

Lab 12: Random numbers, nested if-else statements, and switch statements**Due:** 10/26/22

This assignment consists of two parts in which you will solve the same problem but using two different control structures.

In this part of the lab you will practice decision making using if statements. First you will review a program that demonstrates the use of if statements, then you will write your own C++ code to practice this skill.

Example Program

Our example program simulates a drive-thru restaurant. The user selects the item she wishes to order, and the program prints the amount of money she owes.

The **LargeJoes.cpp** program provided illustrates the skills you are learning in this lab. Open it in your IDE, read and understand the code and finally run the program with different inputs to see how it works.

Once you understood this program start working on the following problem.

Problem: Suppose you are asked to simulate a Magic 8-Ball. The Magic 8-Ball is a fortune telling toy. The player asks a question and the Magic 8-Ball randomly answers the question with one of 20 possible answers. A description of the game, along with the twenty answers can be found on this [Wikipedia page](#). The Wikipedia page includes a link to an online version of the game.

For this program you only need to use the following 5 answers according to the following table:

Random Number	Answer
0	Yes
1	Maybe
2	No
3	Ask again later
None of the above	I don't know

The answers will be generated at random using the following expressions:

- 1) `srand(static_cast<int>(time(NULL)))` to seed the random number generator
- 2) `(rand() % (highRange - lowRange + 1)) + lowRange` to generate the random number

Your task: implement in C++ the algorithm solution shown below.

Algorithm solution (in pseudocode):

To solve this problem your program must perform the following tasks:

Part A: Solves the problem using **multi-branch if-else statements**

Declare the function prototypes for the functions above main()

Declare a variable named **randomNumber** that holds **whole** numbers

Declare a variable named **lowRange** that holds **whole** numbers and initializes it to 0

Declare a variable named **highRange** that holds **whole** numbers and initializes it to 4

Seed the random number generator using expression 1) shown above

Prompt the user to enter a question

Ignore the user input

Call function **randNumGen()** to generate a random number and assign it to **randomNumber**

Display title "Part A solution"

Display the message shown below

"Answer: ", call function **fortuneTellerA()** to get the answer

1) You **must** define function **randNumGen()** according to the following specifications:

- I. The function receives two whole numbers: the first one is the upper boundary and the second one is the lower boundary used to generate a random number.
- II. Returns the random number generated using expression 2)

2) You **must** define function **fortuneTellerA()** that uses **multi-branch if-else statements** to determine the answer to be returned based on the number received.

This function receives the random number (whole number) and returns the corresponding answer based on the table shown above.

Important: in this solution make your function directly return the answer in each branch of the multi-branch if-else statements.

Part B: Solves the problem using **switch statements**

Once you finished part A go over the example on the use of a **switch** statement to write a program that simulates a drive-thru restaurant.

The **LargeJoes_switch.cpp** program provided illustrates the skills you are learning in this lab. Add it to a project in Visual C++, read and understand the code and finally run the program with different inputs to see how it works.

Compare how a switch statement is used to solve this problem with how we used if statements in the example for part A to do the same. When the condition for our decision is simply an integer or character, switch statements provide a clearer and quicker way to write the program.

Once you understood this program solve the Magic 8-Ball problem using a **switch** statement.

3) You **must** define function **fortuneTellerB()** that uses a **switch statement** to determine the answer to be returned based on the number received.

This function receives the random number (whole number) and returns the corresponding answer based on the table shown above.

Important: in this solution declare a local variable that holds text and assign the corresponding answer in each case of the switch statement. Upon exiting the switch return the value in the local variable.

Once you have defined this function add the steps shown below at the bottom of your program and implement them.

Display title "Part B solution"

Display the message shown below

"Answer: ", call function **fortuneTellerB()** to get the answer

The program must compile without errors or warnings.

