LAB 4.1 Relational Operators and the `if` Statement

Copy and paste the following program into Visual Studio IDE.

```
// This program tests whether or not an initialized value of num2
// is equal to a value of num1 input by the user.

#include <iostream>
using namespace std;

int main( )
{
  int num1,                         // num1 is not initialized
  num2 = 5;                         // num2 has been initialized to 5

  cout << "Please enter an integer" << endl;
  cin >> num1;
  cout << "num1 = " << num1 << " and num2 = " << num2 << endl;

  if (num1 = num2)
      cout << "The two values are the same. Hey, that’s a coincidence!" << endl;

  if (num1 != num2)
     cout << "The two values are not the same" << endl;

   return 0;
}
```

Exercise 1: Run the program several times using some input values that are not equal to the initialized value of num2, and a value that is equal to num2. Does the program do what you expect? If not, locate the error and fix it.

Exercise 2: Modify the program so that the user inputs both values for num1 and num2 to be tested for equality. Make sure you have a prompt for each input. Test the program with pairs of values that are the same and that are different.

Exercise 3: Modify the program so that when the numbers are the same it prints the following two lines:

```
The values are the same.
Hey that’s a coincidence!
```

Note: You must print each of the above two lines using separate `cout` statements.

(Print out the program and the running result for *Exercise 3*, and hand them in with the rest of the lab.

Note: Print out two copies of running result, one for two identical input values and another for two different input values.)

LAB 4.2 `if/else` and `if/else if` Statements

Copy and paste the following program into Visual Studio IDE.

```
// This program prints "You Pass" if a student's average is
// 60 or higher and prints "You Fail" otherwise

#include <iostream>
using namespace std;

int main()
{
   float average;                    // holds the grade average

   cout << "Input your average:" << endl;
   cin >> average;

   if (average > 60)
       cout << "You Pass" << endl;

   if (average < 60)
       cout << "You Fail" << endl;

   return 0;
}
```

Exercise 1: Run the program three times using 80, 55 and 60 for the average. What happens when you input 60 as the average? Modify the first if statement so that the program will also print “You Pass” if the average equals 60.

Exercise 2: Modify the program so that it uses an `if/else` statement rather than two `if` statements.

Exercise 3: Modify the program from Exercise 2 to allow the following five categories: Invalid data (grade is either above 100 or below 0), “A” category (90–100), “B” category (80–89), “You Pass” category (60–79), “You Fail” category (0–59).

(Print out the program and the running result for *Exercise 3,* and hand them in with the rest of the lab.

Note: Please print out five copies of running result, one for each category.)

Lab 4.3 Logical Operators

Copy and paste the following program into Visual Studio IDE.

```
// This program illustrates the use of logical operators

#include <iostream>
using namespace std;

int main()
{
    int year;
    float gpa;

    cout << "What year student are you ?" << endl;
    cout << "Enter 1 (freshman), 2 (sophomore), 3 (junior), or 4 (senior)"
         << endl << endl;
    cin >> year;

    cout << "Now enter your GPA" << endl;
    cin >> gpa;

    // Add decision (if) statements below to display "It is time to graduate soon"
    // if the student is a senior and the gpa is greater than or equal to 2.0.
    // If the student is not a senior or the gpa is less than 2.0,
    // display "You need more schooling".

    return 0;
}
```

Exercise 1: Complete the program and then run it. Test the program using different combinations of data.

Print out the program and two running results: one for input data representing a senior student with a gpa ≥ 2.0, and the other for input data representing a student who is a non-senior or his/her gpa is less than 2.0.

LAB 4.4 The `switch` Statement

Copy and paste the following program into Visual Studio IDE.

```
// This program illustrates the use of the switch statement.

#include <iostream>
using namespace std;

int main()
{

   char grade;

   cout << "What grade did you earn in Programming I ?" << endl;
   cin >> grade;

   switch( grade )            // This is where the switch statement begins
   {
       case 'A': cout << "an A - excellent work !" << endl;
                break;
       case 'B': cout << "you got a B - good job" << endl;
                break;
       case 'C': cout << "earning a C is satisfactory" << endl;
                break;
       case 'D': cout << "while  D is passing, there is a problem" << endl;
                break;
       case 'F': cout << "you failed - better luck next time" << endl
                break;
       default: cout << "You did not enter an A, B, C, D, or F" << endl;
   }

   return 0;
}
```

***Exercise 1**:* Add an additional `switch` statement that allows for a Passing option for a grade of D or better. Use the sample run given below to model your output.

Sample Run 1:

```
What grade did you earn in Programming I ?
A
YOU PASSED!
an A - excellent work!
```

Sample Run 2:

```
What grade did you earn in Programming I ?
F
You failed – better luck next time
```

Sample Run 3:

```
What grade did you earn in Programming I ?
J
You did not enter an A, B, C, D, or F
```

Exercise 2: Rewrite the program using `if / else if` statements rather than a `switch` statement. Use a trailing `else` in your new version to correspond to the default case in the `switch` statement.

(Print out the program and the running results for both *Exercise 1* and *Exercise 2*. Print out three copies of running results for each exercise corresponding to the three sample runs above.)