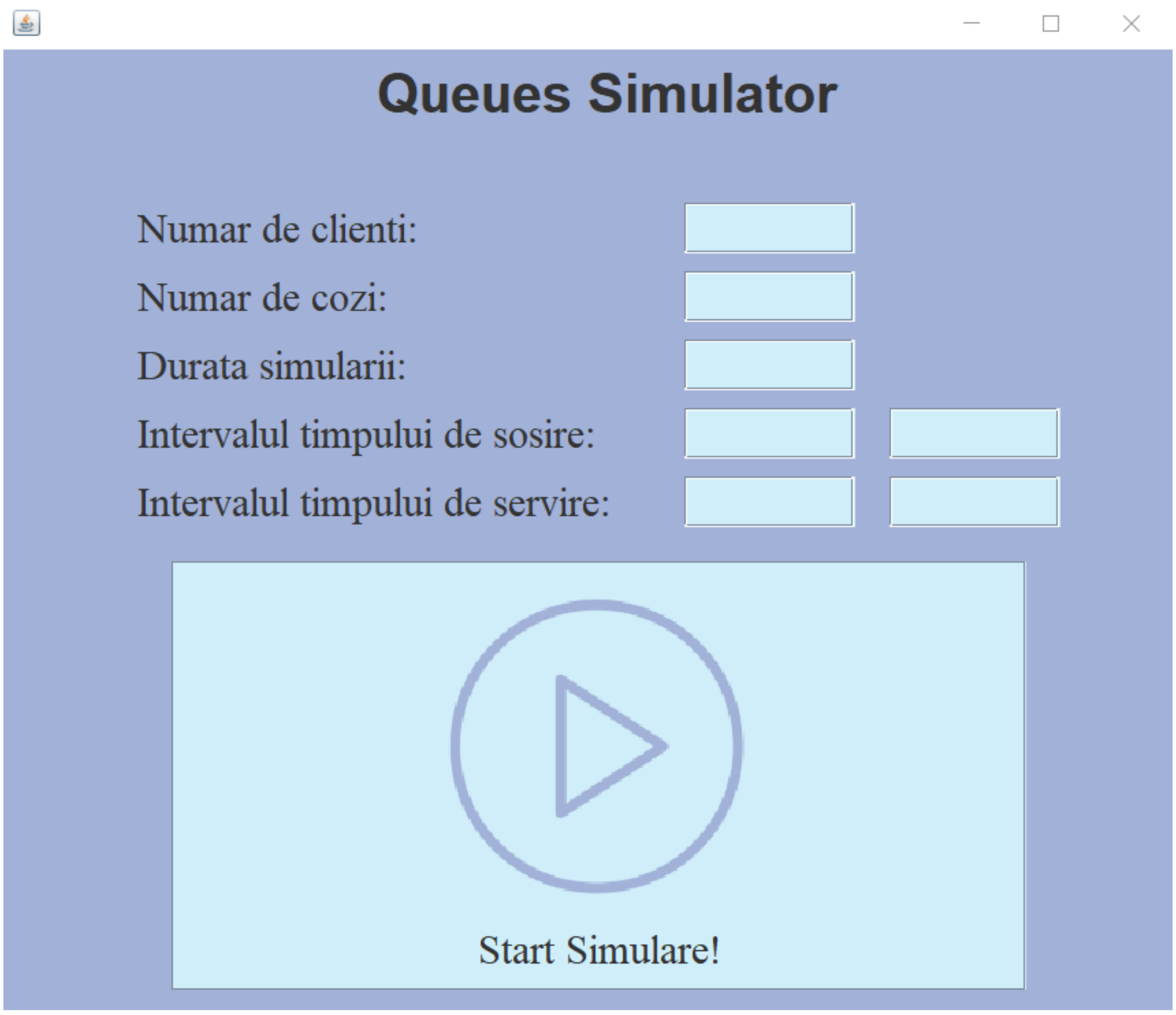
FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
SPECIALIZAREA CALCULATOARE SI TEHNOLOGIA INFORMATIEI

Simularea si analizarea unui sistem de cozi



-Documentație-

Ana-Maria Cusco

An academic: 2020 – 2021

Cuprins

1. Obiectivul temei..3

2. Analiza problemei, modelare scenarii, cazuri de utilizare..4

3. Proiectare (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, interfețe, relații, packages, algoritmi, interfață utilizator) ……………9

4. Implementare………………………………………14

5. Rezultate……………………………………….....16

6. Concluzii……………………………………….....17

7. Bibliografie…………………………………….....17

8. Anexa (fișiere Log)……………………………….....18

Obiectivul Temei

Obiectivul temei este dezvoltarea unei aplicații care sa analizeze sistemele bazate pe cozi pentru a determina si minimiza timpul de așteptare al clienților.

