San José State University
Department of Computer Engineering

CMPE 180A
Data Structures and Algorithms in C++

Spring 2020
Instructor: Ron Mak

Assignment #2

Assigned:	Tuesday, February 4
Due:	Tuesday, February 11 at 5:30 PM
CodeCheck:	http://codecheck.it/files/2002040935f3w5v63ro9tfinzo8y4kyd4nw **Note:** Use Firefox or Chrome, not Safari.
Canvas:	Assignment 2: Monty Hall Problem
Points:	100

Monty Hall Problem
This assignment will give you practice decomposing a problem into smaller subproblems whose solutions you implement with functions. Submit a C++ program that simulates the Monty Hall Puzzle to demonstrate empirically whether it's better to stay with an original door choice or to switch doors (or it doesn't matter).

Choose the right door and win the new car!
Monty Hall was a popular game show host on American television. In one segment of his show, you are a contestant from the studio audience. He shows you three doors, Door #1, Door #2, and Door #3. Hidden behind one door is a brand new car, and behind each of the other two doors is a goat.

Monty asks you to pick a door. Of course, you want to pick the right door and win the car.

Monty knows which door hides the car. After you've picked a door, he opens one of the other doors and shows you a goat.

Monty then offers you a chance to stay with the door you originally picked, or you can switch your choice to the remaining third door. Would staying or switching be better?

The simulation
Write a C++ program to simulate the above scenario. The program should:

1. Randomly pick a door to hide the car.
2. Acting as you, the contestant, randomly pick a door as your first door choice.
3. Acting as Monty, open a door to reveal a goat. Since Monty (i.e., the program) knows which door hides the car:
    a. As the contestant, if your first door choice was correct, the other two doors each hides a goat. Monty randomly chooses one of the two doors to open.
    b. If your first door choice was incorrect (your door hides a goat), Monty opens the other door that hides a goat.
4. This leaves a third door. You, the contestant, don't know whether it hides the car or a goat. Should you stay with your first door as your original choice, or should you switch to the third door as your second door choice? Your program should keep track of whether you win by staying with your first door choice or you win if you switch to your second door choice.

Run the simulation 100 times. How many times do you win by staying with your first door choice vs. switching to a second door choice? Does it make a difference? What is the ratio of second door choice wins to first door choice wins?

Stay or switch?
Many people, including some prominent mathematicians, intuitively believed that it shouldn't make any difference whether to stay with your first door choice or to switch to a second choice. Either way, the probability of winning the car is one third. Or is it?

A statistician can explain the results using conditional probabilities. See https://en.wikipedia.org/wiki/Monty_Hall_problem. But this simulation will demonstrate empirically which strategy is better.

Random number generation
As described in the above scenario, each simulation involves up to three random numbers. Use the predefined **`srand`** function to seed the random number generator at the start, then subsequently use the predefined **`rand`** function to generate the next random number. The generated pseudo-random numbers will be nonnegative integers.

Since the random numbers represent door numbers, you will need to modify the generated random numbers to be either 1, 2, or 3.

`srand`: http://www.cplusplus.com/reference/cstdlib/srand/
`rand`: http://www.cplusplus.com/reference/cstdlib/rand/

Sample output
Due to the random nature of the output, **ignore CodeCheck's comparison** of your program's output to the master. Here's sample output with 10 simulations (your program should do 100 simulations):

```
   #     Car   First  Opened  Second    Win    Win
        here  choice    door  choice  first second

   1       3       1       2       3           yes
   2       2       1       3       2           yes
   3       2       2       1       3    yes
   4       2       2       3       1    yes
   5       1       1       3       2    yes
   6       3       2       1       3           yes
   7       2       2       1       3    yes
   8       1       3       2       1           yes
   9       2       3       1       2           yes
  10       1       3       2       1           yes

   4 wins if stay with the first choice
   6 wins if switch to the second choice
Win ratio of switch over stay: 1.5
```

For each simulation, your output should show:

- which door hides the car (randomly generated)
- which door is your first choice (randomly generated)
- which door Monty opens (possibly random)
- which door is your second choice (the remaining unopened third door)
- whether the first or second choice wins

Your program structure
Go to the above CodeCheck URL for a partially written program MontyHall.cpp. Your program must be decomposed into functions. The provided function declarations are suggestions only. You are not obligated to use these declarations, and you can add other functions. Put your function declarations before the main, and the function definitions after the main.

Submission into Canvas
When you're satisfied with your program in CodeCheck, click the "Download" link at the very bottom of the Report screen to download a signed zip file of your solution. Submit this zip file into Canvas. You can submit as many times as you want until the deadline, and the number of submissions will not affect your score. Only your last submission will be graded.

Due to the random nature of the output, **ignore CodeCheck's comparison** of your program's output to the master.

Submit the signed zip file from CodeCheck into Canvas:
Assignment #2. Monty Hall Problem.

Note: You must submit the signed zip file that you download from CodeCheck, or your submission will not be graded. Do not rename the zip file.

Rubric
Your program will be graded according to these criteria:

Criteria	Max points
Good output • Correct output values. • Correct output format.	**30** • 20 • 10
Good program design • Good functional decomposition. • Good choice of function names. • Good use of function arguments. • Good comments that describe what each function does, what its parameters are, and what it returns.	**50** • 20 • 10 • 10 • 10
Good program style • Descriptive variable names. • Meaningful comments inside functions.	**20** • 10 • 10

Academic integrity

You may study together and discuss the assignments, but what you turn in must be your individual work. Assignment submissions will be checked for plagiarism using Moss (http://theory.stanford.edu/~aiken/moss/). **Copying another student's program or sharing your program is a violation of academic integrity.** Moss is not fooled by renaming variables, reformatting source code, or re-ordering functions.

Violators of academic integrity will suffer severe sanctions, including academic probation. Students who are on academic probation are not eligible for work as instructional assistants in the university or for internships at local companies.