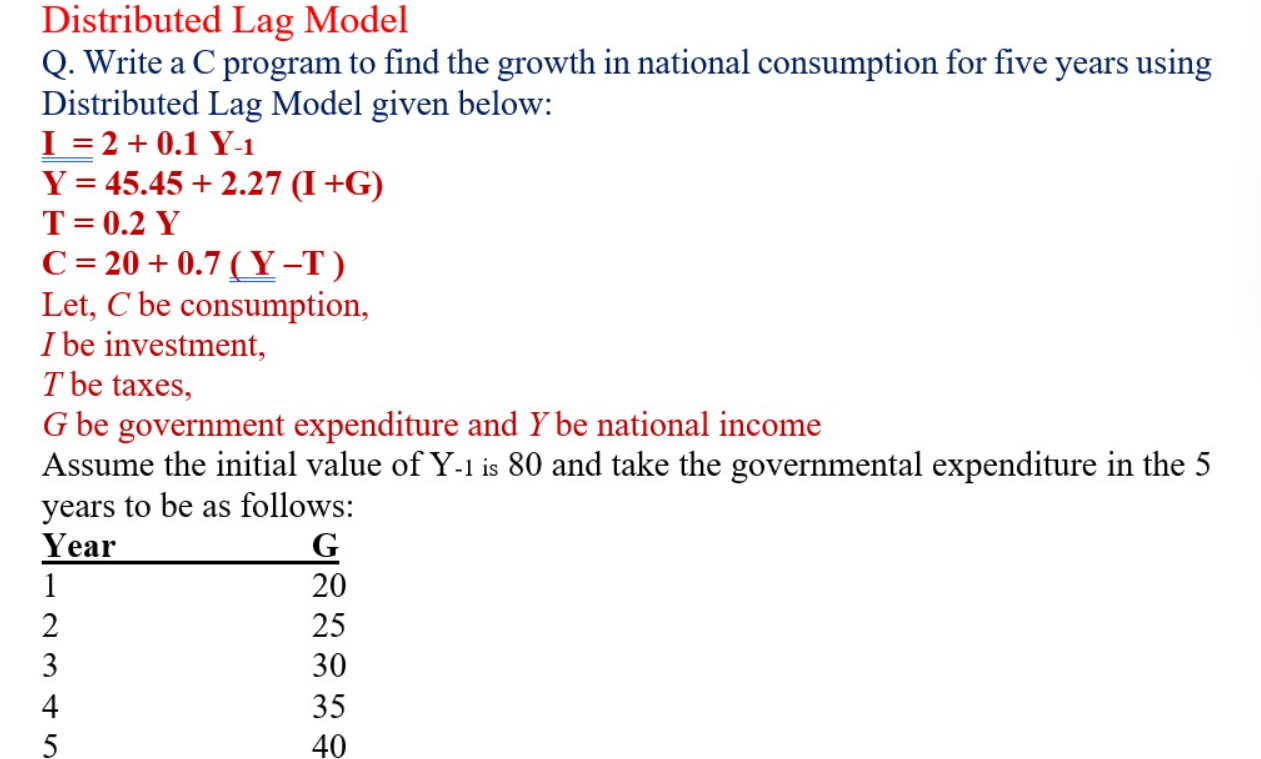
**Lab 1**



**Code:**

#include <stdio.h>

int main() {

// Given Data

double Y\_prev = 80; // Initial value of Y-1

double G[5] = {20, 25, 30, 35, 40}; // Governmental expenditure for 5 years

double I, Y, T, C;

printf("Year\tInvestment (I)\tIncome (Y)\tTaxes (T)\tConsumption (C)\n");

for (int year = 0; year < 5; year++) {

I = 2 + 0.1 \* Y\_prev; // Investment equation

Y = 45.45 + 2.27 \* (I + G[year]); // National income equation

T = 0.2 \* Y; // Tax equation

C = 20 + 0.7 \* (Y - T); // Consumption equation

printf("%d\t%.2f\t\t%.2f\t\t%.2f\t\t%.2f\n", year + 1, I, Y, T, C);

