td>
</tr>
<tr>
<td>Social</td>
<td>Do you care about what your classmates and teacher thinks of your skill in programming? Is it important to you to look like a good math student or poor math student to your friends and teachers?</td>
</tr>
<tr>
<td>Social</td>
<td>What do your friends think about programming? Do they like it? Do they see it as useful? Do they work hard in this subject?</td>
</tr>
<tr>
<td>Intrinsic</td>
<td>How keen are you in learning subject?</td>
</tr>
<tr>
<td>Intrinsic</td>
<td>Do you think it take special talent to do well in programming? Do you have such talent? Can people do okay in programming without special talent?</td>
</tr>
<tr>
<td>Intrinsic</td>
<td>When someone makes in mistake in programming, does it mean that person is weak in programming? (probe for explanation – particularly if student feels making mistakes is part of the learning process)</td>
</tr>
<tr>
<td>Achievement</td>
<td>How often do you work hard in programming just to learn the material?</td>
</tr>
<tr>
<td>Achievement</td>
<td>How important is it for you to get a good grade in this subject? Why (probe for explanation)?</td>
</tr>
<tr>
<td>Achievement</td>
<td>How much does your teacher’s grading system affect what you try to learn? Do you every try to learn things that you know you won’t be grade on?</td>
</tr>
<tr>
<td>Achievement</td>
<td>Do you intent to enroll in second year computer science?</td>
</tr>
</tbody>
</table>

Table 5.1.3 – Motivation type questionnaires
CONSTRUCTIVIST STRATEGIES QUADRANT

This quadrant consists of the delivery and teaching strategies that can be employed by lecturers to ensure that the required knowledge is transferred from student to lecturers in a more seamless and effective manner. Based on the findings reported in this study, this quadrant consists of strategies that have been previously explained and adopted into the framework namely:

- Code Walkthrough
- Writing Code
- Code Debugging
- Pair Programming
- Scaffolding

Each or a combination of constructivist strategy employed should be selected based on criteria’s such as suitability of applying the strategy based on subject contents, students receptiveness to the ideas and activities involved and the time and resources available. Lecturers should always keep in mind that students need to be active in the learning process by interacting with the concepts, testing out theories and being allowed the space to make mistakes. It is imperative that each lecturer understand how each constructivist should be used and when it should be applied. This understanding can be obtained from various academic literature and resources such as journals, white papers or any other technical publications. In addition the system prototype which is described in chapter six provides some functionality that will assist lecturer to apply each constructivist strategy more effectively.
LEARNING SUPPORT ENVIRONMENT QUANDRANT

This quadrant consist of support strategies needed to better facilitate the learning environment in order to provide a more comprehensive and effective learning experience. Based on the finding reported in this study, two types of learning support resources namely Alternate Program Sequence (APS) and Introduction of Tutors have been adopted into the framework. The implementation of the APS program will involve a two-course sequence, where student who receive a grade C or less will be required to revisit topics with which the majority of student has had problems. Students taking this alternative introductory programming sequence end up with one additional course that can be used in their program as a professional elective.

Another variation of APS which is based on the archival research in the literature review is to group students into categories namely Rocket Scientist, Average, Strugglers and Serious Struggler. Each category of students will ideally be placed in different classrooms and given separate routes to achieve their required grade. The categorization of students can be accomplished by administrating various amplitude test and informal interviews designed by the lecturers. The following table depicts the recommended guideline for this process.
### Table 5.1.4 – APS guidelines

The next form of support resources is the Introduction of Tutor to assist lecturers in their teachings. Tutor must be trained to enable them to provide the required benefits.

The role of tutor in FSKTM should evolve into the following:-

(i) Provide guidance to student during tutorial and lab assignment.

(ii) Provide asynchronous communication via emails of forums using the prototype described in chapter six.

(iii) Provide online communication tools such chat room and messenger using the prototype described in chapter six.

<table>
<thead>
<tr>
<th>ROCKET SCIENTIST</th>
<th>AVERAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Pace and delivery of subject materials should focus on more advance topics</td>
<td>• Each tutorial and lab classes should supported with a minimum of two tutor or more.</td>
</tr>
<tr>
<td>• More advance and challenging tutorials question.</td>
<td>• Pace and delivery of subject materials should focus on a mixture of intermediate and advance topics to enable student ability to be better measure and a more detailed analysis of student capability can be made.</td>
</tr>
<tr>
<td>• Minimum supervision required</td>
<td>• Average amount of supervision required</td>
</tr>
<tr>
<td>• Interaction between student and their lecturers can be heavily depended on asynchronous or offline communications</td>
<td>• Intermediate tutorials question</td>
</tr>
<tr>
<td></td>
<td>• Interaction between students and lecturer should have a good mix of face to face communication and asynchronous or offline communication.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>STRUGLERS &amp; SERIOUS STRUGLERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Maximum supervision required</td>
</tr>
<tr>
<td>• Each tutorial and lab classes should be supported with more that five tutor (one tutor for a group of five – eight students)</td>
</tr>
<tr>
<td>• Pace and delivery of subject materials should focus on a mixture of beginners to intermediate topics. Some topics may be revisit for numerous times to ensure that the fundamental of the subject are ingrained onto the students.</td>
</tr>
<tr>
<td>• Heavy face to face communication together with asynchronous and offline communication to ensure that the progress of closely monitored.</td>
</tr>
</tbody>
</table>
(iv) Able to identify the cognitive characteristics such as preferred learning style, poor learning tendencies and type of motivation as described in the Cognitive Characteristic quadrant.
APPLICATION SUPPORT ENVIRONMENT QUANDRANT

This quadrant consists of application support tools that can be used to enhance the teaching of introductory programming which inevitably improves the learning environment. These application tools have become more of a basic requirement rather surplus for any subject. Static information such as presentation slides and course performa can be easily obtained by student before even starting the semester. However this quadrant emphasizes more on the need to have dynamic and interactive content which can be used both by the lecturer and student. Discussion forums, on line chats, student progress chart are just some of the features this quadrant should provide. Both faculty and lecturer should continuously provide input for the new development of application tools to enable more improvement to the learning and teaching environment. The actual implementation of this quadrant of FSKTM in described in chapter six. This chapter will explain the type of application tool available for student of introductory program in FSKTM. Based on the finding reported in this study application support tools such as subject website (static and dynamic) as well as interactive media are tools that student can easily related too and are most beneficial to their learning process.
In summary the diagram below summarizes the teaching support framework resulting from this research. It is believed that the four key problems and their contributing factors described in this study will be significantly reduced with the implementation of this framework. Though not written in stone each component of this framework is seen to either eliminate or reduced the problems and contributing factors described in this study.