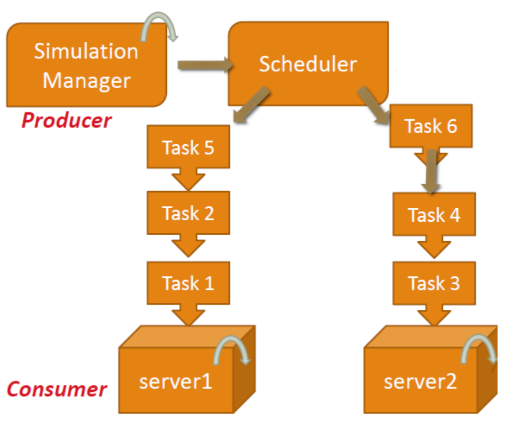
În procesul proiectării, realizării și exploatării sistemelor complexe, modelarea și simularea joacă un rol de incontestabilă importanță, atestată și de sumele imense cheltuite în acest scop în țările dezvoltate. Simularea este tehnica de a imita comportamentul unor anumite situații sau sisteme (economice, mecanice, etc.) cu ajutorul unui model analog celui real, în scopul obținerii unor informații suplimentare sau a specializării personalului. Cu alte cuvinte, simularea este tehnica prin care se construiește modelul unui sistem real, astfel încât comportamentul sistemului în anumite condiții să poată fi studiat și astfel cunoscut. Una din cheile unei bune simulări este abilitatea de a modela comportamentul unui sistem de-a lungul timpului.

Multe simulări conțin cozi ca parte a modelului. Teoria cozilor se refera la modelele matematice folosite pentru a simula aceste cozi.

Cozile sunt de cele mai multe ori folosite pentru a modela probleme reale. Principalul obiectiv al unei cozi este de a-i oferi ‘clientului’ un loc unde sa aștepte înainte de a primi un serviciu. Managementul sistemelor bazate pe cozi își propune sa minimizeze timpul de așteptare al clienților înainte de a primii un serviciu. O modalitate de a minimiza acest timp este de a adaugă mai multe servere (mai multe cozi in sistem – considerând că fiecare coada are un procesor asociat), dar aceasta abordare crește foarte mult costurile producătorului.

Un sistem bazat pe cozi se compune din cel puțin un server (consumer), o coada de așteptare mai multi clienți (sau task-uri), si un manager de simulare (producer) care include un planificator.

Exemple de astfel de sisteme: oamenii care așteaptă la un ATM, cererile unui server web, mașinile care așteaptă la spălătorie.



Multiple servers, multiple tasks (clients)

Analiza problemei, modelare scenarii, cazuri de utilizare

Cerința aplicației este de a simula (prin definirea unui timp de simulare $t_{simulation}$) o serie de N clienți care sosesc pentru servire, intră in Q cozi, așteaptă, sunt serviți si părăsesc cozile. Toți clienții sunt generați când începe simularea și sunt descriși de 3 parametrii: ID (un număr între 1 si N), $t_{arrival}$ (timpul simulării la care ei sunt gata sa intre in coada) si $t_{service}$ (timpul sau durata necesară servirii clientului). Aplicația calculează timpul total petrecut de fiecare client în coadă și timpul mediu de așteptare. Fiecare client e adăugat la coada cu timpul minim de așteptare atunci când timpul sau de sosire $t_{arrival}$ este mai mare sau egal cu timpul de simulare ($t_{arrival} \geq t_{simulation}$.