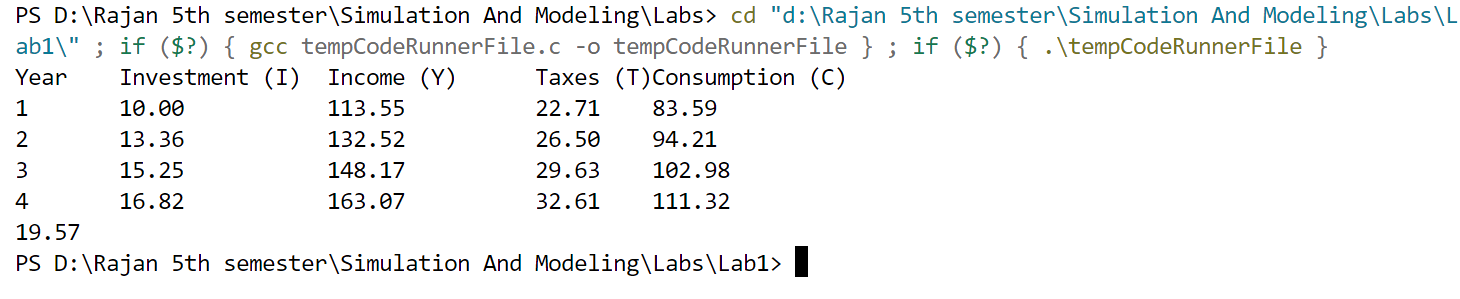
Y\_prev = Y; // Update Y-1 for next year

}

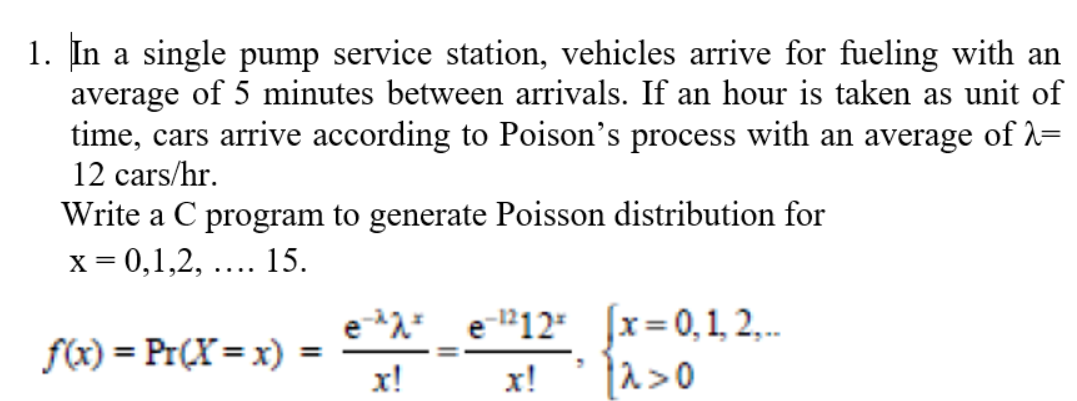
return 0;

}

**Output:**

****

**Lab 2**

****

**Code:**

#include<stdio.h>

#include <math.h>

#include <stdlib.h>

double poisson(int x, double lambda) {

return exp(-lambda) \* pow(lambda, x) / tgamma(x + 1);

}

int main() {

double lambda = 12.0; // average arrival rate per hour

printf("Poisson Distribution for x = 0 to 15:\n");

for (int x = 0; x<= 15; x++) {

double probability = poisson(x, lambda);