Figure 5.1.1 – Teaching Support Framework Diagram
5.2 Discussion

This study has provided valuable insight to the problems and their contributing factors in teaching introductory programming. It has also revealed various initiatives by educators around the world in improving the teaching or introductory programming as well as the acceptance of educators and students of FSKTM of the various methods suggested. One key aspect that was observed from the key findings of this study is the fact and readiness of educators and students to adapt and change to a more practical and efficient learning model. However this change will require immense support and resources by the faculty. Overall there is a belief that a new and improved teaching system is out here and within the framework described in this study.

It is only a matter of readiness and encouragement by the all parties namely the academic faculty, lecturer and students that this realization can take place.

5.3 Recommendation

The most obvious recommendation for this study is the practical usage of the framework within FSKTM. This will allow the elements within the framework to be tested and verified for its effectiveness. One important factor when applying the framework is to apply by stages and not all in one go. This will allow a comprehensive analysis of the benefits and well as the pitfalls to be realized by the implementation of the framework. It is also important that a committee compromising of administration members of the faculty, lecturers, tutors and student be formed to administer the implementation and usage of the framework. The feedback obtained as a result of the implementation is to be documentation and reviewed accordingly to enable changes and improvement to the framework. Once all elements of the framework has been tested and verified accordingly, only then should it become a de-facto standard to teach introductory programming in FSKTM.
Chapter 6 – System Prototype

6.1 Introduction

The system prototype is an implementation of the support framework that described in chapter 5 – Conclusion. It consists of eight major modules which are later subdivided into smaller sections. A depiction of all modules is described figure (Figure 6.1.1 – System prototype below:)

![Diagram of the system prototype](image-url)
6.2 System Prototype Modules

The following section provides an overview each sub module of the system prototype.

6.2.1 Student Profile Monitor

The student profile monitor consist of data that will assist lecturers in obtaining key information about the student cognitive characteristics, personal details and current academic performance in their subject.

i. Student particulars

ii. Preferred learning style of student

iii. Type of motivation of student

iv. Academic scores and status for each tutorial, assignment and examination scores

v. Student attendance

6.2.2 Student Online Assessment

The online assessment in an area where students are able to assess their understanding and skills with a variety choices online questions pertaining to the subject to which their result can be immediately posted to them via email or directly on the web page.

6.2.2.1 Student Cognitive Assessment

This web based interface consist of questions that are aimed to gather information on the student cognitive characteristics namely; preferred learning style, type of motivation and poor learning tendencies.

6.2.2.2 Subject Assessment

This section consists of on-line test, quizzes and questionnaires that are designed by lecturers to which feedback and marks can be immediately obtained and archived for historical analysis and evaluation.
6.2.3 Online Asset Library

The Online Asset Library is a collection of tools and information that are relevant to the subject and can be used by both lecturer and students. The following list below describes the functionalities of module within this module:-

6.2.3.1 Online Subject Repository

This section consists of static information such as presentation slides, subject notes, journals and white papers that are relevant to the subject material. Both lecturers and student are able to access this knowledge base.

6.2.3.2 Online Activity & Questions Vault

This section consists of an archive of documented activities and questions that corresponds to the contents and materials for an introductory programming subject. These activities and questions are aimed to assist and guide lecturers to better deliver the required knowledge based on a constructivist model. The documented archived are compiled and stored using a web based interface where lecturers are able to upload, index and categorize each activity and questions which can be selected and used for various constructivist activities, lab session, tutorial and even assignments.

6.2.3.3 Interactive Media

This section consist of all dynamic information and tools such as web cast, interactive multimedia, IDE tools, sample programs, animated examples that are used to facilitate the understanding of a specific topic.
6.2.3.4 Exam Builder

This is a wizard that enables online questions for test, quizzes and exam to be automatically created dynamically by the lecturer.

6.2.3.6 e-Announcement

Lecturer can use this section to broadcast any relevant announcements.

6.2.3.7 Meeting Scheduler

Student can view the availability of the lecturer and tutor to schedule their appointments both face to face or online chat.

6.2.3.8 e-Submission

This upload tool enables student to automatically submit their assignment, coursework and project papers.

6.2.3.9 Clickers

This tool is placed on student table where it will be able to transmit student answers for question displayed by the lecturers. This creates anonymity for student and lecturer will be better informed of the overall understanding and response of students.

6.2.4 Communication X-Change

All form of communication that are conducted online will done in this area.
6.2.4.1 Subject Online-Forum and Discussion Board

This section will consist of topics that can be posted on a forum and discussed in an asynchronously between lecturer and student. The three main categories of online forums are described below:-

- **Student to Student**
- **Student to Lecturer**
- **Student to Tutor**

6.2.4.2 Subject Online-Chat

This section will consist of chat rooms where students can communicate interactively between themselves, lecturers and tutors. This chartrooms will be moderated by administrator to ensure that no form of abuse is prevalent. The three main type of online chat room are:-

- **Student to Student**

  Student of the same class and log in and discuss subject materials.

- **Student to Lecturer**

  Appointment based communication, where the student will initially make use of the online meeting scheduler to make appointment with lecturer to conduct this communications. Student can decide to either work in anonymous or non-anonymous mode.

- **Student to Tutor**

  Appointment based communication, where the student will initially make use of the online meeting scheduler to make appointment with tutors to conduct this communications. Student can decide to either work in anonymous or non-anonymous mode.
6.2.5 History

Contains all archived sets of data such student’s performance, grades, attendance, cognitive characteristic to which can access and analyzed for statistical purpose.

6.3 Functional Prototype

A full functional prototype was developed for the Student Profile Monitor module. The section below provides explanation and details of this module.

<table>
<thead>
<tr>
<th>Project Worksheet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module Name</td>
</tr>
<tr>
<td>Sub Modules</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Development Language</td>
</tr>
<tr>
<td>Database Engine</td>
</tr>
<tr>
<td>Web Server</td>
</tr>
</tbody>
</table>

Table 6.3.1 – System prototype system specifications
6.3.1 Main Menu

The main menu is a fully functional interface with various links to the entire system. A brief explanation of each module is provided here and the simple, organize and attractive interface provides effective and concise navigation.
6.3.2 Main Menu – (valid login)

Once a user has successfully entered a valid user id and login the following information depicted below will be displayed on the left pane.
6.3.3 Main Menu – (valid login)

Once a user has successfully entered a valid user id and login the following information depicted below will be displayed on the left pane.
6.3.4 Main Menu – (invalid login)

Once a user has enter the following an invalid user id and login the following information depicted below will be displayed.
6.3.5 Student Profile Monitor (Summary)

The initial screen for the student profile monitor displays a list of all active intakes. A lecturer will select the intake to which he/her would like to view and click the “Go” button.
6.3.6 Student Profile Monitor (Summary)

Once a valid intake is selected the key information will be displayed:

(i) Student summary listing for that particular intake

(ii) Bar chart summarising the preferred learning styles of students of the selected intake.