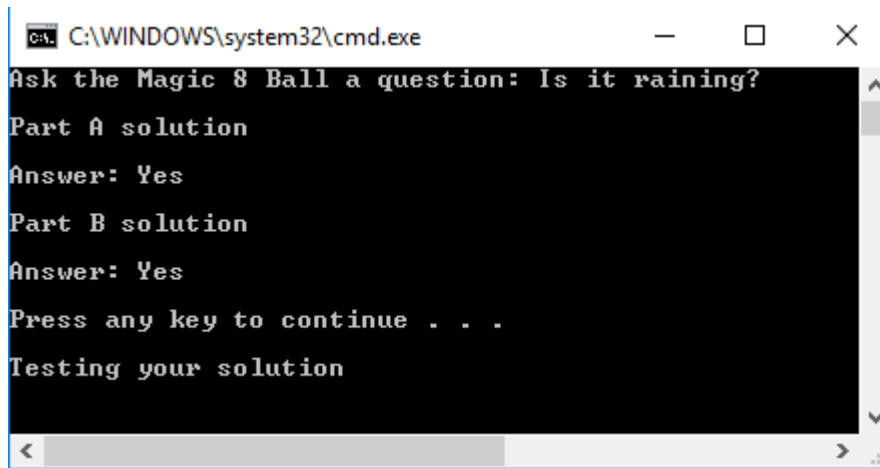
Open **lab12.cpp** in your IDE and implement the above algorithm (already provided in the source code as comments).

Note:

- Do NOT remove or modify the statements that I use to test certain things in your program.
- Run my sample solution to know how your program must behave. Pay attention to the input and the output formats. Your solution must behave exactly like mine.

<https://replit.com/@GDietrich/1370-lab12sample>

- Carefully analyze the following figure and use it as a reference to ensure you do the right things.



```
C:\WINDOWS\system32\cmd.exe
Ask the Magic 8 Ball a question: Is it raining?
Part A solution
Answer: Yes
Part B solution
Answer: Yes
Press any key to continue . . .
Testing your solution
```

- I am posting the executable of my solution for your reference. Test and compare your solution with mine for different inputs. Create your own **test plans** to test your program with the necessary inputs to ensure that **all** possible answers are generated.
- Your program must pass all my tests.
- You must define the most appropriate type of function, the parameter list (using the most appropriate parameters), and the body of the function to be implemented.

If you have concerns or specific questions, post them on the Discussion Board of Blackboard.

Don't forget to include at the top of the program the comments shown below with your information (name, class and section number, etc.)

```
////////////////////////////////////  
//  
// Name: <Put your name here>  
// Date: <Today's date>  
// Class: <Your class number and section number, like: CSCI 1370.02>  
// Semester: <This semester, like: Spring 2012>  
// CSCI/CMPE 1370 Instructor: <Your lecture instructor's name>  
//  
// Program Description: Enter here your description of what the program does  
//  
////////////////////////////////////
```

When done, submit your solution through Blackboard using the “Assignments” tool. Do Not email it.

Paste the [link](#) to your solution and the [source code](#) in the textbox corresponding to Text Submission (click on the [Write Submission button](#)) before you click on Submit.

The following is the basic criteria to be used to grade this part of the assignment:

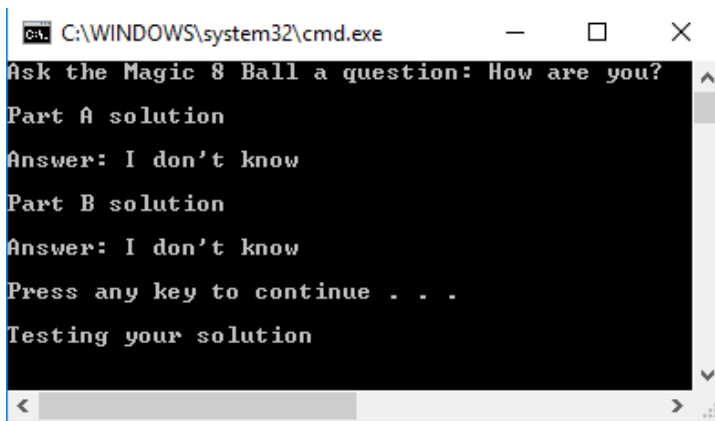
You start with 100 points and then lose points as you don't do something that is required.

