**PRACTICAL 9**

**AIM:** Implement Two server(multi) queuing system.

**THEORY:**

The system consists of multiple servers and a common queue for all items. When any item requests for the server, it is allocated if at-least one server is available. Else the queue begins to start until the server is free. In this system, we assume that all servers are identical, i.e. there is no difference which server is chosen for which item.There is an exception of utilization.

Multi server queue has two or more service facilities in parallel providing identical service. All the. customers in the waiting line can be served by more than one station. The arrival time and the service time. follow poisson and exponential distribution.

**PROGRAM:**

#include<bits/stdc++.h>

using namespace std;

constexpr int FLOAT\_MIN = 0;

constexpr int FLOAT\_MAX = 1;

int main()

{

std::random\_device rd;

std::default\_random\_engine eng(rd());

std::uniform\_real\_distribution<float> distr(FLOAT\_MIN, FLOAT\_MAX);

float r,iat,clock=0,nat,it1,it2,run=150,cit1=0,cit2=0;

float mean, lemda1, lemda2;

cout<<"enter mean time: ";

cin>>mean;

cout<<"service time of server1: ";

cin>>lemda1;

cout<<"service time of server2: ";

cin>>lemda2;

float se1=0,se2=0;

int k,q=0,qmax=3,kont=0,counter;

printf("\n CLOCK IAT NAT SE1 SE2 QUE KONT CIT1 CIT2");

r=distr(eng);

iat=(-mean)\*log(1-r);

nat=nat+iat;

se1=lemda1;

counter=1;

printf("\n %6.2f %6.2f %6.2f %6.2f %6.2f %d %d %6.2f %6.2f ",clock,iat,nat,se1,se2,q,kont,cit1,cit2);

while(clock<=run)

{

if(nat<=se1 && nat<=se2)

{

clock=nat;

q=q+1;

r=distr(eng);

iat = (-mean)\*log(1-r);

nat=nat+iat;

counter=counter+1;

}

else if (se1<=nat && se1<=se2)

clock=se1;

else

clock=se2;

if(q>qmax)

{

kont=kont+1;

q=q-1;

}

else if(q>=1 && se1<=clock)

{

it1=clock-se1;

cit1=cit1+it1;

se1=clock+lemda1;

q=q-1;

}

else if(q>=1 && se2<=clock)

{

it2=clock-se2;

cit2=cit2+it2;

se2=clock+lemda2;

q=q-1;

}

else if(q==0&& se1<=clock)

{

clock=nat;

it1=clock-se1;

cit1=cit1+it1;

se1=nat+lemda1;

r=distr(eng);

iat=(-mean)\*log(1-r);

nat=nat+iat;

counter=counter+1;

}

else if(q==0 && se2<=clock)

{

clock=nat;

it2=clock-se2;

cit2=cit2+it2;

se2=nat+lemda2;

r=distr(eng);

iat=(-mean)\*log(1-r);

nat=nat+iat;

counter=counter+1;

}

printf("\n %6.2f %6.2f %6.2f %6.2f %6.2f %d %d %6.2f %6.2f ",clock,iat,nat,se1,se2,q,kont,cit1,cit2);

}

printf("\n clock=%8.2f cit1=%6.2f cit2=%6.2f counter=%d",clock,cit1,cit2,counter);

printf("\n\n Mean arrival time = %5.2f minutes exponentially distributed",mean);

printf("\n Service time : \nServer1=%5.2f minutes\nServer2=%5.2f minutes",lemda1,lemda2);

printf("\nSimulation run(Elapsed time)=%7.2f minutes",clock);

printf("\nNumber of customers arrived=%d",counter);

printf("\nNumber of customers returned without service=%d",kont);

printf("\nIdle time of server1 = %6.2f minutes",cit1);

printf("\nIdle time of server2 = %6.2f minutes\n",cit2);

}

**OUTPUT:**

