An Overview on Common Mistakes by Students for Introduction, Basic Elements and Selection Control Structure in Fundamentals of Computer Problem Solving (CSC128) Course

Naemah Abdul Wahab, Wan Anisha Wan Mohammad and Azlina Mohd Mydin
naema586@uitm.edu.my, wanan122@uitm.edu.my, azlin143@uitm.edu.my

Jabatan Sains Komputer & Matematik (JSKM), Universiti Teknologi MARA Cawangan Pulau Pinang, Malaysia

Introduction

Computer programming is an abstract subject that requires correct understanding, suitable teaching approaches and tools as well as proper learning materials to support the teaching and learning process of novice learners. According to Robins et. al (2003), they stated that du Boulay (1989) describe five domains that a novice programmers must mastered in learning programming are the general orientation of a program, the notational machine model, the notation of a programming language, programming structures and pragmatics. A brief explanation of each domain is stated in the Table 1.

Table 1: Five Computer Programming Domains (du Boulay, 1989)

Domain	**Description**
General Orientation	What programs are for and what can be done with them.
Notational Machine	A model of the computer as it relates to executing programs.
Notation	The syntax and semantics of a particular programming language.
Structures	Schemas or plans.
Pragmatics	The skills of planning, developing, testing, debugging and others.

In view of the above research on the key aspects of learning to program, the following section will discuss on the background of computer problem solving course that is offered in our university to a non-major of computer course learners, what are the common mistakes that these novice programmers done according to the first three chapters in our syllabus and a summarisation of suggestion for the learners to ponder.

Background of Computer Problem Solving Course (CSC128) for Novice Programmer

Fundamentals of Computer Problem Solving (CSC128) are an introductory course that highlights on various aspects of problem solving to using computer software. It primarily comprising of the problem domain, phases of problem solving via structured programming using Program Development Life Cycle (PDLC) and basic techniques in designing a solution using chosen programming language which is C++.

In Universiti Teknologi MARA Cawangan Pulau Pinang, Malaysia, CSC128 is a compulsory subject taken by students' from Diploma in Mechanical Engineering and Diploma in Civil Engineering. This subject is taught for 14 weeks during each semester and it consists of a two hours lecture session with another two hours of practical session which is conducted at computer laboratory. CSC128 covers five main topics which are the introduction to computer programs, component of a programming language, selection control structure, repetition control structure and function.

At the end of each semester, we hope that students will achieve three learning outcomes which are able to identify the steps and requirements of given problems using systematic problem solving approach, manage to write complete programs using structural and modular approach and finally demonstrating basic program to solve daily problem using designated programming control structures: selection, repetition and function.

Common Mistakes made by Students in CSC128 Course by Chapters

Fundamentals of Computer Problem Solving (CSC128) is an introductory programming course that entail five basic chapters which are the introduction to computer programs, component of a programming language, selection control structure, repetition control structure and function. As this subject is taken by students from non-computer major courses, our paper would like to highlight on some typical errors done by these programming beginners'. In this article, we will explain the common mistakes done by students based on CSC128 chapters which are from Chapter 1 until Chapter 3.

Introduction to computer programs

Among the usual misconception of students in the first chapter on introduction to computer programs are from the sub-section on Program Development Life Cycle (PDLC) as stated in Table 2.

Table 2: Common Mistakes in Introduction to Computer Programs

Chapter 1	**Common Mistakes**
Brief history of C++	None (Definition, Concept and Theory)
Preparation for programming: a) What is a computer program and importance of computer programming? b) Importance of good programs. Relationship between compilers, interpreters, assemblers	None (Definition, Concept and Theory)
Program Development Life Cycle: a) Problem solving phases: problem definition, algorithm design and implementation b) Analysis, design, coding, maintenance	✗ In the analysis phase, students are unable to understand the problem, thus, incapable of listing out the correct input, output and processes involve in a question given. ✗ In the design phase, learners are supposed to write out the pseudocode and flowchart for a stated problem. Unfortunately, some of the learners are incapable to use appropriate symbol when drawing the flowchart. ✗ In the implementation (coding) phase, students are able to write codes in C++. However, some of them are incompetent to correct their own errors (syntax, logic or runtime) without the assistance of their lecturer or friends. ✗ Most students do not experience problem in the testing phase as well as the maintenance phase.