(iii) Bar chart summarising the motivations type of student of the selected intake.

Student details such as student id, name, email and hpno can be easily accessible by selecting the intake from the dropdown and clicking the “Go” button. Students records are paged to a maximum of 10 records to enable easy reading. Further details of student can be accessible by clicking the “view” link at the beginning of each record.

Graphical representation of student preferred learning style is depicted here. This will enable lecturer to gain a quick insight of the type preferred learning style in that particular intake.
Graphical representation of student’s response questions to determine their motivation.
6.3.8 Student Profile Monitor (Detailed.)

In this section all details relevant to a particular student within the selected intake is depicted. There exist six key sub sections namely Student Details, Poor Learning Tendencies, Preferred Learning Style, Student Attendance, Student Grade Book and Student Motivation. There also exist interactive help to help assist users of this webpage.

Student biodata and picture is listed here for a more comprehensive viewing of student details.

Summary of student poor learning tendencies (PTL) SA – Superficial Attention, IA – Impulsive Attention, SS – Staying

Summary of student preferred learning styles

Student attendance, click ‘details’ for breakdown of each day/time L - Lecture, T - Tutorial

Lecturer assessment (marks, comments) of student’s assignment and tutorials is detailed here. Date and time the documents was uploaded by students is also displayed. The actual physical document can be viewed by clicking the ‘view’ button
Details of each question to determine the student motivation is listed here.
### 6.3.9 Student Profile Monitor (Detailed Contd.)

This page is displayed when the ‘details’ link in the Student Attendance is clicked. It consist of each day to which a student has attended their tutorial or lectures.

<table>
<thead>
<tr>
<th>Date</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>20-Mar-06(Mon)</td>
<td>Tutorial</td>
</tr>
<tr>
<td>21-Mar-06(Tue)</td>
<td>Tutorial</td>
</tr>
<tr>
<td>22-Mar-06(Wed)</td>
<td>Tutorial</td>
</tr>
<tr>
<td>23-Mar-06(Thu)</td>
<td>Tutorial</td>
</tr>
<tr>
<td>24-Mar-06(Fri)</td>
<td>Tutorial</td>
</tr>
<tr>
<td>27-Mar-06(Mon)</td>
<td>Tutorial</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Date</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>20-Mar-06(Mon)</td>
<td>Lecture</td>
</tr>
<tr>
<td>21-Mar-06(Tue)</td>
<td>Lecture</td>
</tr>
<tr>
<td>22-Mar-06(Wed)</td>
<td>Lecture</td>
</tr>
<tr>
<td>23-Mar-06(Thu)</td>
<td>Lecture</td>
</tr>
<tr>
<td>24-Mar-06(Fri)</td>
<td>Lecture</td>
</tr>
<tr>
<td>27-Mar-06(Mon)</td>
<td>Lecture</td>
</tr>
</tbody>
</table>

**Detail listing of student attendance**
6.3.10 Student Profile Monitor (Detailed Contd.)

MSN Type popup is displayed at the bottom right hand corner to provide more assistance to the user of this page.

Student Details

Field

StudentId
IntakeID
Name
MRIC
Email
Hpo
Telno

Student Gradebook

Date
Grade
Marks
Comments

Assign 1
A
10/10/2006 12:00:00 AM
79
Good work

Assign 2
A
10/25/2006 12:00:00 AM
65
Satisfactory

Tutorial 1
T
2/26/2006 12:00:00 AM
78
Satisfactory

Tutorial 2
T
3/14/2006 12:00:00 AM
69
Poor

Tutorial 3
T
3/16/2006 12:00:00 AM
74
Good

Student Motivation

QuestionID
QuestionDesc

1
Strongly Agree
2
Agree
3
Disagree
4
Strongly Disagree

MSN type message box will popup when the following icon is clicked. This popup is aimed to provide more interactivity and additional help for the user of the website.
6.3.11 Page under Construction

No prototype will be complete with the famous ‘under construction’ page. In keep with the essence of an easy to navigate and user friendly web-site the following page is displayed when any not completed links are clicked.
6.3.12 Print

Print any page on the website by clicking the print button on the top right hand corner.
Appendix A

TEACHERS QUESTIONNAIRES

These questionnaires are to be answered by FSKTM lecturers with more than 1 year experience teaching introductory programming as part of UM curriculum.

OBJECTIVE/INSTRUCTIONS

Question 1 - 8
The following are examples of constructivist strategies that are commonly employed by educators to improve the process of teaching introductory programming. Kindly state if you have employed these strategies together with the frequency and effectiveness to which it has been applied in the teaching of introductory programming.

Question 9 - 17
These set of questions are aimed to obtain the academic performance of students based on the current teaching delivery strategies/techniques. Kindly state and describe your feedback toward the questions below.

Question 18 - 20
These question attempts to obtain your views on the introduction of the alternative introductory program sequence which is to be fitted into the current teaching environment and your perception of introducing tutor to assist in the teaching of introductory programming. Kindly state and describe your feedback towards the questions below.

Question 21 - 27
These questions are aimed to obtain feedback from the educator on the need and importance of identifying the cognitive characteristic of students to enhance the teaching of introductory programming. Kindly state and describe your feedback towards the questions below.
CONSTRUCTIVIST STRATEGY : CODE WALKTHROUGH

“Students are assigned to groups and given a set of pseudo-code, flowchart or programming code to which they are required to trace thru and predict the relevant output. It is important to note that each group member is solely responsible for tracing through only a section of the code. This will ensure that the strongest student in the group does not dominate this activity and each student contributes and produces their respective solutions.”

(i) Have you experienced the following method being applied in your introductory programming subjects?

☐ YES  ☐ NO  ☐ SOMewhat EXPERIENCED

Please continue to answer QUESTION 1a if you selected “YES/SOMewhat EXPERIENCED” or QUESTION 1b if you selected “NO”.

(If you selected “YES/SOMewhat EXPERIENCED for QUESTION 1)  

(i) Please indicate the number of lessons to which you have applied this strategy:

<table>
<thead>
<tr>
<th>Lessons</th>
<th>1 – 3 lessons</th>
<th>4 – 6 lessons</th>
<th>7 – 10 lessons</th>
<th>11 - 14 lessons</th>
<th>&gt; 15 lessons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

(ii) Please indicate the effectiveness of this strategy in terms of improving student UNDERSTANDING towards the topic or contents being taught:

<table>
<thead>
<tr>
<th>Effectiveness</th>
<th>Extremely Ineffective</th>
<th>Not Effective</th>
<th>Satisfactory</th>
<th>Effective</th>
<th>Very Effective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

(iii) Please provide your comments to validate why you find the above mentioned strategy effective or ineffective?