Următoarele date se considera date de intrare pentru aplicație si trebuie să fie inserate de către utilizator în interfața grafică:

-Numărul de clienți (N)

-Numărul de cozi (Q)

- Durata simulării $t_{simulation}^{MAX}$

-Intervalul timpului de sosire ($t_{arrival}^{MIN} \leq t_{arrival} \leq t_{arrival}^{MAX}$)

-Intervalul timpului de sosire ($t_{service}^{MIN} \leq t_{service} \leq t_{service}^{MAX}$)

In proiectarea unui astfel de sistem bazat pe cozi de așteptare trebuie să ținem cont de mai mulți factori:

Diferite politici de planificare pot fi folosite în nodurile de așteptare:

First in first out (Primul înăuntru primul afară)

Numit și primul-venit, primul-servit (first come, first served), acest principiu afirmă că sunt serviți clienții unul după altul și că acel client care a așteptat cel mai mult este servit primul (conform imaginii).

Last in first out (Ultimul înăuntru primul afară)

Acest principiu, de asemenea, servește clienții unul după altul,, dar clientul cu cel mai scurt timp de așteptare va fi servit primul. Cunoscut și sub numele de stivă.

Partajarea procesorului

Capacitatea de deservire este împărțită în mod egal între clienți.

Prioritate

Clienții cu prioritate mare sunt serviți primii. Cozile cu priorități pot fi de două tipuri, non-preemptive (în cazul în care un loc de muncă în service nu poate fi întrerupt) și premptive (în care o desrvire în curs poate fi întreruptă de o deservire cu o prioritate superioară). În niciunul dintre modele nu se pierde muncă.

Shortest job first (cea mai scurtă deservire prima la rând)

Următoarea deservire este cea mai scurtă ca timp de desfășurare

Preemptive shortest job first (cea mai scurtă deservire preemptivă prima la rând)

Următoarea deservire este cea mai scurtă ca timp de desfășurare de la bun început

Shortest remaining processing time (cel mai scurt timp de procesare rămas)

Următoarea deservire este cea cu cel mai scurt timp de procesare rămas.

Facilitatea deservirii

Single server: customers line up and there is only one server

Parallel servers: customers line up and there are several servers

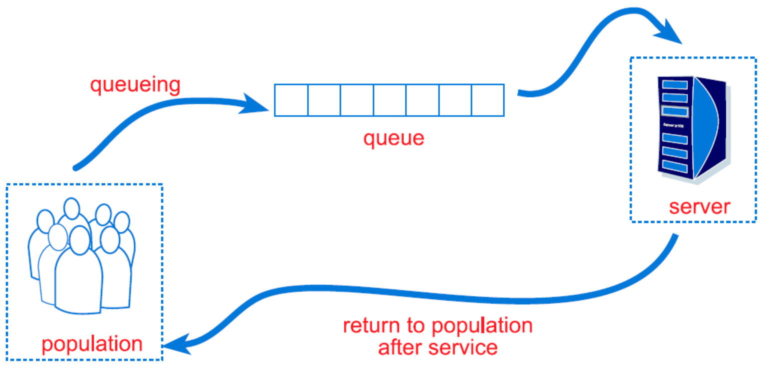
Tandem queue: there are many counters and customers can decide going where to queue

Comportamentul clienților la coadă

Balking: clienții decid să nu se mai așeze la coadă dacă este prea lungă

Jockeying: clienții se mută de la o coadă la alta

Reneging: clienții părăsesc coada de așteptare dacă au așteptat prea mult pentru a fi serviți



