Lecture 4.1

Topics

1. Extended Conditional Structure – **`switch`** Statement

---

1. Extended Conditional Structure – `switch` Statement

C has one structure that can handle multiple options beside the extended **`if-else-if`** statements. This structure is called a **`switch`** statement, which is a composite statement used to make a selection among many options.

1.1 `switch` Syntax and Flowchart

Its syntax is given as follows,

```
switch ( testExpression ) {
  case constantValue1 :
    statement1
    break;
  case constantValue2 :
    statement2
    break;
  ………
  case constantValueN :
    statementN
    break;
  default :
    statementDefault
}
```

where

(1) **`testExpression`** must produce an integral value. It is commonly given as a unary expression in the form of an identifier.

(2) **`constantValue1, constantValue2, …, constantValueN`** represent all possible values matching with the above integral value (i.e., **testExpression** or its result).

The **`switch`** statement will have the following characteristics.

a. The test expression after the **`switch`** keyword must be an integral type.
b. The expression after the **`case`** keyword must be a constant expression. The expression together with the **`case`** keyword is called a **`case-label`** statement. Note that a constant expression is an expression that is evaluated at compiled time, not run time.
c. No two **`case`** labels can have the same value.
d. Two **`case`** labels can be associated with the same statements.
e. The **`default`** label is not required. If there is no match, then the control jumps outside of the **`switch`** statement.
f. There can be at most one **`default`** label. It can be placed anywhere, but it is mostly placed last in the structure.

