Selection control structures

Selection control structures are those that do a certain series of codes depending on the condition the statement falls under (The Selection Control Structure, 2006). Examples of selection control structures are the all famous “ If… Else” statement, and Select Cases. The two are basically the same, only different in syntax. If a Select Case have multiple cases in one clause, the “ If… Else If… Else” statement is the counterpart of that particular control structure. In a program, a selection control structure is really important since it adds flexibility to the program.

Imagine a programming language without the power of selection control structures. It would probably take up lines and lines of codes, just to perform a single task. What if a lot of conditions are involved, it would take longer to complete. And to think is there any programming language not using a selection control structure? They all have control structures, just differing in syntax. Selection control structures make it really easy to manage program flows, especially if a series of conditions are present.

Just by specifying the conditions needed for different situation and the codes that are needed to be performed, a program can really be powerful. A simple example of a program containing selection control structures is the calculator. No matter if it is the calculator we see is on Windows, or that we do on exercises at school, it still has a selection control structure within its program code. It can be done in either an “ If… Else” Statement or a Select Case approach.

But, whatever approach is used, it is constructed containing at least four basic conditions – addition, subtraction, multiplication, and division. In the

said example program, given conditions which points to which operation to use, by testing certain keystrokes, or maybe if it’s a simple calculator, by specifying what operation you want to use before hand, the program can automatically select which code segment it should use. Assuming your simple code contains A, B, C, and D selections for Addition, Subtraction, Multiplication and Division respectively, then that means by selecting A, you can write a code that when A is selected, then it would perform the addition operation. So without the selection control structures, it would be immensely difficult to accomplish what appear to be just simple programs.

Probably, simple programs would be very complicated without the presence of selection control structures. So, if doing a simple calculator would be difficult without the aid of selection control structures, what more for complicated programs that require more conditions to be executed, it would really be tedious work. Which is why despite the simplicity of these control structures, they sure are a great help for program developing. Repetition Control Structures Repetition control structures, also known as the looping and iteration control structure, are the process of repeatedly executing a set of statements until the condition is met (The Repetition Control Structure, 2006). Compared to selection control structures, the algorithm being performed by the program loops, unless it reaches a certain condition it has to meet.

It basically has three parts – the loop termination decision, the body of the loop, and transferring it back to the start of the loop. Just like selection control structures, without the existence of repetition control structures, a code just to accomplish a certain task would take too many lines to

complete. Taking for example a program which prints the numbers 1 through 10, if asked to write the program without the aid of looping control structures, it would take more or less 10 lines to complete this simple task. You'd probably print the numbers one by one, numeral by numeral. What if you were asked to make a program which prints the numbers 1 through 1000? Would that mean you would be taking up 1, 000 lines to print it? Fortunately, that is what the repetition control structures are for.

You can just set a condition on when the program should stop executing. Even if you are asked to print 1 through 10, 000, then it wouldn't be much of a problem. It would only take about 5 to 15 lines maximum to print the numbers 1 to 10, 000, unless of course you have other set of conditions to perform aside from just printing the numerals. One good example of programs with repetition control structures are probably cash registers, whether at restaurants or grocery stores. Once it has been activated, items can be added for as long as there are items still available to swipe in the bar code reader.

But once the cashier types in a series, or even just a certain key, then the cash register program stops adding items, and adds up the total costs of the purchased items. In the example cited, the key the cash register keys in is where the program is conditioned to stop. But without the key or key combination being typed in, then the program will just keep adding items to the list, no matter how long the list goes. It would be really hard work if you'd have to keep telling the program to add more items.

So, repetition control structures come into good news in these types of situations. With the example mentioned, no matter how simple the program is, it still needs the help of repetition control structures. There are other programs requiring more of these control structure power. Without them, it would take forever to code the program, very tedious work, and once you commit a mistake, the whole program malfunctions, which make it hard for you to actually pinpoint where the problem is if you have a lot of program codes in the program.

With repetition control structures, you could just change a certain code segment if it needs to be altered, and the rest of the code does not have to suffer from the alteration.