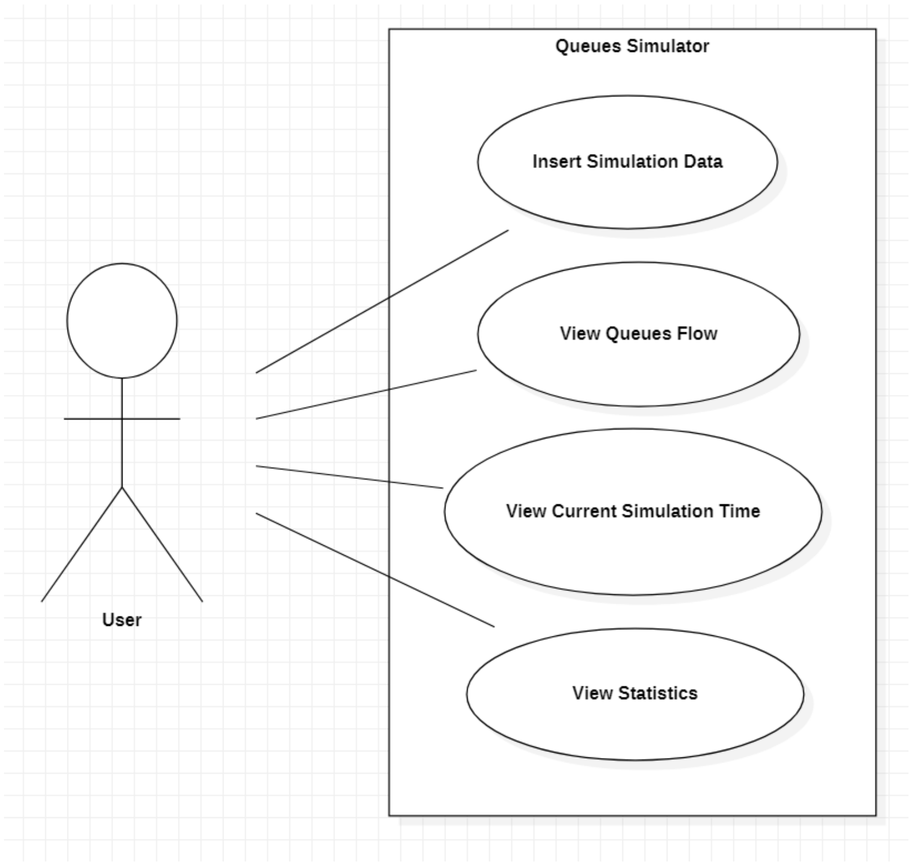
One server one queue model

Din specificațiile proiectului ne rezulta imediat ca politica de planificare va fi una de tip FIFO, iar cea de selecție va fi SHORTEST_TIME (clienții vor fi distribuiți în cozile cu cel mai scurt timp de așteptare).

Problema principală pe care o ridică această aplicație este dată de modul în care modelăm programul astfel încât fluxul de clienți care intră în cozi să fie gestionat în mod eficient și clienții să poată fi repartizați in coada cu cel mai mic timp de așteptare. De asemenea cozile trebuie monitorizate in timp real prin intermediul interfeței grafice, utilizatorul fiind capabil să vadă la fiecare moment de timp numărul de clienți din fiecare coada, momentul când un client părăsește coada sau când se așază un client nou într-o coada. De asemenea, utilizatorul trebuie sa vizualizeze după încheierea simulării o statistică care să cuprindă: ora de vârf (peak hour) – momentul de timp când în cozi au fost cei mai mulți clienți, timpul mediu de așteptare pe cozi (average waiting time) , si timpul mediu de servire (average service time).

Cazuri de utilizare:

-In ceea ce priveste interactiunea utilizatorului cu aplicatia acesta trebuie sa poata introduce datele de simulare, sa vizualizeze timpul curent de simulare, continutul cozilor la fiecare moment de timp precum si statisticile dupa incheierea simualarii:



Use Case Diagram

Pentru a intelege cat mai bine modul in care aplicatia functioneaza o sa prezint in continuare o scurta trasare a fluxului de executie:

