SWITCH STATEMENT:

if and if/else statements can become quite confusing when nested too deeply, and C++ offers an alternative. Unlike if, which evaluates one value, switch statements allow you to branch on any of a number of different values. The general form of the switch statement is:

```
switch(Expression)
{
   case Choice1:
      Statement1;
   case Choice2:
      Statement2;
   case Choice-n:
      Statement-n;
}
```

```
switch (expression)
{
case valueOne: statement;
break;
case valueTwo: statement;
break;
....
case valueN:   statement;
break;
default:       statement;
}
```

expression is any legal C++ expression, and the statements are any legal C++ statements or block of statements. switch evaluates expression and compares the result to each of the case values. Note, however, that the evaluation is only for equality; relational operators may not be used here, nor can Boolean operations.

WORKING:

If one of the case values matches the expression, execution jumps to those statements and continues to the end of the switch block, unless a break statement is encountered. If nothing matches, execution branches to the optional default statement. If there is no default and there is no matching value, execution falls through the switch statement and the statement ends.
It is important to note that if there is no break statement at the end of a case statement, execution will fall through to the next case statement. This is sometimes necessary, but usually is an error. If you decide to let execution fall through, be sure to put a comment, indicating that you didn't just forget the break.

This C++ program illustrates use of the switch statement.

EXAMPLE 1:

```
1:  //
2:  // Demonstrates switch statement
3:
4:  #include <iostream.h>
5:
6:  int main()
7:  {
8:    unsigned short int number;
9:    cout << "Enter a number between 1 and 5: ";
```

```
10:   cin >> number;
11:   switch (number)
12:   {
13:      case 0:   cout << "Too small, sorry!";
14:                break;
15:      case 5:  cout << "Good job!\n";  // fall through
16:      case 4:  cout << "Nice Pick!\n"; // fall through
17:      case 3:  cout << "Excellent!\n"; // fall through
18:      case 2:  cout << "Masterful!\n"; // fall through
19:      case 1:  cout << "Incredible!\n";
20:               break;
21:      default: cout << "Too large!\n";
22:               break;
23:   }
24:   cout << "\n\n";
25:    return 0;
26: }
```

Output:

```
 Enter a number between 1 and 5: 3
Excellent!
Masterful!
Incredible!

Enter a number between 1 and 5: 8
Too large!
```

Analysis:

The user is prompted for a number. That number is given to the switch statement. If the number is 0, the case statement on line 13 matches, the message Too small, sorry! is printed, and the break statement ends the switch. If the value is 5, execution switches to line 15 where a message is printed, and then falls through to line 16, another message is printed, and so forth until hitting the break on line 20.

The net effect of these statements is that for a number between 1 and 5, that many messages are printed. If the value of number is not 0-5, it is assumed to be too large, and the default statement is invoked on line 21.

EXAMPLE 2:

```
switch (choice)
{
case 0:
      cout << "Zero!" << endl;
      break
case 1:
      cout << "One!" << endl;
     break;
case 2:
     cout << "Two!" << endl;
default:
     cout << "Default!" << endl;
}
```

When defining an expression whose result would lead to a specific program execution, the switch statement considers that result and executes a statement based on the possible outcome of that expression, this possible outcome is called a case.

The different outcomes are listed in the body of the switch statement and each case has its own execution, if necessary. The body of a switch statement is delimited from an opening to a closing curly brackets: "{" to "}". The syntax of the switch statement is:

The expression to examine is an integer. Since an enumeration (enum) and the character (char) data types are just other forms of integers, they can be used too. Here is an example of using the switch statement:

EXAMPLE 3:

```
#include <iostream>
using namespace std;

int main()
{
	int Number;

	cout << "Type a number between 1 and 3: ";
	cin >> Number;

	switch (Number)
	{
	case 1:
		cout << "\nYou typed 1.";
	case 2:
		cout << "\nYou typed 2.";
	case 3:
		cout << "\nYou typed 3.";
	}

	return 0;
}
```

The program above would request a number from the user. If the user types 1, it would execute the first, the second, and the third cases. If she types 2, the program would execute the second and third cases. If she supplies 3, only the third case would be considered. If the user types any other number, no case would execute.

When establishing the possible outcomes that the switch statement should consider, at times there will be other possibilities other than those listed and you will be likely to consider them. This special case is handled by the default keyword. The default case would be considered if none of the listed cases matches the supplied answer. The syntax of the switch statement that considers the default case would be:

```
switch(Expression)
{
   case Choice1:
      Statement1;
   case Choice2:
      Statement2;
   case Choice-n:
      Statement-n;
   default:
      Other-Possibility;
}
```

EXAMPLE 4:

```
#include <iostream.h>
#include <iomanip.h>
#include <stdio.h>
#include <conio.h>
int main()
{
        int Number;

        cout << "Type a number between 1 and 3: ";
        cin >> Number;

        switch (Number)
        {
        case 1:
                cout << "\nYou typed 1.";
        case 2:
                cout << "\nYou typed 2.";
        case 3:
                cout << "\nYou typed 3.";
        default:
                cout << endl << Number << " is out of the requested range.";
        }

        return 0;
}
```

TASK:

Write C++ code for calculator using switch statement. Program should ask the user to select the operation and then perform the specified operation.