The Switch Statement

C++ Notes Mrs. Alano

There are situations where a program must take one of several actions depending on the value of some variable or expression. Such situations often arise when the program is processing commands, events, menu choices, or transactions of different types.
If the program has to handle just **two** or **three** possible actions, you can easily use the **if-else** statement. However, if the number of possible actions is large, the use of if-else becomes inefficient. The **switch statement** can be used for such situations.

General Form:

```
switch (expression)
{
      case A: //Take action A
            statementA1;
            statementA2;
            …
            break;
      case B: //Take action B
            statementB1;
            statementB2;
            …
            break;
      …. //Can have more cases from C – Y here.
      case Z: //Take action Z
            statementZ1;
            statementZ2;
            …
            break;
      default: //Take some default action
            ….
            break;
}
```

-- A,B, …, Z are integer or character **constants**. When a switch is compiled, the compiler creates
a table of these values and the associated addresses of the corresponding "cases" (code).
-- When the switch is executed, the program first evaluates expression to an integer. It then finds
it in the table and jumps to the corresponding "case".
-- If the value is not in the table, the program jumps to **default.**
-- The **break** statement at the end of each case tells the program to jump out of the *switch* and continue with the first statement after the switch.

****NOTE****
Break is used to break out of a loop or a switch. A *break* must be inside a loop or switch. In the case of nested loops or switches, a break tells the program to break out of the innermost loop or switch that contains it but does not affect the control flow in the outer switches or loops.