A general flowchart is given in **Figures 1 & 2** as follows,

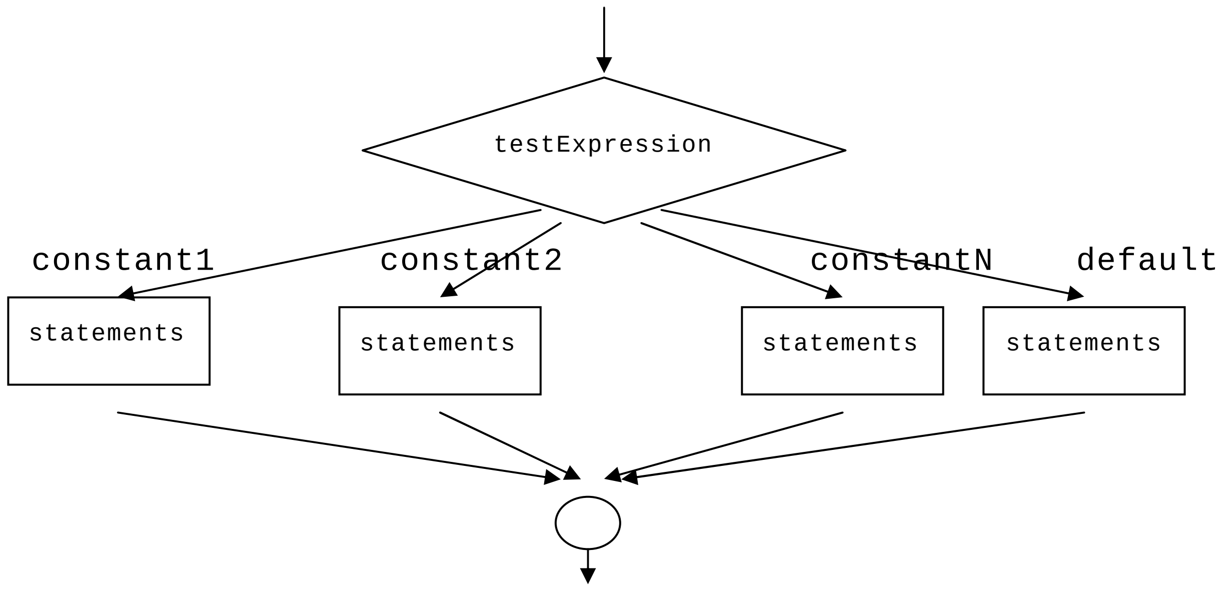

Figure 1 A general **`switch`** structure

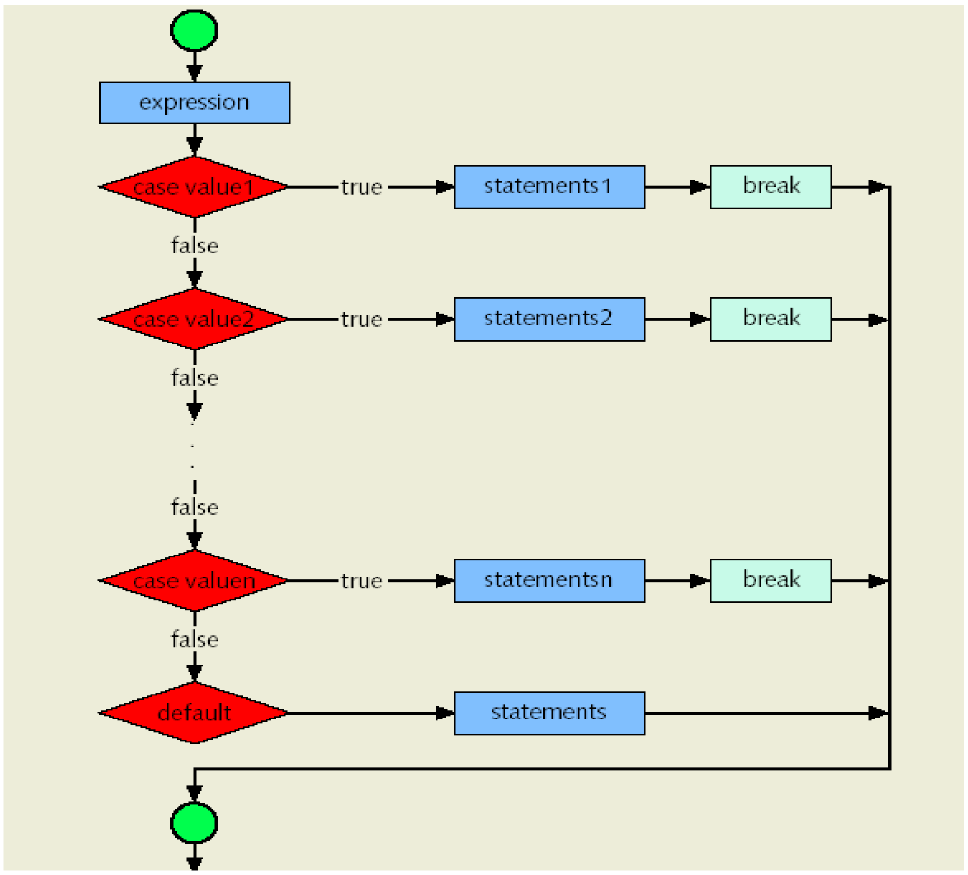

Figure 2 A **`switch`** structure

Then, the above **`printDaySwitch()`** example can be rewritten as follows,

```
void printDaySwitch( int iDay ) {

  switch( iDay ) {
    case 1: printf( "\nIt is Sunday!" );
            break;
    case 2: printf( "\nIt is Monday!" );
            break;
    case 3: printf( "\nIt is Tuesday!" );
            break;
    case 4: printf( "\nIt is Wednesday!" );
            break;
    case 5: printf( "\nIt is Thursday!" );
            break;
    case 6: printf( "\nIt is Friday!" );
            break;
    case 7: printf( "\nIt is Saturday!" );
            break;
    default: printf( "\nIt is an INVALID selection!" );
  }

  return;
}
```

1.2 Example – Menu setup

Recall that a menu program will provide the user with options and selections. The execution will continue after an option is selected and entered to the program.

Let’s consider a menu of four basic arithmetic operations:

(1) Add

(2), Subtract

(3) Multiply

(4) Divide

When the application (i.e., program) is run, the monitor will show the following output:

- Menu with options will be displayed, and
- The user must select and enter the (appropriate) option, and
- The program, based on the selection, will perform the desired operation.

