

## **Title: Uncertainty Problems(Monty hall problem)**

**EX. NO : 06**

**Name: Parth Langalia**

**DATE : 6-03-2023**

**Reg No. : RA2011033010033**

### **AIM :**

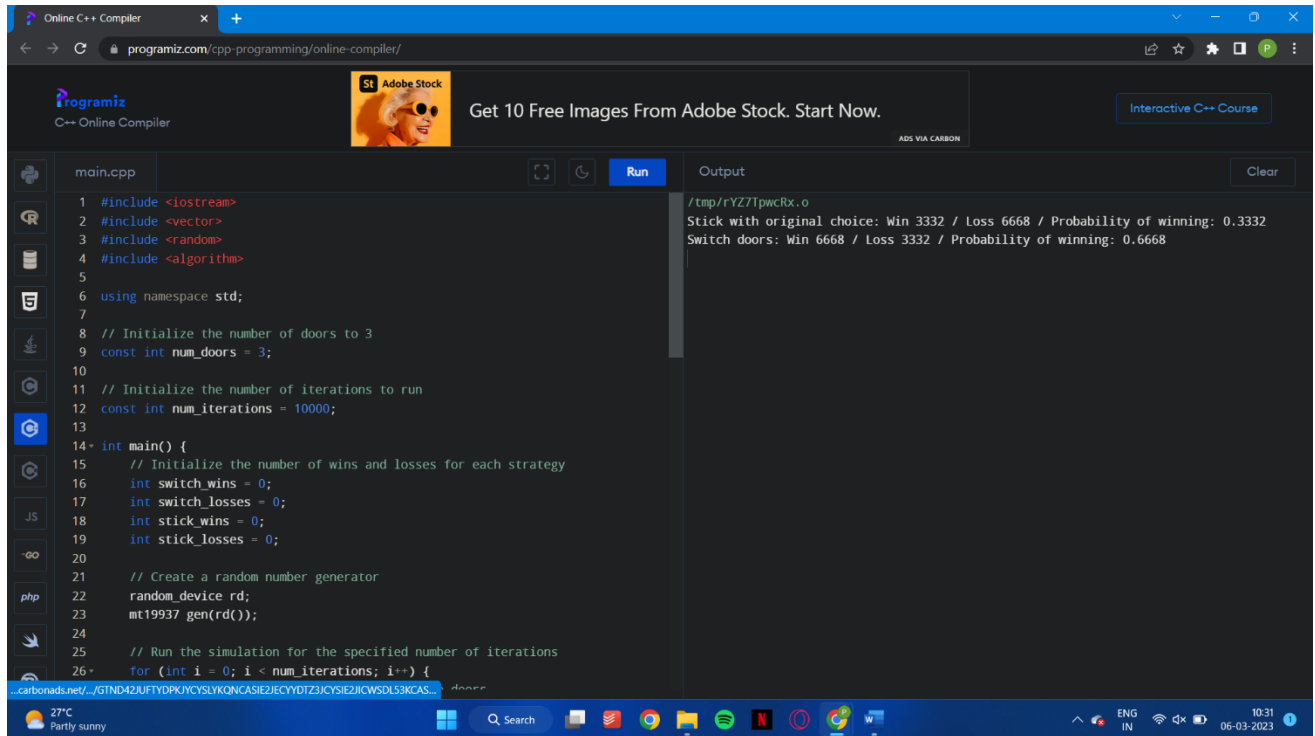
To implement Bayesian Belief Networks to model the problem of Monty Hall.

### **PSEUDO CODE :**

1. Initialize the number of doors to 3.
2. Randomly assign a prize to one of the doors, and assign nothing to the other two doors.
3. Have the player select one of the doors.
4. Have the game show host select one of the doors that the player didn't choose, and that doesn't have the prize behind it. If there is more than one such door, randomly select one of them.
5. Ask the player if they want to switch their choice to the remaining unopened door, or if they want to stick with their original choice.
6. If the player decides to switch, have them select the remaining unopened door.
7. Reveal what's behind the door that the player has selected, whether they switched or stuck with their original choice.
8. If the player selected the door with the prize, report that they have won. If they did not select the door with the prize, report that they have lost.

### **PROGRAM(with OUTPUT) :**

## Monty Hall Problem:



The screenshot shows a web browser window with the URL `programiz.com/cpp-programming/online-compiler/`. The page features a dark-themed code editor with a file named `main.cpp`. The code implements a simulation of the Monty Hall problem, running 10,000 iterations. It compares two strategies: 'stick' (staying with the original choice) and 'switch' (switching to the other door). The results are displayed in the 'Output' panel on the right.

```
1 #include <iostream>
2 #include <vector>
3 #include <random>
4 #include <algorithm>
5
6 using namespace std;
7
8 // Initialize the number of doors to 3
9 const int num_doors = 3;
10
11 // Initialize the number of iterations to run
12 const int num_iterations = 10000;
13
14 int main() {
15     // Initialize the number of wins and losses for each strategy
16     int switch_wins = 0;
17     int switch_losses = 0;
18     int stick_wins = 0;
19     int stick_losses = 0;
20
21     // Create a random number generator
22     random_device rd;
23     mt19937 gen(rd());
24
25     // Run the simulation for the specified number of iterations
26     for (int i = 0; i < num_iterations; i++) {
```

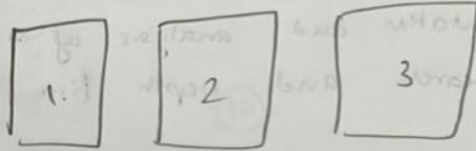
The output panel shows the following results:

```
/tmp/rYZ71pwcRx.o
Stick with original choice: Win 3332 / Loss 6668 / Probability of winning: 0.3332
Switch doors: Win 6668 / Loss 3332 / Probability of winning: 0.6668
```

The Windows taskbar at the bottom shows the date and time as 06-03-2023, 10:31.

## Manual Calculations :

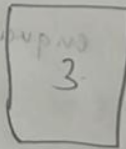
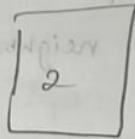
Manual Calculation / Input-output :



Player chooses : 2

∴ Game host eliminates 1/3:

∴ The door 1 will be eliminated.



⇒ now host asks the player to change or keep choice.

If gift is behind 2, and the player chooses door 2, he wins if

gift is behind door 2.  
probability is  $\frac{1}{3}$  or 33%

and will not

$\frac{1}{2}$  or 50%.

**RESULT :**

The uncertainty problem(Monty hall problem) has been successfully implemented and solved.