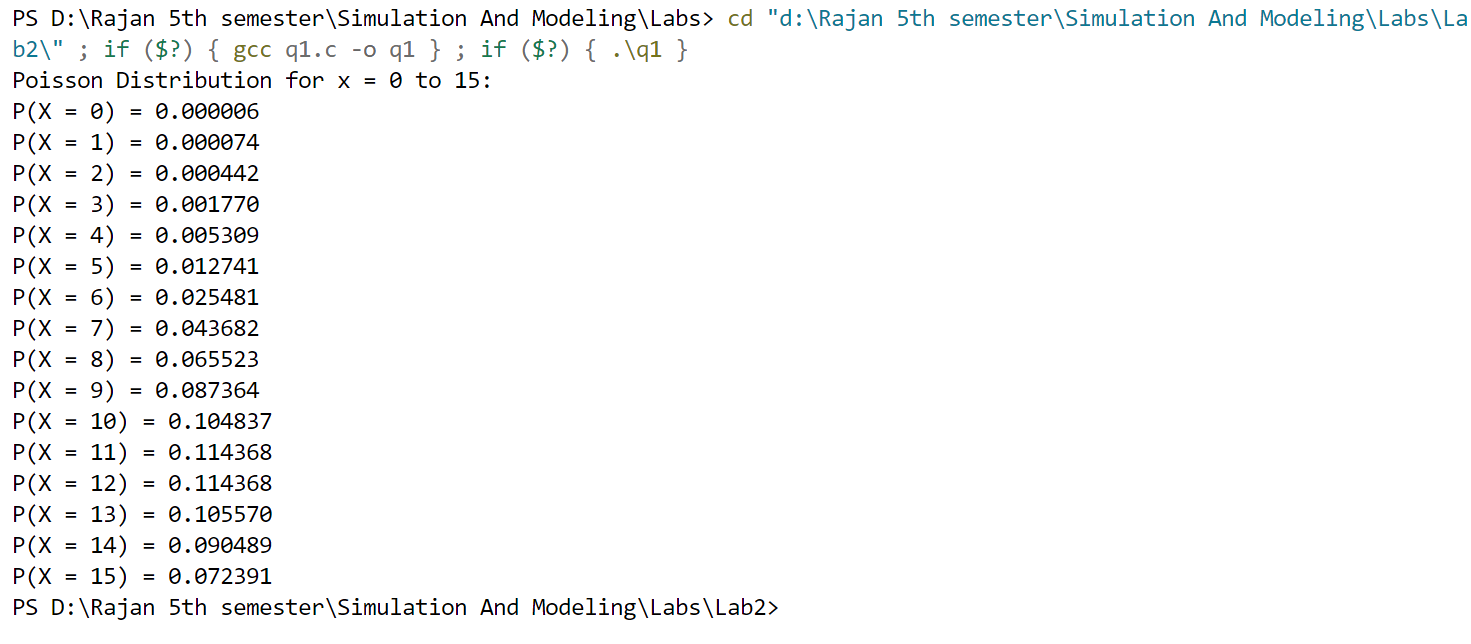
printf("P(X = %d) = %f\n", x, probability);

}

return 0;

}

**Output:**

****

**Lab 3**

**1. Write a program to estimate the value of PI using Monte Carlo Simulation.**

**Code:**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int main() {

int total\_points, points\_inside\_circle = 0;

double x, y, pi\_estimate;

// Ask for the total number of points

printf("Enter the total number of points: ");

scanf("%d", &total\_points);

// Validate input

if (total\_points <= 0) {

printf("Total points must be a positive integer.\n");

return 1;

}

// Seed the random number generator

srand(time(NULL));

for (int i = 0; i < total\_points; i++) {

// Generate random points in the range [0, 1]

x = (double)rand() / RAND\_MAX;

y = (double)rand() / RAND\_MAX;

// Check if the point is inside the unit circle

if (x \* x + y \* y <= 1.0) {

points\_inside\_circle++;

}

}

// Calculate the estimated value of PI

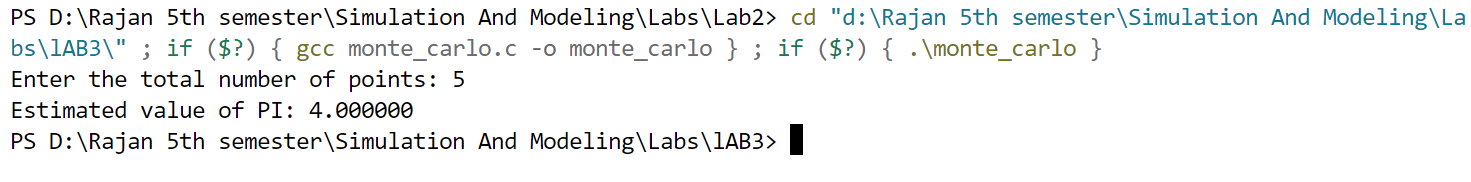
pi\_estimate = 4.0 \* points\_inside\_circle / total\_points;

printf("Estimated value of PI: %f\n", pi\_estimate);

return 0;

}

**Output:**

****

**2. Write a program to estimate the area under the curve using the Monte Carlo Simulation.**

**Code:**

#include <iostream>

#include <cmath>

#include <cstdlib>

#include <ctime>

using namespace std;

// Function to be integrated

double f(double x) {

return sin(x);

}

int main() {

double a, b;

int numIterations;

// Get user input

cout << "Enter the lower limit of the interval: ";

cin >> a;

cout << "Enter the upper limit of the interval: ";

cin >> b;

cout << "Enter the number of iterations: ";

cin >> numIterations;

// Validate input

if (a >= b) {

cout << "Error: Lower limit must be less than the upper limit." << endl;

return 1;

}

if (numIterations <= 0) {

cout << "Error: Number of iterations must be a positive integer." << endl;

return 1;

