

Assignment 2

CS 135: Computer Science I
Spring 2007

Objectives

1. You will understand and demonstrate use of Logical and Relational Operators in `if-else` statements in C/C++
2. You will understand and demonstrate use of the stream abstraction in C/C++ especially with respect to input and output from and to a file.

1 Blackjack (10 points)

In Blackjack, the cards 2 through 10 are counted at their face values, regardless of suit. All face cards (J, Q, K) are counted as 10. An ace is special. An ace is counted as 11 only if the resulting sum of all card values in a player's hand (including the ace) does not exceed 21. Otherwise it is counted as 1.

Now, using this information write and test a C/C++ program that generates three card values as input, calculates the total value of the cards correctly and displays this value on standard output.

There are several constraints on this assignment.

- No global variables
- Use the C/C++ random number generator (`random`) to generate the three card values.
- Display the cards dealt (by the random number generator) to standard output.
- You must define and use the function `int blackjackSum(int card1, int card2, int card3);` to compute the total value of the cards. This function is passed the three card values generated by the random number generator and returns the total value of the cards computed according to Blackjack's rules.

Make sure you handle the cases when the player is dealt two or three aces.

Here's sample output:

```
sushil@flash as3/code > ./bjSimple
Cards dealt: 3, 1, 9
Hand value is: 13
sushil@flash as3/code > ./bjSimple
Cards dealt: 4, 10, 8
Hand value is: 22
```

```
sushil@flash as3/code > ./bjSimple
Cards dealt: 6, 1, 1
Hand value is: 18
```

2 Triangles (10 points)

Design, implement, and test a C/C++ program that, given the length of the sides, identifies the triangle's type. You will read the three lengths (all triangles have three sides) from an input file called "triangle.data" The information in this file is not ordered.

Your program must

- Figure out whether the triangle is equilateral, isosceles, or scalene
- Figure out whether the triangle has a right angle, an obtuse angle, or is an acute angled triangle. If the triangle is equilateral or isosceles, it is an acute angled triangle. If it is scalene, then the following relationship holds for right angled scalene triangles:

$$h^2 = s_1^2 + s_2^2$$

Where h is the longest side and s_1 and s_2 are the other two sides. So, obtuse scalene triangles must have:

$$h^2 > s_1^2 + s_2^2$$

and acute scalene triangles must have.

$$h^2 < s_1^2 + s_2^2$$

You will then write the triangle's type information (both items above) to an output file named "triangleType.txt" and to standard output.

There are several constraints on this assignment.

- No global variables
- You must define and use the following functions
 - `int triType(int side1, int side2, int side3);` that takes the three lengths and returns whether the triangle is equilateral (1), isosceles (2), or scalene (3).
 - `int longestSide(int side1, int side2, int side3);` which takes the three side lengths and returns the length of the longest side.
 - `int angleType(int side1, int side2, int side3);` which takes the three side lengths and returns whether the triangle's angle type is acute (1), right(2), or obtuse(3).
- You must check for any errors when opening the input file for reading and when opening the output file for writing.

Suppose your input file: "triangle.data" contained:

3 4 5

Then when you run your program you should get the following output on standard output.

```
Triangle sides: 3, 4, 5  
Triangle type: Scalene  
Triangle angle type: Right
```

and the file "triangleType.txt" should contain

```
Triangle sides: 3, 4, 5  
Triangle type: Scalene  
Triangle angle type: Right
```

3 Handing it in

Turn in a Folder (Binder) containing:

1. Cover sheet with
 - (a) Assignment Number
 - (b) Section Number
 - (c) Your name
 - (d) Your TA's name
2. Source code (.cpp) file (s) on a CD or Floppy with your name and section number written on your disk.
3. Hardcopy of your source (printout)
4. Printout of sample runs of your program on the test cases listed above.