Example

```
/**
 *Program Name: cis25L0411.cpp
 *Discussion:   Default Arguments
 *              Automatic Type Conversion (Type promotion)
 */

#include <iostream>
using namespace std;

/*Function prototypes*/

void displayMenu(void);
double add(double, double);
double subtract(double, double);
double multiply(double, double);
double divide(double, double);


int main() {
  int iOption;
  double dNum1;
  double dNum2;
  double dResult;

  displayMenu();

  cout << "\n\nSelect and enter an integer for option + ENTER: ";
  cin >> iOption;

  cout << "\nEnter first operand: ";
  cin >> dNum1;

  cout << "\nEnter second operand: ";
  cin >> dNum2;

  switch (iOption) {
    case 1:
      dResult = add(dNum1, dNum2);
      cout << "\n" << dNum1 << " + " << dNum2 << " --> "
        << dResult << endl;
      break;
    case 2:
      dResult = subtract(dNum1, dNum2);
      cout << "\n" << dNum1 << " - " << dNum2 << " --> "
        << dResult << endl;
      break;
    case 3:
      dResult = multiply(dNum1, dNum2);
      cout << "\n" << dNum1 << " * " << dNum2 << " --> "
        << dResult << endl;
      break;
    case 4:
      dResult = divide(dNum1, dNum2);
      cout << "\n" << dNum1 << " / " << dNum2 << " --> "
        << dResult << endl;
      break;
    default:
      cout << "\nInvalid Option!" << endl;
  }

  return 0;
}

/**
 *Function Name: displayMenu()
 *Description  : Displaying operation menu
 *Pre          : None
 *Post         : None
 */
void displayMenu() {
  cout << "\n  MENU:\n\t(1) Add\n\t(2) Subtract"
    << "\n\t(3) Multiply\n\t(4) Divide" << endl;
  return;
}

/**
 *Function Name: add()
 *Description  : Adding two numbers
 *Pre          : Two numbers
 *Post         : Sum of two numbers
 */
double add(double dOld1, double dOld2) {
  return (dOld1 + dOld2);
}

/**
 *Function Name: subtract()
 *Description  : Subtracting two numbers
 *Pre          : Two numbers
 *Post         : Difference of two numbers
 */
double subtract(double dOld1, double dOld2) {
  return (dOld1 - dOld2);
}

/**
 *Function Name: multiply()
 *Description  : Multiplying two numbers
 *Pre          : Two numbers
 *Post         : Product of two numbers
 */
double multiply(double dOld1, double dOld2) {
  return (dOld1 * dOld2);
}

/**
 *Function Name: divide()
 *Description  : Dividing two numbers
 *Pre          : Two numbers
 *Post         : Result of the division of two numbers
 */
double divide(double dOld1, double dOld2) {
  return (dOld1 / dOld2);
}
```

OUTPUT – Sample Run #1

```
  MENU:
        (1) Add
        (2) Subtract
        (3) Multiply
        (4) Divide


Select and enter an integer for option + ENTER: 1

Enter first operand: 4.0

Enter second operand: 5.0
```

```
4 + 5 --> 9
```

OUTPUT – Sample Run #2

```
  MENU:
        (1) Add
        (2) Subtract
        (3) Multiply
        (4) Divide


Select and enter an integer for option + ENTER: 2

Enter first operand: 4.0

Enter second operand: 5.0

4 - 5 --> -1
```

OUTPUT – Sample Run #3

```
  MENU:
        (1) Add
        (2) Subtract
        (3) Multiply
        (4) Divide


Select and enter an integer for option + ENTER: 3

Enter first operand: 4.0

Enter second operand: 5.0

4 * 5 --> 20
```

OUTPUT – Sample Run #4

```
  MENU:
        (1) Add
        (2) Subtract
        (3) Multiply
        (4) Divide


Select and enter an integer for option + ENTER: 4

Enter first operand: 4.0

Enter second operand: 5.0

4 / 5 --> 0.8
```