}

// Seed the random number generator

srand(time(NULL));

int numPointsInside = 0;

for (int i = 0; i < numIterations; i++) {

double x = (double) rand() / RAND\_MAX \* (b - a) + a;

double y = (double) rand() / RAND\_MAX; // sin(x) is at most 1, so max y = 1

if (y <= f(x)) {

numPointsInside++;

}

}

// Compute estimated area

double totalArea = (b - a) \* 1.0;

double areaUnderCurve = totalArea \* (double) numPointsInside / numIterations;

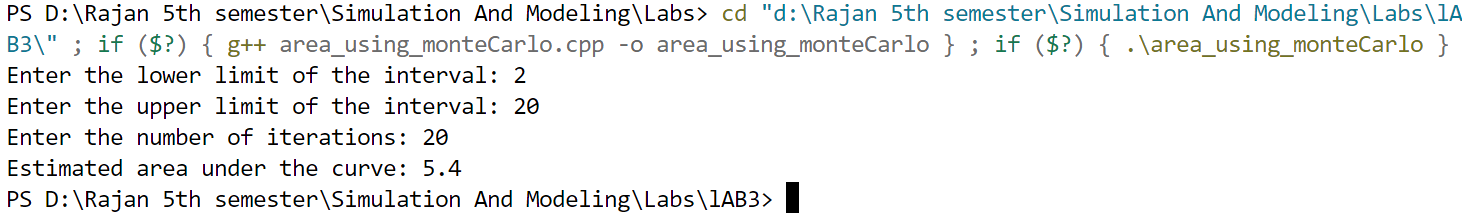
// Output the result

cout << "Estimated area under the curve: " << areaUnderCurve << endl;

return 0;

}

**Output:**

****

**Lab 4**

**1.Customers arrive in a bank according to a Poisson's process with mean inter arrival time of 10 minutes. Customers spend an average of 5 minutes on the single available counter, and leave.**

**Write a program in C to find:**

**a Probability that a customer will not have to wait at the counter.**

**b.Expected number of customers in the bank.**

**c.Time a customer expects to spend in the bank.**

**Code:**

#include <stdio.h>

#include <math.h>

int main() {

double lambda = 1.0 / 10.0; // Mean inter-arrival rate (customers per minute)

double mu = 1.0 / 5.0; // Service rate (customers per minute)

// Probability that a customer will not have to wait at the counter (P0)

double rho = lambda / mu;

double P0 = 1.0 - rho;

// Expected number of customers in the bank (L)

if (mu - lambda <= 0) {

printf("Error: The service rate must be greater than the arrival rate to avoid system overload.\n");

return 1;

}

double L = lambda / (mu - lambda);

// Average time a customer can expect to spend in the bank (W)

double W = 1.0 / (mu - lambda);

// Print results

printf("Probability that a customer will not have to wait: %lf\n", P0);

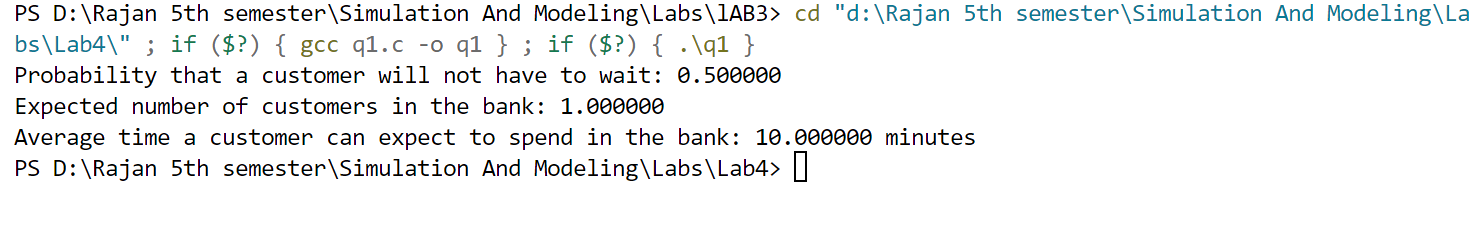
printf("Expected number of customers in the bank: %lf\n", L);

printf("Average time a customer can expect to spend in the bank: %lf minutes\n", W);

return 0;

}

**Output:**

****

**2. At the ticket counter of football stadium, people come in queue and purchase tickets. Arrival rate of customers is 1/min. It takes at the average 20 seconds to purchase the ticket.**

**WAP in C to calculate total time spent by a sports fan to be seated in his seat, if it takes 1.5 minutes to reach the correct seat after purchasing the ticket.**

