CS 1410 Programming Assignment #2
If Statements

Note that this assignment contains two programs.

Program 1

Copy roots.cpp from Lab #1 to roots2.cpp and modify the program as follows:
1. The quadratic formula fails if coefficient \(a\) is 0. If the user enters a 0 for \(a\), print an appropriate error message and end the program (skip all calculations, do not loop). You may use `exit(0);` if you want. If you do, #include `<stdlib.h>`
2. The discriminant, \(b^2 - 4ac\), describes the roots of a quadratic equation: if it is 0, there are two equal real roots; if it is greater than 0, there are two real roots; and if it is less than 0, there are two complex roots. Complex numbers are displayed with a real part and an imaginary part: e.g., \(6 + 3i\) and \(6 - 3i\)
   a. The real part (6 in the example) is calculated by the formula \(-b/2a\)
   b. The imaginary part (3i in the example) is calculated by the formula \(\sqrt{\text{discriminant}}/2a\) with an ‘i’ character printed at the end
   c. Taking the square root of a negative number with the sqrt function causes a runtime error, so the discriminant must be negated (i.e., multiplied by -1) before taking the square root
3. Your program should find the roots in all three situations (the lab 1 version should work when the discriminant = 0 or > 0); your modifications for lab 2 should handle the case when the discriminant is < 0.
4. Follow the link in the lab 2 notes to see some formulas and discussion that might be useful.

Test Case:
\[a = 2, b = 2, c = 2; x1 = -0.5 + 0.866025i, x2 = -0.5 - 0.866025i\]
The test cases from lab 1 should still work as before
Program #2

Create a number guessing game program named `guess.cpp` (10 points) that generates a pseudo random number in the range 0 to 99. (A pseudo random number is a number that “looks” random – i.e., passes certain statistical tests for randomness – but which is generated deterministically or non-randomly.) The following pseudo code describes the program’s behavior:

1. seed the random number generator
2. generate a target number between 0 and 99
3. loop
   a. prompt the user to enter a guess between 0 and 99
   b. read the guess
   c. if the guess and the target number are equal
      i. print a success message (e.g., “Right” or “You Win”)
      ii. break out of the loop
   d. if the guess is less than 0, break out of the loop (a way to quit early)
   e. if the guess is less than the target number, print “Low”
   f. if the guess is greater than the target number, print “High”

Generating Pseudo Random Numbers

The standard C/C++ library provides a function named `rand` that will generate pseudo random numbers, one per call, in the range of 0 to RAND_MAX (some big number). The sequence of numbers produced by a pseudo random number generator eventually repeats. A seed number determines where in the cycle the generator starts. A second function named `srand` seeds or initializes the random number generator. It is common to use the current system time, a value that is constantly changing, to seed the generator. The following code fragment demonstrates how to seed the generator and how to generate a pseudo random number between 0 and 99:

```cpp
#include <iostream>
#include <stdlib.h>
#include <time.h>
using namespace std;

int main()
{
    srand((unsigned)time(NULL)); // seed the generator
    int target = rand() % 100; // numbers [0..99]
    .
    .
    .
}
```

Program Submission and Grading

Upload `roots2.cpp` and `guess.cpp` to Blackboard. Do not zip the files. Make sure that your program files are named correctly. Do not execute a pause or “dummy” read at the end of the programs.