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

(If you selected “NO” for QUESTION 1)  

Do you feel that the above strategy is effective in improving the teaching of introductory programming?

(i) ☐ YES

Please provide your comments to validate why you find the above mentioned strategy effective?

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

(ii) ☐ NO

Please provide your comments to validate why you find the above mentioned strategy ineffective?

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________
CONSTRUCTIVIST STRATEGY: WRITING CODE
“Students are assigned in pairs or groups to write codes that solve a small problem. While the students are working together and solving the problem, you (teacher) continue to walk around the room and observe the students working together. Guidance by the teachers will be provided when appropriate but this assistance will be limited as the teachers are encouraged to remain quite and to allow students to answer their own questions.”

(ii) Have you experienced the following method being applied in your introductory programming subjects?
☐ YES ☐ NO ☐ SOMEWHAT EXPERIENCED

Please continue to answer QUESTION 2a if you selected “YES/SOMEWHAT EXPERIENCED or QUESTION 2b if you selected “NO”.

(If you selected “YES/SOMEWHAT EXPERIENCED for QUESTION 2)

(i) Please indicate the number of lessons to which you have applied this strategy:
1 – 3 lessons 4 – 6 lessons 7 – 10 lessons 11-14 lessons > 15 lessons

(ii) Please indicate the effectiveness of this strategy in terms of improving student UNDERSTANDING towards the topic or contents being taught:
Extremely Ineffective Not Effective Satisfactory Effective Very Effective

(iii) Please provide your comments to validate why you find the above mentioned strategy effective or ineffective?
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(If you selected “NO” for QUESTION 2)

Do you feel that the above strategy is effective in improving the teaching of introductory programming?

(i) ☐ YES
Please provide your comments to validate why you find the above mentioned strategy effective?
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(ii) ☐ NO
Please provide your comments to validate why you find the above mentioned strategy ineffective?
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
**CONTRACTIVIST STRATEGY: WRITING CODE**

“Students are assigned one or two small problems to solve before coming to class. Once completed, each student must bring their completed solutions to class and break into their individual groups. Each group member will walk through each other’s code to compare solutions and then collaborate on a group solution.”

(i) Have you experienced the following method being applied in your introductory programming subjects?

☐ YES  ☐ NO  ☐ SOMEWHAT EXPERIENCED

Please continue to answer QUESTION 3a if you selected “YES/SOMEWHAT EXPERIENCED” or QUESTION 3b if you selected “NO”.

(If you selected “YES/SOMEWHAT EXPERIENCED for QUESTION 3)

(i) Please indicate the number of lessons to which you have applied this strategy:

- 1 – 3 lessons
- 4 – 6 lessons
- 7 – 10 lessons
- 11 - 14 lessons
- > 15 lessons

(ii) Please indicate the effectiveness of this strategy in terms of improving student UNDERSTANDING towards the topic or contents being taught:

- Extremely Ineffective
- Not Effective
- Satisfactory
- Effective
- Very Effective

(iii) Please provide your comments to validate why you find the above mentioned strategy effective or ineffective?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(If you selected “NO” for QUESTION 3)

Do you feel that the above strategy is effective in improving the teaching of introductory programming?

(i) ☐ YES

Please provide your comments to validate why you find the above mentioned strategy effective?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(ii) ☐ NO

Please provide your comments to validate why you find the above mentioned strategy ineffective?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
CONSTRUCTIVIST STRATEGY: SCAFFOLDING

Students are divided into groups of two and given code that solves a problem. Students are then required to insert comments to describe the semantics of the code.

(i) Have you experienced the following method being applied in my introductory programming subjects?

☐ YES  ☐ NO  ☐ SOMEWHAT EXPERIENCED

Please continue to answer QUESTION 4a if you selected “YES/SOMEWHAT EXPERIENCED” or QUESTION 4b if you selected “NO”.

(If you selected “YES/SOMEWHAT EXPERIENCED for QUESTION 4)

(i) Please indicate the number of lessons to which you have applied this strategy:

<table>
<thead>
<tr>
<th>Number of Lessons</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 – 3 lessons</td>
<td>1</td>
</tr>
<tr>
<td>4 – 6 lessons</td>
<td>2</td>
</tr>
<tr>
<td>7 – 10 lessons</td>
<td>3</td>
</tr>
<tr>
<td>11 - 14 lessons</td>
<td>4</td>
</tr>
<tr>
<td>&gt; 15 lessons</td>
<td>5</td>
</tr>
</tbody>
</table>

(ii) Please indicate the effectiveness of this strategy in terms of improving student UNDERSTANDING towards the topic or contents being taught:

<table>
<thead>
<tr>
<th>Extremely Ineffective</th>
<th>Not Effective</th>
<th>Satisfactory</th>
<th>Effective</th>
<th>Very Effective</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

(iii) Please provide your comments to validate why you find the above mentioned strategy effective or ineffective?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(If you selected “NO” for QUESTION 4)

Do you feel that the above strategy is effective in improving the teaching of introductory programming?

(i)  ☐ YES

Please provide your comments to validate why you find the above mentioned strategy effective?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(ii) ☐ NO

Please provide your comments to validate why you find the above mentioned strategy ineffective?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
CONSTRUCTIVIST STRATEGY: CODE DEBUGGING

“Students are given syntactically and logically buggy code and are required to identify errors in a code. Student will first complete this activity individually and then be divided into groups to share their results.”

(i) Have you experienced the following method being applied in my introductory programming subjects?

☐ YES  ☐ NO  ☐ SOMEWHAT EXPERIENCED

Please continue to answer QUESTION 5a if you selected “YES/SOMEWHAT EXPERIENCED” or QUESTION 5b if you selected “NO”.

(If you selected “YES/SOMEWHAT EXPERIENCED” for QUESTION 5)

(ii) Please indicate the number of lessons to which you have applied this strategy:

1 – 3 lessons  4 – 6 lessons  7 – 10 lessons  11 - 14 lessons  > 15 lessons

1  2  3  4  5

(iii) Please indicate the effectiveness of this strategy in terms of improving student UNDERSTANDING towards the topic or contents being taught:

Extremely Ineffective  Not Effective  Satisfactory  Effective  Very Effective

1  2  3  4  5

Please provide your comments to validate why you find the above mentioned strategy effective or ineffective?

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

(If you selected “NO” for QUESTION 5)

Do you feel that the above strategy is effective in improving the teaching of introductory programming?

(i)  ☐ YES

Please provide your comments to validate why you find the above mentioned strategy effective?

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________

(ii)  ☐ NO

Please provide your comments to validate why you find the above mentioned strategy ineffective?