To overcome this misunderstanding, we suggest the learners to follow the phases in the PDLC every time they want to write a code to solve a problem given. Novice programmers normally have the tendency of writing a complete coding (third step in PDLC) by skipping the first and second steps of PDLC, which are analysing a problem and designing the pseudocode or flowchart of a program. This practice contributes to the learners' incompetency to refine their understanding on problem-solving procedures (PDLC), whereby each problem can be solved by decomposing into steps that will be translated into lines of code. Thus, the lecturers can remind the students on importance of following the sequence of Problem Development Life Cycle phases at the beginning of each practical or tutorial session.

Component of a programming language

The second chapter on component of a programming language sees students' common mistakes on the sub-topics of variable naming and declaration, identification of data type, input statements, and C++ block structure as well as assignment concept that is specified by Table 3.

Table 3: Common Mistakes in Component of a Programming Language

Chapter 2	**Common Mistakes**
Identifier, variable, constant, statement	✖ Some students make careless mistakes by not following the rules in naming variables/identifiers. For example, they declared a variable name that consists of spaces in between the name.
Standard data type (int, float, double, char)	✖ A few students are still confused on the data type of a variable. If the variable is supposed to be a floating point number, for instance, they declared as integer. Some are still mixed up on the concept of non-number data type (char or string).
Input/output statement	✖ Students are easily overlooked on the input statements of non-number data which string character and string data type.
C++ block structure	✖ In the early stages of learning to code, some student writes the C++ block structure not in proper sequences. However, these errors are resolved later with a lot of exercises and practical sessions.
Arithmetic expressions	✖ Some of the students are confused with the % symbols as this symbols means modulus which the remainder of a division instead of percentage. Since most of the students usually use calculators to solve mathematical equation, they tend to make mistakes especially when they solve equations which involve division (/). The students must understand that an integer divide with another integer number, the answer will also be an integer. To make the answer a decimal number, one of the number must be a decimal number.
Operators - Unary operator ++, --,	✖ Some students might get confused between postfix and prefix whereby in some situation, the results for postfix and prefix will show no differences. However, in certain situation, prefix and postfix will show a different in the answer.
Assignment concepts	✖ There are students who are uncertain to assign value of non-numbers data (char or string).

These typical errors can be avoided through hands-on exercises done during the practical sessions. The more programming tutorial questions the novice learners' practice in class with the guidance of the lecturer, the more basic concepts that can be learned and easily remembered through mistakes made.

Selection control structure

Selection control structure exposes students' common errors on the relational and logical operators, the **if** statement as well as the **switch** statements as indicated by Table 4.

Table 4: Common Mistakes in Selection Control Structure

Chapter 3		**Common Mistakes**
Concept of condition - Boolean statements	✘	Most of the mistakes come from the relational and logical operators' application in writing a condition statement. In using logical operators, students usually mixed up between the usage of **AND (&&)** and **OR (**‖) in writing condition statement. Thus, the final output of the program produced logic error. Moreover, students also exchange symbols in the use of relational operators, for instance instead of using >=, the students mistakenly wrote =>.
Selection : One- way, two –way, multi-way , nested selection	✘	Among the mistake done in coding an **if** statement is writing a condition after an else statement. Another typical error by students will be missing curly brackets **{}** within an **if** statement that consist of multiple lines of code.
Selection : Switch statement	✘	Students usually make mistakes by overlooking the inverted comma **‘’** when using single character data type and putting **‘’** to data that is declared as integer. Another common error by students will be using the else term/keyword for default cases.

Programming tutorials and past years' examination question can be used as exercises during practical lessons to enhance understanding and exposed students to a variety of questions on selection control structures concept.

Conclusion

Learning programming needs a lot of understanding on abstract concepts. It requires continuously practical experiences and a lot of programming tutorials so that students will be able to understand the basic concepts and solve whatever problem given to them. With the knowledge that they have learnt in this course and the typical mistakes that we have acknowledged in this paper, we hope that it can assist the students in learning this course more efficiently and avoid them from making the same errors.

References:

du Boulay, B. (1989). Some difficulties of learning to program. In E. Soloway & J.C. Spohrer (Eds.), (pp. 283–299). Hillsdale, NJ: Lawrence Erlbaum.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. Computer science education, 13(2), 137-172.