**If a fan comes exactly before 2 minutes before the game starts, can sports fan expect to be seated for the kick-off?**

**Code:**

#include <stdio.h>

int main() {

// Given data

double lambda = 1.0; // Arrival rate (customers per minute)

double mu = 3.0; // Service rate (customers per minute)

double service\_time = 1.0 / mu; // 20 seconds = 1/3 minutes

double walking\_time = 1.5; // Time to reach the seat (in minutes)

// Traffic intensity (ρ)

double rho = lambda / mu;

// Average waiting time in queue (Wq)

double Wq = rho / (mu - lambda);

// Average time spent in the system (Ws)

double Ws = Wq + service\_time;

// Total time to be seated

double total\_time = Ws + walking\_time;

// Display results

printf("Average waiting time in queue: %.2f minutes\n", Wq);

printf("Total time spent in the system: %.2f minutes\n", Ws);

printf("Total time to be seated: %.2f minutes\n", total\_time);

// Checking if the fan can be seated before kick-off

double time\_before\_game = 2.0;

if (total\_time <= time\_before\_game) {

printf("Yes, the sports fan can expect to be seated for the kick-off!\n");

} else {

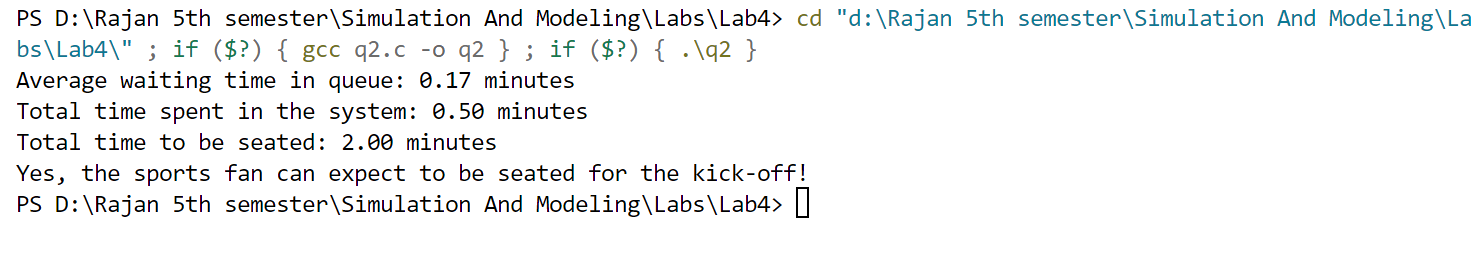
printf("No, the sports fan will likely miss the kick-off.\n");

}

return 0;

}

**Output:**

****

**3.Write a program to calculate measures of a M/M/1 Queue for a given value of Arrival Rate and Service Rate.**

**Code:**

#include <iostream>

#include <cmath>

using namespace std;

// Function to calculate M/M/1 queue metrics

void mm1\_queue(int lambda, int mu, int n) {

if (lambda >= mu) {

cout << "Error: The arrival rate must be less than the service rate to avoid system overload." << endl;

return;

}

double rho = lambda / (double)mu; // Server utilization

double Lq = pow(rho, 2) / (1 - rho); // Avg no. of customers in the queue

double Wq = Lq / (double)lambda; // Avg time a customer spends in the queue

double L = lambda / (double)(mu - lambda); // Avg no. of customers in the system

double W = 1.0 / (double)(mu - lambda); // Avg time a customer spends in the system

double P0 = 1 - rho; // Probability of zero jobs in the system

double Pn = P0 \* pow(rho, n); // Probability that 'n' customers are in the system

// Output the results

cout << "Rho (Traffic Intensity) = " << rho << endl;

cout << "Expected Number of Jobs in the Queue (Lq) = " << Lq << endl;

cout << "Expected Time in the Queue (Wq) = " << Wq << " minutes" << endl;

cout << "Expected Number of Jobs in the System (L) = " << L << endl;

cout << "Expected Time in the System (W) = " << W << " minutes" << endl;

cout << "Probability of Zero Jobs in the System (P0) = " << P0 << endl;

cout << "Probability of " << n << " Jobs in the System (Pn) = " << Pn << endl;

cout << "Utilization = " << rho \* 100 << "%\n";

}

int main() {

int lambda, mu, n;

// User input

cout << "Enter the Arrival Rate (customers per minute): ";

cin >> lambda;

cout << "Enter the Service Rate (customers per minute): ";

cin >> mu;

cout << "Enter the number of jobs in the system (n): ";

cin >> n;

cout << endl;

mm1\_queue(lambda, mu, n);

return 0;