______________________________________________________________________

______________________________________________________________________

______________________________________________________________________
CONTRUCTIVIST STRATEGY: LECTURE NOTE RECONSTRUCTION

Ask students to not take notes during a mini lecture but instead to pay close attention. After 15 minutes of lecture they have a few minutes to reconstruct an outline of the lecture from memory. They then meet in their groups and refine their notes further.

(i) Have you experienced the following strategy being applied in my introductory programming subjects?

☐ YES  ☐ NO  ☐ SOMEWHAT EXPERIENCED

Please continue to answer QUESTION 6a if you selected “YES/SOMEWHAT EXPERIENCED” or QUESTION 6b if you selected “NO”.

(If you selected “YES/SOMEWHAT EXPERIENCED” for QUESTION 6)

(i) Please indicate the number of lessons to which you have applied this strategy:

<table>
<thead>
<tr>
<th>Number of Lessons</th>
<th>1 – 3 lessons</th>
<th>4 – 6 lessons</th>
<th>7 – 10 lessons</th>
<th>11 – 14 lessons</th>
<th>&gt; 15 lessons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choice</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

(ii) Please indicate the effectiveness of this strategy in terms of improving student UNDERSTANDING towards the topic or contents being taught:

<table>
<thead>
<tr>
<th>Effectiveness</th>
<th>Extremely Ineffective</th>
<th>Not Effective</th>
<th>Satisfactory</th>
<th>Effective</th>
<th>Very Effective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choice</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

(iii) Please provide your comments to validate why you find the above mentioned strategy effective or ineffective?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(If you selected “NO” for QUESTION 6)

Do you feel that the above strategy is effective in improving the teaching of introductory programming?

(i) ☐ YES

Please provide your comments to validate why you find the above mentioned strategy effective?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(ii) ☐ NO

Please provide your comments to validate why you find the above mentioned strategy ineffective?

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
CONSTRUCTIVIST STRATEGY: PAIR PROGRAMMING

“Students are paired together to work collaboratively at one computer on the same design, algorithm or code. Each student is required to communicate and solve any problem that arises with their partner while working on the computer.”

(i) Have you experienced the following method being applied in my introductory programming subjects?
☐ YES  ☐ NO  ☐ SOMewhat EXPERIENCED

Please continue to answer QUESTION 7a if you selected “YES/SOMewhat EXPERIENCED” or QUESTION 7b if you selected “NO”.

(If you selected “YES/SOMewhat EXPERIENCED” for QUESTION 7)

(i) Please indicate the number of lessons to which you have applied this strategy:

<table>
<thead>
<tr>
<th>Lessons</th>
<th>1–3 lessons</th>
<th>4–6 lessons</th>
<th>7–10 lessons</th>
<th>11–14 lessons</th>
<th>&gt;15 lessons</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

(ii) Please indicate the effectiveness of this strategy in terms of improving student UNDERSTANDING towards the topic or contents being taught:

<table>
<thead>
<tr>
<th>Effectiveness</th>
<th>Extremely Ineffective</th>
<th>Not Effective</th>
<th>Satisfactory</th>
<th>Effective</th>
<th>Very Effective</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

(iii) Please provide your comments to validate why you find the above mentioned strategy effective or ineffective?
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(If you selected “NO” for QUESTION 7)

Do you feel that the above strategy is effective in improving the teaching of introductory programming?

(i) ☐ YES

Please provide your comments to validate why you find the above mentioned strategy effective?
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(ii) ☐ NO

Please provide your comments to validate why you find the above mentioned strategy ineffective?
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
CONTRIVIST STRATEGY: DISCUSSION CLASSES

“Discussion classes are introduced in addition to the normal lecture and lab work. Each discussion class is conducted in a group and without the presence of any computers. Teachers walk around the room and observe groups working together and provide guidance when required.”

(i) Have you experienced the following method being applied in my introductory programming subjects?
☐ YES  ☐ NO  ☐ SOMEWHAT EXPERIENCED

Please continue to answer QUESTION 8a if you selected “YES/SOMEWHAT EXPERIENCED” or QUESTION 8b if you selected “NO”.

(If you selected “YES/SOMEWHAT EXPERIENCED” for QUESTION 8)

(i) Please indicate the number of lessons to which you have applied this strategy:

<table>
<thead>
<tr>
<th>1 – 3 lessons</th>
<th>4 – 6 lessons</th>
<th>7 – 10 lessons</th>
<th>11 – 14 lessons</th>
<th>&gt; 15 lessons</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

(ii) Please indicate the effectiveness of this strategy in terms of improving student understanding towards the topic or contents being taught:

<table>
<thead>
<tr>
<th>Extremely Ineffective</th>
<th>Not Effective</th>
<th>Satisfactory</th>
<th>Effective</th>
<th>Very Effective</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

(iii) Please provide your comments to validate why you find the above mentioned strategy effective or ineffective?
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________

(If you selected “NO” for QUESTION 8)

(i) Do you feel that the above strategy is effective in improving the teaching of introductory programming?
☐ YES
☐ NO

Please provide your comments to validate why you find the above mentioned strategy effective?
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
How many weeks have been allocated per semester to cover and complete the teaching of this subject? ____ weeks

How many topics/modules are required to be completed within this time period? ____ topics/modules

How many times is the subject taught in a week?

<table>
<thead>
<tr>
<th>1-2 times per week</th>
<th>3 - 4 times per week</th>
<th>More than 4 times per week</th>
<th>Other (please specify)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

What is the duration of each lesson?

<table>
<thead>
<tr>
<th>0 – 60mins</th>
<th>61 – 120mins</th>
<th>121 – 180mins</th>
<th>Other (please specify)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

How much class time is lecture based (where you as a lecturer present and explain the relevant topics with the aid of slides, notes or textbooks)?

<table>
<thead>
<tr>
<th>0 – 60mins</th>
<th>61 – 120mins</th>
<th>121 – 180mins</th>
<th>Other (please specify)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

How much class time is tutorial (where student engaged in activities such as group or lab assignments) based?

<table>
<thead>
<tr>
<th>0 – 60mins</th>
<th>61 – 120mins</th>
<th>121 – 180mins</th>
<th>Other (please specify)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Are there any other techniques other than “lecture” or “tutorial” based approaches used to conduct a lesson.

□ YES  □ NO

(i) If “YES”, please state what are the techniques
______________________________________________________________________________
______________________________________________________________________________
______________________________________________________________________________

In your opinion is the required time given during a semester to teach introductory programming sufficient to embark student on the required knowledge?

□ YES  □ NO

(i) If “NO”, please state your suggestions to improve this situation
______________________________________________________________________________
______________________________________________________________________________


Based on your previous teachings, kindly provide the average grade score (exam + coursework) obtained by students in this subject.