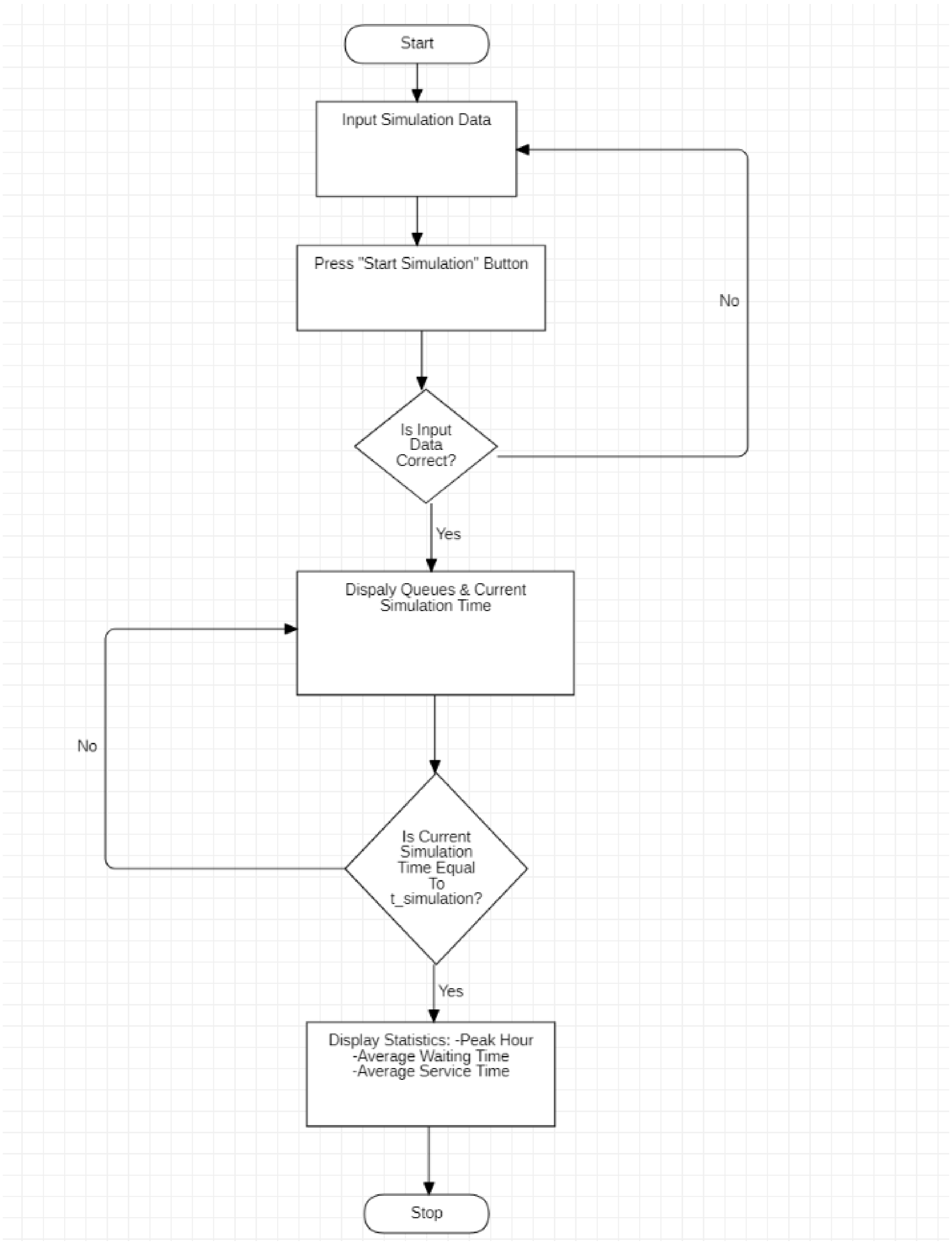
Date de intrare: N=4, Q=2, $t_{simulation}$ = 12, $t_{arrival}^{MIN}$=2, $t_{arrival}^{MAX}$= 10, $t_{service}^{MIN}$=2, t_{serice}^{MAX}= 3

Jurnalul evenimentelor	*Explicatie*
Time 0 **Waiting clients: (1,2,2); (2,3,3); (3,4,3); (4,10,2)** **Queue 1: closed** **Queue 2: closed**	La momentul $t_{simulation}$=0, sunt generarati 4 clienti identificati prin: ID, $t_{arrival}, t_{service}$. Clientul cu ID=1 are un timp de sosire egal cu 2, cee ace implica faptul ca va intra intr-una ditre cozi cand $t_{simulation} \geq 2$. Mai mult decat atat, el are un timp de procesare (servire) egal cu 2, asta inseamna ca va trebui sa stea 2 unitati de timp in fata cozii inainte de a-o parasi. Aceleasi reguli se aplica si pentru urmatorii 3 clienti. Cele doua cozi sunt inchide intrucat nu avem clienti disponibili.
Time 1 **Waiting clients: (1,2,2); (2,3,3); (3,4,3); (4,10,2)** **Queue 1: closed** **Queue 2: closed**	La momentul $t_{simulation} = 1$ niciun client nu poate fi trimis in cozi deoarece nici unul dintre ei nu are $t_{arrival} \leq t_{simulation}$. Cele doua cozi sunt inchide intrucat nu avem clienti disponibili.
Time 2 **Waiting clients: (2,3,3); (3,4,3); (4,10,2)** **Queue 1: (1,2,2);** **Queue 2: closed**	Coada 1 este deschisa si clientul cu ID=1 este plasat primul in aceasta coada intrucat $t_{arrival}^{1} \geq t_{simulation}$ =2. Ceilalti clienti inca asteapta. Coada 2 este inchisa.