- 1) -5: wrong identifiers
- 2) -5: wrong variable types
- 3) -5: no comments or too few comments in source code
- 4) -20: missing or wrong implementation of function **randNumGen()**
- 5) -5: (Part A) doesn't properly determine the answer corresponding to the number received (each random number)
- 6) -5: (Part A) doesn't have a default case
- 7) -10: (Part A) doesn't directly return the answer in each branch of the multi-branch if-else statements
- 8) -40: (Part A) missing or wrong implementation (doesn't use **multi-branch if-else statements**) of function **fortuneTellerA()**
- 9) -5: (Part B) doesn't properly determine the answer corresponding to the number received (each random number)
- 10) -5: (Part B) doesn't have a default case
- 11) -10: (Part B) doesn't use a local variable to hold the corresponding answer for each case in the switch statement
- 12) -40: (Part B) missing or wrong implementation (doesn't use **switch statements**) of function **fortuneTellerB()**
- 13) -10: incorrect type of the function (each)
- 14) -10: incorrect function call (each)
- 15) -10: incorrect type of parameters (value or reference)
- 16) -5: incorrect input format
- 17) -5: incorrect output format
- 18) -50: program doesn't compile

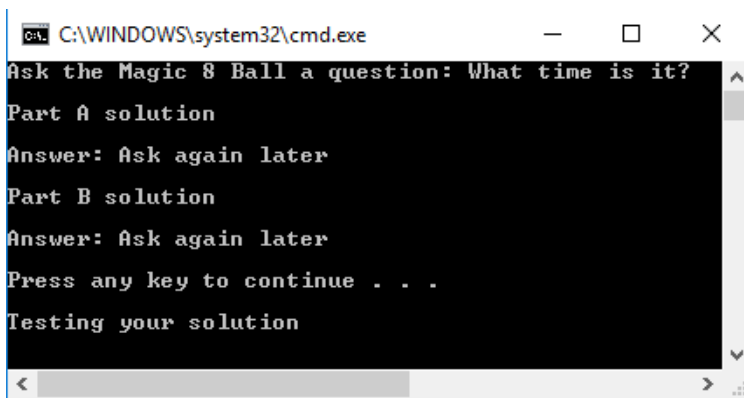
- 19) -20: does not pass all tests
- 20) -20: program does not implement the provided algorithm
- 21) -20: Incorrect/missing source code
- 22) -20: Incorrect/missing link to your Repl.it solution
- 23) -100: The code submitted is not your creation (you got it from a web site or another person)
- 24) -10: Late

Important: more points may be lost for other reasons not specified here.

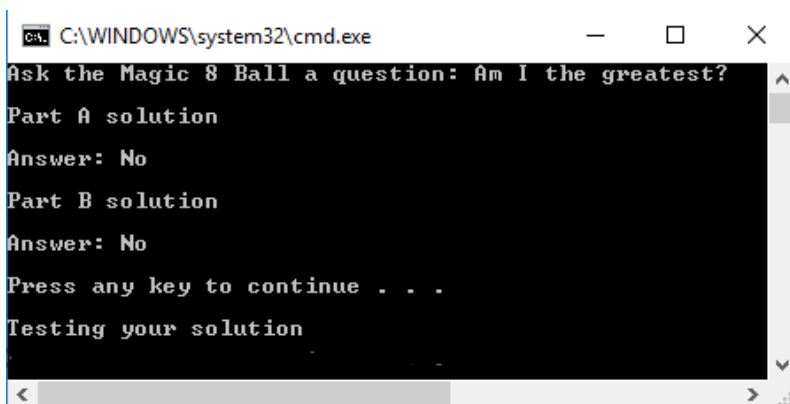
Sample runs of my program:



```
C:\WINDOWS\system32\cmd.exe
Ask the Magic 8 Ball a question: How are you?
Part A solution
Answer: I don't know
Part B solution
Answer: I don't know
Press any key to continue . . .
Testing your solution
```



```
C:\WINDOWS\system32\cmd.exe
Ask the Magic 8 Ball a question: What time is it?
Part A solution
Answer: Ask again later
Part B solution
Answer: Ask again later
Press any key to continue . . .
Testing your solution
```



```
C:\WINDOWS\system32\cmd.exe
Ask the Magic 8 Ball a question: Am I the greatest?
Part A solution
Answer: No
Part B solution
Answer: No
Press any key to continue . . .
Testing your solution
```