<table>
<thead>
<tr>
<th>Grade Score</th>
<th>&gt; 75%</th>
<th>60% - 74%</th>
<th>50% - 59%</th>
<th>40% - 50%</th>
<th>30% - 39%</th>
<th>&lt; 30%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage of Student</td>
<td>___%</td>
<td>___%</td>
<td>___%</td>
<td>___%</td>
<td>___%</td>
<td>___%</td>
</tr>
</tbody>
</table>

Do you feel that the introduction of an alternative introductory program sequence where students who obtain lower than a C grade are required to revisit topics from the previous programming course before moving to the following level beneficial?

☐ YES  ☐ NO

What do you think are the concerns or challenges in implementing the above mentioned approach?

☐ Shortage of lecturers.

☐ Insufficient time due to the large number of subject to cover during an academic year.

☐ Other (please comment)

Do you think that the introduction of tutor to assist during lessons creates a positive impact on the teaching of introductory programming?

☐ YES  ☐ NO

(i) What are the advantages or disadvantage of having a tutor?

**Advantage**

_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________

**Disadvantage**

_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
Do you determine the type of learner (i.e. active, reflective, sensing, intuitive, visual, verbal, sequential, global, inductive and deductive learners) a student is?

☐ YES  ☐ NO

If you selected “YES” for the previous question.
(i) How do you determine the type of learner a student is?

_____________________________________________________________________________
_____________________________________________________________________________

If you selected “NO” for the previous question.
(ii) Do you think that it is important to determine the type of learner a student is?

☐ YES  ☐ NO

Why do you think that it is **important** or **not important** to determine the type of learner a student is?

_____________________________________________________________________________
_____________________________________________________________________________

Do you determine the type of motivation (i.e. extrinsic, intrinsic, social, achievement and null) a student has in pursuing an introductory programming subject?

☐ YES  ☐ NO

If you selected “YES” for the previous question.
(ii) How do you determine the type of motivation a student has?

_____________________________________________________________________________
_____________________________________________________________________________

If you selected “NO” for the previous question.
(i) Do you think that it is important to determine the type of motivation a student has?

☐ YES  ☐ NO

Why do you think that it is important or not important to determine the type of motivation student has?

_____________________________________________________________________________
_____________________________________________________________________________

If you were aware what type of learner and the motivation of your students, how would you use this information to improve the process of teaching introductory programming?

_____________________________________________________________________________
_____________________________________________________________________________
Do you feel that students who have prior experience in introductory programming will likely to perform better compared with those without prior experience?

☐ YES  ☐ NO

If you selected “YES” for the previous question.
(i) How does prior experience influence the academic performance in introductory programming?
___________________________________________________________________________
___________________________________________________________________________
___________________________________________________________________________

If you selected “NO” for the previous question.
(i) Do you think that it is important that students who intend to pursue introductory programming have prior experience on topics relevant to the subject (e.g. mathematics)?

☐ YES  ☐ NO

Why do you think that it is **important** or **not important** that students who intend to pursue introductory programming have prior experience on topics relevant to the subject (e.g. mathematics)?
### Appendix B

#### STUDENTS QUESTIONNAIRES

These questionnaires are to be answered by FSKTM undergraduates who are currently learning introductory programming as part of UM curriculum.

### QUESTION 1 – 2

These set of questions are aimed to identify the motivation of students in pursuing introductory programming. Kindly circle your answer for each corresponding question below.

### QUESTION 3 - 13

These set of questions are examples of constructivist strategies/application support tools that are commonly employed by educators to improve the process of teaching introductory programming. Kindly circle/tick the answers for the corresponding questions below to state if you have experienced these strategies being employed together with its impact on your academic performance and motivational/involvement level.

**IMPORTANT DEFINITION**

**MOTIVATIONAL LEVEL:** The degree to which students are inspired to learn and achieve the required goal in learning introductory programming.

**INVOLVEMENT LEVEL:** The degree to which students are continuous engaging and participating in activities that is beneficial and related towards the learning of introductory programming.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(i)</td>
<td><strong>Please indicate your current grade score for this subject</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0% – 39%</td>
<td>40% – 59%</td>
<td>60 - 79%</td>
<td>80 - 100%</td>
</tr>
<tr>
<td></td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>(ii)</td>
<td><strong>Please indicate your targeted grade score for this subject</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0% – 39%</td>
<td>40% – 59%</td>
<td>60 - 79%</td>
<td>80 - 100%</td>
</tr>
<tr>
<td></td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

(i) Do you have any prior programming experience?

☐ YES  ☐ NO

(ii) Do you feel that learning this subject is important?

☐ YES  ☐ NO

(iii) Why do you feel that learning this subject is important/not important?

______________________________________________________________________

(iv) Do you intend to take more advance programming subjects in your following academic year(s)?

☐ YES  ☐ NO

(v) Do you intend to pursue a carrier in programming?

☐ YES  ☐ NO
CONTRACTIVE STRATEGY: CODE WALKTHROUGH

“You are assigned to groups and given a set of pseudo-code, flowchart or programming code to which you are required to trace thru and predict the relevant output. It is important to note that each group member is solely responsible for tracing through only a section of the code. This will ensure that the strongest student in the group does not dominate this activity and each student contributes and produces their respective solutions.”

(i) Have you experienced the following strategy being applied in this subject?

☐ YES ☐ NO ☐ SOMEWHAT EXPERIENCED

(ii) Please indicate the impact this strategy (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.

☐ The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:

0%-24% 25% – 49% 50% - 74% 75% - 100% Don’t Know

☐ The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:

0%-24% 25% – 49% 50% - 74% 75% - 100% Don’t Know

☐ There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.

(iii) Please indicate the impact this strategy (has or will have) on the MOTIVATION level in the learning the subject.

☐ The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:

0%-24% 25% – 49% 50% - 74% 75% - 100% Don’t Know

☐ The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:

0%-24% 25% – 49% 50% - 74% 75% - 100% Don’t Know

☐ There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.
**CONSTRUCTIVIST STRATEGY: WRITING CODE**

“You are assigned in pairs or groups to write codes that solve a small problem. While you are working together and solving the problem, your teacher continues to walk around the room and observe you and your group working together. Guidance by the teachers will be provided when appropriate but this assistance will be limited as teachers are encouraged to remain quite and to allow students to answer their own questions.”

(i) **Have you experienced the following strategy being applied in this subject?**

- □ **YES**  □ **NO**  □ **SOMewhat EXPERIENCED**

(ii) **Please indicate the impact this strategy (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.**

- □ The above mentioned strategy (has or will) **increase** the results of my test, assignment, tutorial, quiz or examination by the following percentage:

  - 0%-24%
  - 25% – 49%
  - 50% - 74%
  - 75% - 100%
  - Don’t Know

- □ The above mentioned strategy (has or will) **decrease** the results of my test, assignment, tutorial, by the following percentage:

  - 0%-24%
  - 25% – 49%
  - 50% - 74%
  - 75% - 100%
  - Don’t Know

- □ There (has been or will be) **no changes** in the results of my test, assignment, tutorial, quiz or examination.