Time 3
Waiting clients: (3,4,3); (4,10,2)
Queue 1: (1,2,1);
Queue 2: (2,3,3);
	Coada 2 este deschisa la $t_{simulation}$ = 3, clientul cu ID=2 este plasat in aceasta intrucat $t_{arrival}^{2} \geq t_{simulation}$ =3, si timpul de asteptare al celei de-a doua cozi (0) este mai mic decat timpul de asteptare al primei cozi (1), unde clientul de aici este in curs de procesare (servire).
	
	Ceilalti clienti inca asteapta.

Time 4
Waiting clients: (4,10,2)
Queue 1: (3,4,3);
Queue 2: (2,3,2);
	La momentul $t_{simulation}$ = 4, clientul cu ID=3 este plasat in prima coada intrucat $t_{arrival}^{3} \geq t_{simulation}$ =4.
	
	Mai mult decat atat, clientul cu ID=1 a fost eliminat din prima coada pentru ca timpul lui de servire a ajuns la 0 (a fost 1 la iteratia anterioara si a fost decrementat cu 1 la fiecare pas de simulare).
	
	Clientul din coada 2 are timpul de servire decrementat la 2 pentru ca este procesat.
	
	Ultimul client inca asteapta.

…..	
Statistici: **Timpul mediu de asteptare: 2.5** **Ora de varf: 3 si 4** **Timpul mediu de servire:**	Simularea se termina cand nu mai sunt clienti in coada de asteptare sau in cozile de servicii sau cand $t_{simulation} \geq t_{simulation}^{MAX}$. Ora de varf este momentul de timp cand in cozi se gasesc cei mai multi clienti.


Start
Input Simulation Data
Press "Start Simulation" Button
Is Input Data Correct?
No
Yes
Dispaly Queues & Current Simulation Time
Is Current Simulation Time Equal To t_simulation?
No
Yes
Display Statistics: -Peak Hour
-Average Waiting Time
-Average Service Time
Stop

Application Flowchart

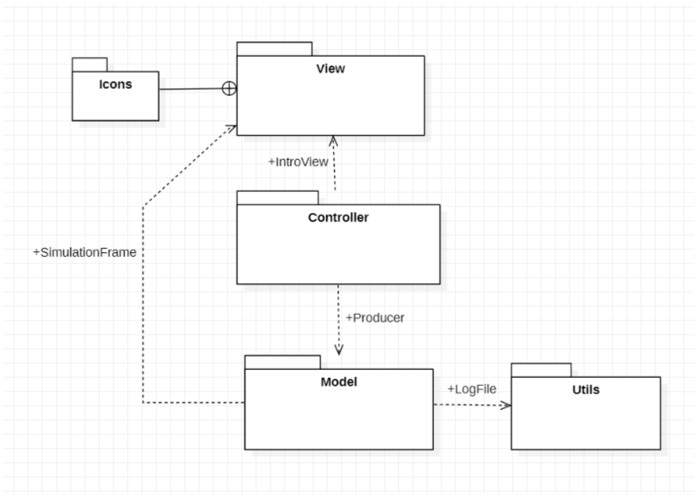
Proiectare

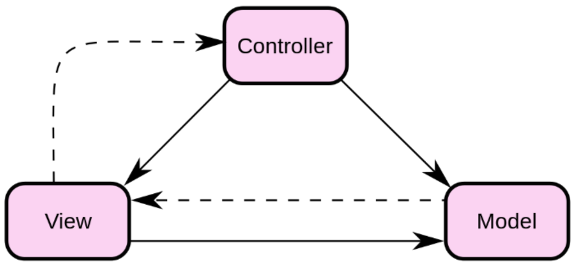
Aceasta aplicatie pe care eu o proiectez, se poate reduce la o problema cunoscuta in Computer Science ca si problema “Producator-Consumator” (sau buffer de legatura). Problema descrie un producator care "produce" in paralel obiecte si le depoziteaza intr-un container comun si avem mai multi consumatori care "consuma" in acelasi timp obiectele depozitate in container de catre producator. Atat producatorul cat si consumatorii vor partaja acelasi container.

