Practical-5

AIM: Implement a program to perform Geometric and Poisson Distribution.

THEORY:

1. GEOMETRIC DISTRIBUTIONThe geometric distribution represents the number of failures before you get a  
success in a series of Bernoulli trials. This discrete probability distribution is  
represented by the probability density function:

f(x) = (1 - p)x - 1p

For example, you ask people outside a polling station who they voted for  
until you find someone that voted for the independent candidate in a local  
election. The geometric distribution would represent the number of people  
who you had to poll before you found someone who voted independent. You  
would need to get a certain number of failures before you got your first  
success.

**Program:**

#include <iostream>

#include <random>

using namespace std;

int main(void) {

const int nrolls = 10000; // number of experiments

const int nstars = 100; // maximum number of stars to distribute

default\_random\_engine generator;

geometric\_distribution <int> distribution (0.3);

int p[10] = {};

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

int number = distribution (generator);

if (number < 10) {

++p[number];

}

}

cout << "geometric\_distribution (0.3):" << endl;

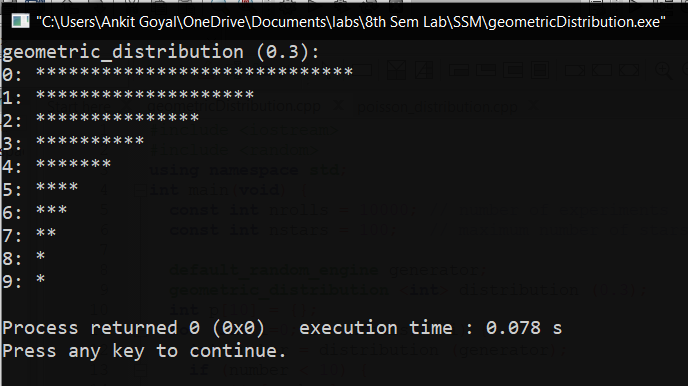
for (int i = 0; i < 10; ++i)

cout << i << ": " << string(p[i] \* nstars / nrolls, '\*') << endl;

return 0;

}

**Output:**



2. POISSON DISTRIBUTIONA Poisson distribution is a tool that helps to predict the probability of certain  
events from happening when you know how often the event has occurred. It  
gives us the probability of a given number of events happening in a fixed  
interval of time.  
The Poisson Distribution pmf is:

P(x; μ) = (e-μ \* μx) / x!

Where:  
● The symbol “!” is a factorial.  
● μ (the expected number of occurrences) is sometimes written as λ.  
Sometimes called the event rate or rate parameter.

**Program:**

#include <iostream>

#include <random>

using namespace std;

int main()

{

const int nrolls = 10000; // number of experiments

const int nstars = 100; // maximum number of stars to distribute

default\_random\_engine generator;

poisson\_distribution<int> distribution(4.1);

int p[10]={};

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

int number = distribution(generator);

if (number<10) ++p[number];

}

cout << "poisson\_distribution (mean=4.1):" << endl;

for (int i=0; i<10; ++i)

cout << i << ": " << string(p[i]\*nstars/nrolls,'\*') << endl;

return 0;

}

**Output:**