(iii) **Please indicate the impact this strategy (has or will have) on the MOTIVATION level in the learning the subject.**

- □ The above mentioned strategy (has or will) **increase** the results of my test, assignment, tutorial, quiz or examination by the following percentage:

  - 0%-24%
  - 25% – 49%
  - 50% - 74%
  - 75% - 100%
  - Don’t Know

- □ The above mentioned strategy (has or will) **decrease** the results of my test, assignment, tutorial, by the following percentage:

  - 0%-24%
  - 25% – 49%
  - 50% - 74%
  - 75% - 100%
  - Don’t Know

- □ There (has been or will be) **no changes** in the results of my test, assignment, tutorial, quiz or examination.
**CONTRACTIVIST STRATEGY: WRITING CODE**

“You are assigned one or two small problems to solve before coming to class. Once completed, you must bring your completed solutions to class and break into your individual groups. Each group member will walk through each other’s code to compare solutions and then collaborate on a group solution.”

(i) Have you experienced the following strategy being applied in this subject?

- [ ] YES
- [ ] NO
- [ ] SOMEWHAT EXPERIENCED

(ii) Please indicate the impact this strategy (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.

- [ ] The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:
  - [ ] 0%-24%
  - [ ] 25% - 49%
  - [ ] 50% - 74%
  - [ ] 75% - 100%
  - [ ] Don’t Know

- [ ] The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:
  - [ ] 0%-24%
  - [ ] 25% - 49%
  - [ ] 50% - 74%
  - [ ] 75% - 100%
  - [ ] Don’t Know

- [ ] There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.

(iii) Please indicate the impact this strategy (has or will have) on the MOTIVATION level in the learning the subject.

- [ ] The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:
  - [ ] 0%-24%
  - [ ] 25% - 49%
  - [ ] 50% - 74%
  - [ ] 75% - 100%
  - [ ] Don’t Know

- [ ] The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:
  - [ ] 0%-24%
  - [ ] 25% - 49%
  - [ ] 50% - 74%
  - [ ] 75% - 100%
  - [ ] Don’t Know

- [ ] There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.
CONSTRUCTIVIST STRATEGY: SCAFFOLDING
“You are divided into groups of two and given code that solves a problem. You are then required to insert comments to describe the semantics of the code.”

(i) Have you experienced the following strategy being applied in this subject?
□ YES □ NO □ SOMEWHAT EXPERIENCED

(ii) Please indicate the impact this strategy (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.
□ The above mentioned strategy (has or will) **increase** the results of my test, assignment, tutorial, quiz or examination by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

□ The above mentioned strategy (has or will) **decrease** the results of my test, assignment, tutorial, by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

□ There (has been or will be) **no changes** in the results of my test, assignment, tutorial, quiz or examination.

(iii) Please indicate the impact this strategy (has or will have) on the MOTIVATION level in the learning the subject.
□ The above mentioned strategy (has or will) **increase** the results of my test, assignment, tutorial, quiz or examination by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

□ The above mentioned strategy (has or will) **decrease** the results of my test, assignment, tutorial, by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

□ There (has been or will be) **no changes** in the results of my test, assignment, tutorial, quiz or examination.
**CONSTRUCTIVIST STRATEGY : CODE DEBUGGING**

“You are given syntactically and logically buggy code and are required to identify errors in a code. You will first complete this activity individually and then be divided into groups to share your results.”

(i) **Have you experienced the following strategy being applied in this subject?**

- [ ] YES  
- [ ] NO  
- [ ] SOMEWHAT EXPERIENCED

(ii) **Please indicate the impact this strategy (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.**

- [ ] The above mentioned strategy (has or will) **increase** the results of my test, assignment, tutorial, quiz or examination by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
</tbody>
</table>

- [ ] The above mentioned strategy (has or will) **decrease** the results of my test, assignment, tutorial, by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
</tbody>
</table>

- [ ] There (has been or will be) **no changes** in the results of my test, assignment, tutorial, quiz or examination.

(iii) **Please indicate the impact this strategy (has or will have) on the MOTIVATION level in the learning the subject.**

- [ ] The above mentioned strategy (has or will) **increase** the results of my test, assignment, tutorial, quiz or examination by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
</tbody>
</table>

- [ ] The above mentioned strategy (has or will) **decrease** the results of my test, assignment, tutorial, by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
</tbody>
</table>

- [ ] There (has been or will be) **no changes** in the results of my test, assignment, tutorial, quiz or examination.
CONSTRUCTIVIST STRATEGY: LECTURE NOTE RECONSTRUCTION

“You are required to pay close attention and not take any notes during a 15 minutes lecture. You are then given few minutes to reconstruct an outline of the lecture from memory. Finally, you will meet your groups and refine your notes further.”

(i) Have you experienced the following strategy being applied in this subject?
☐ YES ☐ NO ☐ SOMEWHAT EXPERIENCED

(ii) Please indicate the impact this strategy (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.

☐ The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:

<table>
<thead>
<tr>
<th>0%-24%</th>
<th>25% - 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

☐ The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:

<table>
<thead>
<tr>
<th>0%-24%</th>
<th>25% - 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

☐ There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.

(iii) Please indicate the impact this strategy (has or will have) on the MOTIVATION level in the learning the subject.

☐ The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:

<table>
<thead>
<tr>
<th>0%-24%</th>
<th>25% - 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

☐ The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:

<table>
<thead>
<tr>
<th>0%-24%</th>
<th>25% - 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

☐ There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.
CONSTRUCTIVIST STRATEGY: PAIR PROGRAMMING

“You are paired together with another student to work collaboratively at one computer on the same design, algorithm or code. You and your partner are required to communicate and attempt to solve any problem that arises while working on the computer.”

(i) Have you experienced the following strategy being applied in this subject?

☐ YES  ☐ NO  ☐ SOMEWHAT EXPERIENCED

(ii) Please indicate the impact this strategy (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.

☐ The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:

0%-24%  25%–49%  50%–74%  75%–100%  Don’t Know

☐ The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:

0%-24%  25%–49%  50%–74%  75%–100%  Don’t Know

☐ There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.

(iii) Please indicate the impact this strategy (has or will have) on the MOTIVATION level in the learning the subject.

☐ The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:

0%-24%  25%–49%  50%–74%  75%–100%  Don’t Know

☐ The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:

0%-24%  25%–49%  50%–74%  75%–100%  Don’t Know

☐ There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.
CONSTRUCTIVIST STRATEGY: DISCUSSION CLASSES

“Discussion classes are introduced in addition to the normal lecture and lab work. Each discussion class is conducted in a group and without the presence of any computers. Teachers walk around the room and observe groups working together and provide guidance when required.”