}

**Output:**

**Lab 5**

**1. WAP to solve the below Weather problem**

**Rainy today => 40% Rainy tomorrow**

**=> 60% not Rainy tomorrow**

**Not rainy today => 20% Rainy tomorrow**

**=> 80% not Rainy tomorrow**

**What will be probability if todays is not raining then not rain the day after tomorrow?**

**Code:**

#include <stdio.h>

int main() {

// Given probabilities

double P\_N\_given\_N = 0.8; // Probability of no rain tomorrow if no rain today

double P\_R\_given\_N = 0.2; // Probability of rain tomorrow if no rain today

double P\_N\_given\_R = 0.6; // Probability of no rain tomorrow if rain today

double P\_R\_given\_R = 0.4; // Probability of rain tomorrow if rain today

// Compute probability of no rain on the day after tomorrow

double P\_N\_day2\_given\_N\_day0 = (P\_N\_given\_N \* P\_N\_given\_N) + (P\_N\_given\_R \* P\_R\_given\_N);

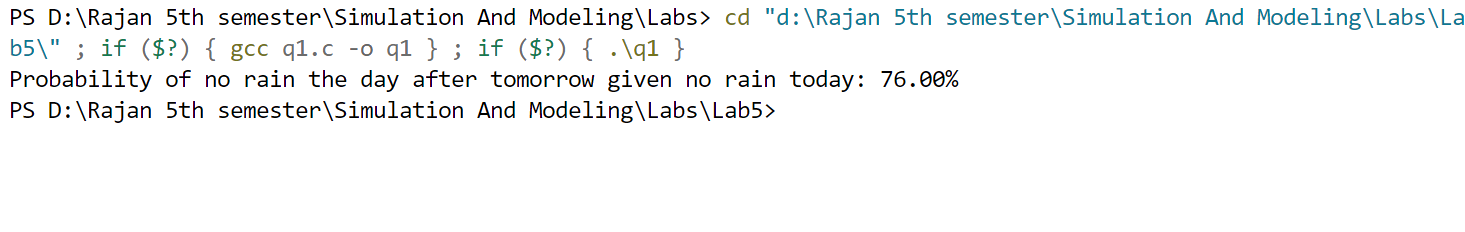
// Display result

printf("Probability of no rain the day after tomorrow given no rain today: %.2f%%\n", P\_N\_day2\_given\_N\_day0 \* 100);

return 0;

}

**Output:**



**2. Coke - Pepsi**

**Coke => 90% Coke**

**Pepsi => 20% Coke**

**Given a person is currently a Pepsi purchaser. What is the probability of purchase of coke after two purchases from now?**

**WAP to solve the above problem.**

**code:**

#include <stdio.h>

int main() {

// Given probabilities

double P\_C\_given\_C = 0.9; // Probability of buying Coke given previous purchase was Coke

double P\_C\_given\_P = 0.2; // Probability of buying Coke given previous purchase was Pepsi

double P\_P\_given\_P = 0.8; // Probability of buying Pepsi given previous purchase was Pepsi

// Compute probability of purchasing Coke after two purchases

double P\_C\_2\_given\_P\_0 = (P\_C\_given\_C \* P\_C\_given\_P) + (P\_C\_given\_P \* P\_P\_given\_P);

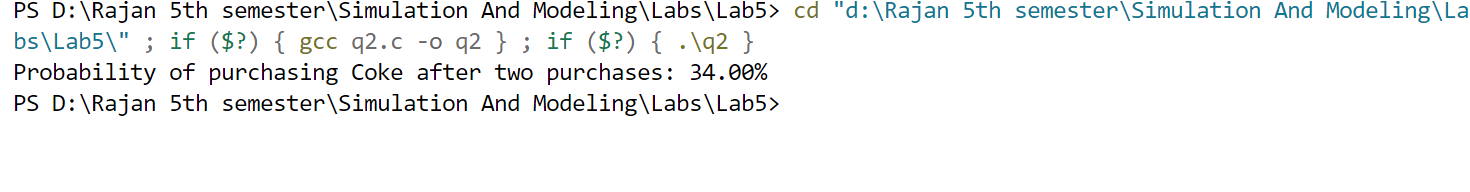
// Display result

printf("Probability of purchasing Coke after two purchases: %.2f%%\n", P\_C\_2\_given\_P\_0 \* 100);

return 0;

}

**Output:**

****