ESC101 : Fundamental of computing

Lecture 12+13 (25 and 27 August, 2008)

In lecture 12 and 13, we covered the following topics.

- Revision of type casting and expression evaluation
- Program for representing integers in binary form.
- Program for representing fractions in binary form.
- Storage and representation of integer data types and floating point data types. (this topic was covered to provide some more insight into the numeric data types and is not part of the course).
- switch statement

Although I had provided all the details of the lecture 12 and 13 on black board, some students feel that the concept of switch statement is not clear to them. So I am providing the syntax and rules of execution of switch statement. I am also providing an example of `switch` statement below.

The `switch` statement is used as a compact alternative for those problems which involve many cases. The example at the end will make this point clear.

Syntax of `switch` statement :
The switch statement has the following syntax.

```
switch(expression)
{
  case  constant1  :  statements;
                       break;
  case  constant2  :  statements;
                       break;
               •
               •
               •
  case  constantk  :  statements;
                       break;
  default          :  statements;
                       break;
}
```

Here *expression* can be an expression of type **byte, short, int**, or **char** only; no other type of expression is permitted. Each of *constant1*,*constant2*,$\cdots$, *constantk* must be literals (constant) of same type as the type of *expression*. Moreover, no two literals can be of same value.

Execution of `switch` statement :
First the expression is evaluated, and let *val* be its value. Then *val* is successively tested against the value of the constants *constant1*,*constant2*,$\cdots$, *constantk*. When a match is found, that is, *constanti* happens to be same as *val*, then the statements associated with the ith case is executed. If no match is found, then the statements associated with the **default** case are executed. The statement **break** serves the following purpose. When this statement is encountered within the statement sequence of a **case**, the **break** statement causes the program counter (instruction pointer) to jump to the next statement outside the **switch** statement.

Example of `switch` statement :

Suppose we want to write a program which will do the following. There is a variable `month` of type `int`. It has been assigned some value. We would like to print the name of the month corresponding to the value of variable `month`. That is, if $month = 1$, program should print **January**, if $month = 2$, program should print **February**, and so on. However, if `month` does not have a valid value (it is less than 1 or greater than 12), the program should print **Invalid month**. We can do this task using a sequence of if statements. But the same can be done in a more compact manner as follows.

```
int month;
// month is assigned some integer value.
switch(month)
{
   case 1  : System.out.println(‘‘January’’);
             break;
   case 2  : System.out.println(‘‘February’’);
             break;
   case 3  : System.out.println(‘‘March’’);
             break;
   case 4  : System.out.println(‘‘April’’);
             break;
   case 5  : System.out.println(‘‘May’’);
             break;
   case 6  : System.out.println(‘‘June’’);
             break;
   case 7  : System.out.println(‘‘July’’);
             break;
   case 8  : System.out.println(‘‘August’’);
             break;
   case 9  : System.out.println(‘‘September’’);
             break;
   case 10 : System.out.println(‘‘October’’);
             break;
   case 11 : System.out.println(‘‘November’’);
             break;
   case 12 : System.out.println(‘‘December’’);
             break;
   default : System.out.println(‘‘Invalid month’’);
}
```

Let us consider the situation when `month` has value 9, then based on the execution rules mentioned above, the output will be

```
September
```

Please note that in the above example there is only one statement for each case (excluding the **break** statement). However,in general it is permitted to have multiple statements for a case. Also note that mentioning **default** case in a switch statement is optional. If we don't mention **default** case, and there is no match of the expression value with any of the constants of the cases, then no statement will be executed in the **switch**.

Remark : A student had asked whether it is permitted to place **int i = 4;** in expression of a switch ? The answer is **NO**. This is because **int i = 4;** is a statement and not an expression.