(i) Have you experienced the following strategy being applied in this subject?
☐ YES ☐ NO ☐ SOMEWHAT EXPERIENCED

(ii) Please indicate the impact this strategy (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.

☐ The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

☐ The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

☐ There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.

(iii) Please indicate the impact this strategy (has or will have) on the MOTIVATION level in the learning the subject.

☐ The above mentioned strategy (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

☐ The above mentioned strategy (has or will) decrease the results of my test, assignment, tutorial, by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

☐ There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.
SUPPORT TOOLS : SUBJECT WEBSITE (STATIC CONTENTS)

“A website is developed where subject materials such as lecture notes, presentation slides, past year questions, timetables and other relevant online archives are available and accessible for all students within and outside of the University”

(i) Have you experienced the following support tools being applied in this subject?  
☐ YES ☐ NO ☐ SOMEWHAT EXPERIENCED

(ii) Please indicate the impact this support tool (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.  
☐ The above mentioned support tool (has or will) increase the results of my test, assignment, tutorial, quiz or examination by the following percentage:  
0%-24% 25% – 49% 50% - 74% 75% - 100% Don’t Know  
☐ The above mentioned support tool (has or will) decrease the results of my test, assignment, tutorial, quiz or examination by the following percentage:  
0%-24% 25% – 49% 50% - 74% 75% - 100% Don’t Know  
☐ There (has been or will be) no changes in the results of my test, assignment, tutorial, quiz or examination.

(iii) Please indicate the impact this support tool (has or will have) on the INVOLVEMENT level in the learning the subject.

☐ As the materials are easily available and accessible from the website, it is easier for me to refer to the relevant topics and subject contents. This in return (has or will) increase my interest and involvement level in learning the subject by the following percentage:  
0%-24% 25% – 49% 50% - 74% 75% - 100% Don’t Know  
☐ Even though the materials are easily available and accessible from the website, I find it more difficult to refer and understand the subject contents. This in return (has or will) decrease my interest and involvement level in learning the subject by the following percentage:  
0%-24% 25% – 49% 50% - 74% 75% - 100% Don’t Know  
☐ Even though the materials are easily available and accessible from the website, there (has been or will be) no changes in my interest or involvement level in learning the subject.
**SUPPORT TOOLS : SUBJECT WEBSITE (INTERACTIVE CONTENTS)**

“*A website is developed where online discussion tools such as discussion boards, forums, emails and chat engine among student, tutor and lectures are accessible and available by all relevant parties to the subject.*”

(i) **Have you experienced the following support tools being applied in this subject?**

- [ ] YES  
- [ ] NO  
- [ ] SOMEWHAT EXPERIENCED

(ii) **Please indicate the impact this support tool (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.**

- [ ] The above mentioned support tool (has or will) **increase** the results of my test, assignment, tutorial, quiz or examination by the following percentage:
  - [ ] 0%-25%  
  - [ ] 25% - 50%  
  - [ ] 50% - 75%  
  - [ ] 75% - 100%  
  - [ ] Don’t Know

- [ ] The above mentioned support tool (has or will) **decrease** the results of my test, assignment, tutorial, by the following percentage:
  - [ ] 0%-25%  
  - [ ] 25% - 50%  
  - [ ] 50% - 75%  
  - [ ] 75% - 100%  
  - [ ] Don’t Know

- [ ] There (has been or will be) **no changes** in the results of my test, assignment, tutorial, quiz or examination.

(iii) **Please indicate the impact this support tool (has or will have) on the INVOLVEMENT level in the learning the subject.**

- [ ] Due to the increased level of interaction between students, tutors and lecturer via online discussion tools, it is easier for me to discuss and obtain feedback on relevant topics and subject contents. The anonymity present via online material enables me to be more comfortable in asking all questions pertaining to the subject. This in return (has or will) **increase** my interest and involvement level in learning the subject by the following percentage:
  - [ ] 0%-24%  
  - [ ] 25% - 49%  
  - [ ] 50% - 74%  
  - [ ] 75% - 100%  
  - [ ] Don’t Know

- [ ] Even though with the increased level of interaction between students, tutors and lecturers via online discussion tools, it is more difficult for me to discuss and obtain feedback on relevant topics and subjects. This in return (has or will) **decrease** my interest and involvement towards learning the subject by the following percentage:
  - [ ] 0%-24%  
  - [ ] 25% - 49%  
  - [ ] 50% - 74%  
  - [ ] 75% - 100%  
  - [ ] Don’t Know

- [ ] Even though with the increased level of interaction between students, tutors and lecturers via online discussion tools, there (has been or will be) **no changes** in my interest or involvement level in learning the subject.
SUPPORT TOOLS : INTERACTIVE MEDIA

“A set of application support tools inclusive of video clips and interactive media of subject contents, program examples and flows are made available and accessible by student, tutors and lectures.”

(i) **Have you experienced the following support tools being applied in this subject?**

□ YES □ NO □ SOMEWHAT EXPERIENCED

(ii) **Please indicate the impact this support tool (has or will have) on the RESULTS of your test, assignment, tutorial, quiz or examination.**

□ The above mentioned **support tool** (has or will) **increase** the results of my test, assignment, tutorial, quiz or examination by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-25%</th>
<th>25% – 50%</th>
<th>50% - 75%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

□ The above mentioned **support tool** (has or will) **decrease** the results of my test, assignment, tutorial, by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

□ There (has been or will be) **no changes** in the results of my test, assignment, tutorial, quiz or examination.

(iii) **Please indicate the impact this support tool (has or will have) on the INVOLVEMENT level in the learning the subject.**

□ The availability of interactive tools such video clips, program animators, application tools and IDE has enabled me to better understand and visualize relevant topics and the subject contents. This in return (has or will) **increase** my interest and involvement towards learning the subject by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

□ Even with the presence of interactive tools such as video clips, program animators, application tools, and IDE it is more difficult for me to understand and visualize relevant topics and subject contents. This has in return (has or will) **decrease** my interest and involvement towards learning the subject by the following percentage:

<table>
<thead>
<tr>
<th>Percentage</th>
<th>0%-24%</th>
<th>25% – 49%</th>
<th>50% - 74%</th>
<th>75% - 100%</th>
<th>Don’t Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

□ Even with the presence of interactive tools such as video clips, program animators, application tools, and IDE, there (has been or will be) **no changes** in my interest or involvement level in learning the subject.
References


Raadt, MD, Watson R and Toleman Mark (2002). *Language Trends in Introductory Programming Courses* University of Southern Queensland, Australia