In cazul nostru vom avea un producator care va reprezenta un thread (fir de executie) ce va genera obiecte de tip Client si le va adauga pe rand in niste cozi de clienti. Tot in acest timp, consumatorii care sunt la randul lor thread-uri vor incerca sa scoata tot din aceleasi cozi, obiectele de tip Client. Fiecare Consumator va avea propria lui coada de obiecte, iar producatorul va avea access la cozile fiecarui consumator, in care va adauga clienti. Cum aceasta problema este una de paralelism intervine conceptul de sincronizare. Nu putem avea 2 fire de executie care sa lucreze cu acelasi obiect in acelasi timp !!!) – in aplicatia noastra nu este permis ca producatorul sa puna obiecte intr-o coada, si consumatorul sa incerce in acelasi timp sa scoata obiecte din aceeasi coada. Operatiile de adaugare si stergere din structura de date partajata – coada in cazul acesta ar trebui sa se faca pe rand, iar consumatorul nu ar trebui sa incerce sa scoata elemente, cat timp coada este goala, respectiv producatorul sa produca, cat timp coada este plina.

Decizii de proiectare

In proiectarea aplicatiei am ales sa folosesc modelul architectural pe mai multe nivele (eng. layers), mai precis arhitectura Model-View-Controller. Acesst tip de abordare mi-a permis sa imi structurez clasele logic in pachete in functie de tip si de utilizare. Pe langa cele trei pachete se mai poate observa si pachetul Utils care contine o clasa LogFile pe care am folosit-o pentru a scrie intr-un fisier jurnalul de evenimente generat in urma rularii aplicatiei. In pachetul View am un subpachet Icons unde am salvat iconitele folosite in realizarea interfetei grafice.



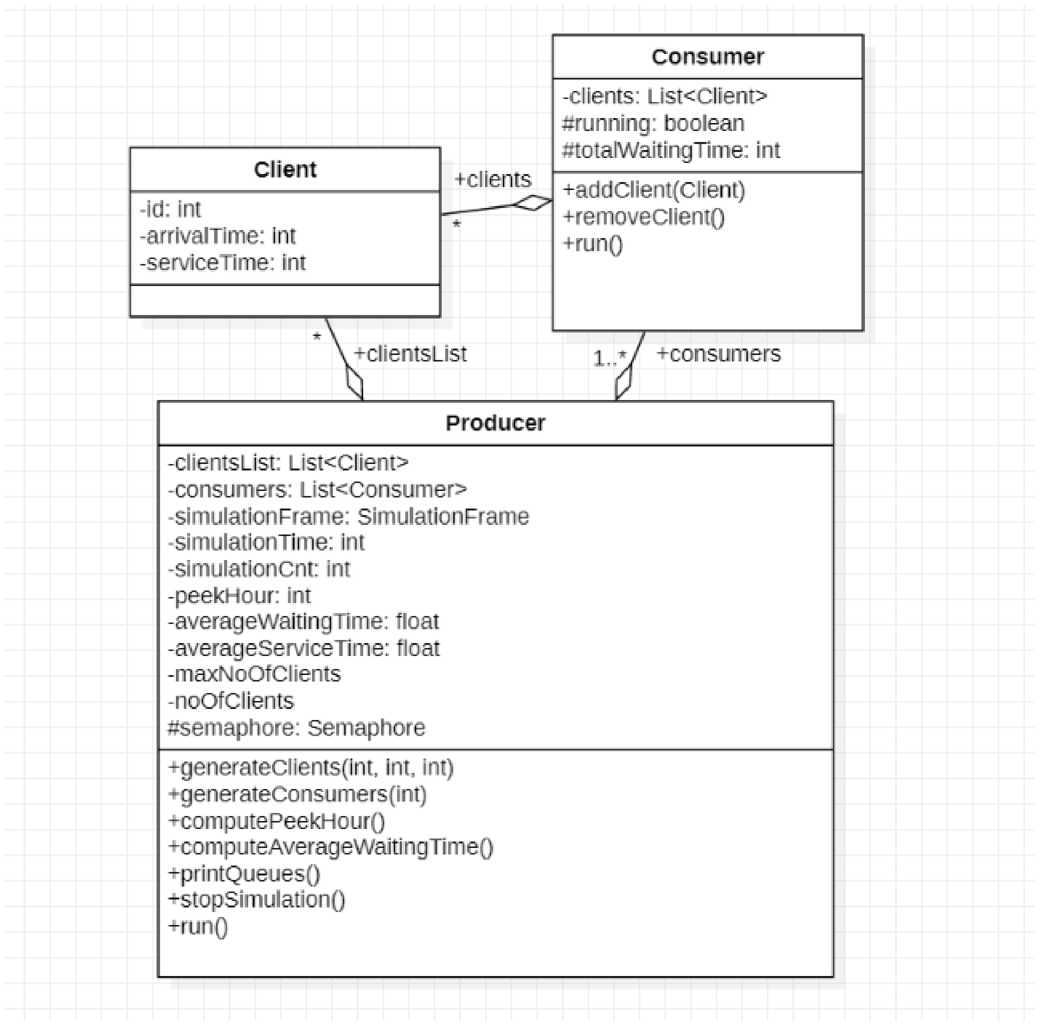


Source: https://ro.wikipedia.org/wiki/Model-view-controller

⬅ Diagrama UML de pachete

Diagrama UML de clase

❖ Pachetul Model contine 3 clase: Client, Producer, Consumer

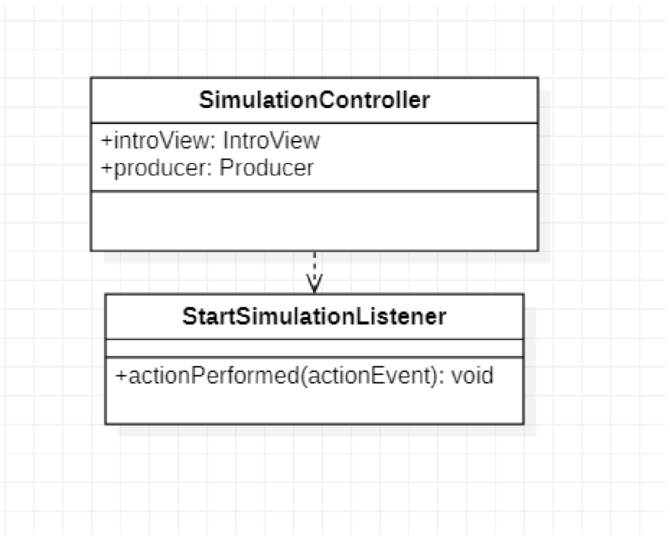


Clasa **Client**: defineste obiectul de lucru, in cazul nostru un client care are care se identifica prin: ID, $t_{arrival}$ -momentul de timp cand clientul este plasat in coada, $t_{service}$ – intervalul de timp in care clientul asteapta in fata cozii, timp in care are loc procesare (sau servirea) acestuia. Alte exemple de astfel de obiecte ar putea fi: task-uri (request-uri) ce vin catre un server, masini care stau la coada la spalatorie. In acest proiect am putea considera clientii dintr-un magazin care asteapta la cozi la casele de marcat.

Clasa **Consumer**: extinde clasa Thread, deci reprezinta un fir de executie si are ca si variabile instanta o lista (coada) de clienti. Foarte important de mentionat ca aceasta structura este impartita cu clasa Producer si trebuie sa fie thread-safe, de aceea am ales sa folosesc metoda syncronizedList() din clasa java.util.Collections pentru a-o sincroniza. Clasa Consumer va scoate din aceasta lista de pe prima pozitie (index 0) cate un client in momentul in care timpul acestuia de procesare (servire) devine 0, cu ajutorul metodei *removeClient()*. De asemenea clasa Producer va apela metoda *addClient(Client)* a acestei clase pentru a adauga in lista unul sau mai multi clienti care au timpul de sosire (arrivalTime) $\leq$ timpul curent de simulare. Tot aici se mai gasesc variabilele instanta running si totalWaitingTime. Variabila running este folosita in metoda *run(),* si cat timp aceasta are valoarea *true* clasa Consumer este in stare activa incercand sa execute codul din interiorul acestei metode, urmand ca Producer-ul sa seteze aceasta variabila la valoarea *false* (opreste Thread-ul) in momentul in care simularea se termina. Variabila totalWaitingTime contorizeaza timpul de asteptare total pe o coada, insumand timpul de asteptare al fiecarui client care este plasat in coada respectiva. Timpul de asteptare al unui client se calculeaza ca fiind suma timpilor de service ale clientilor aflati inaintea sa in coada.

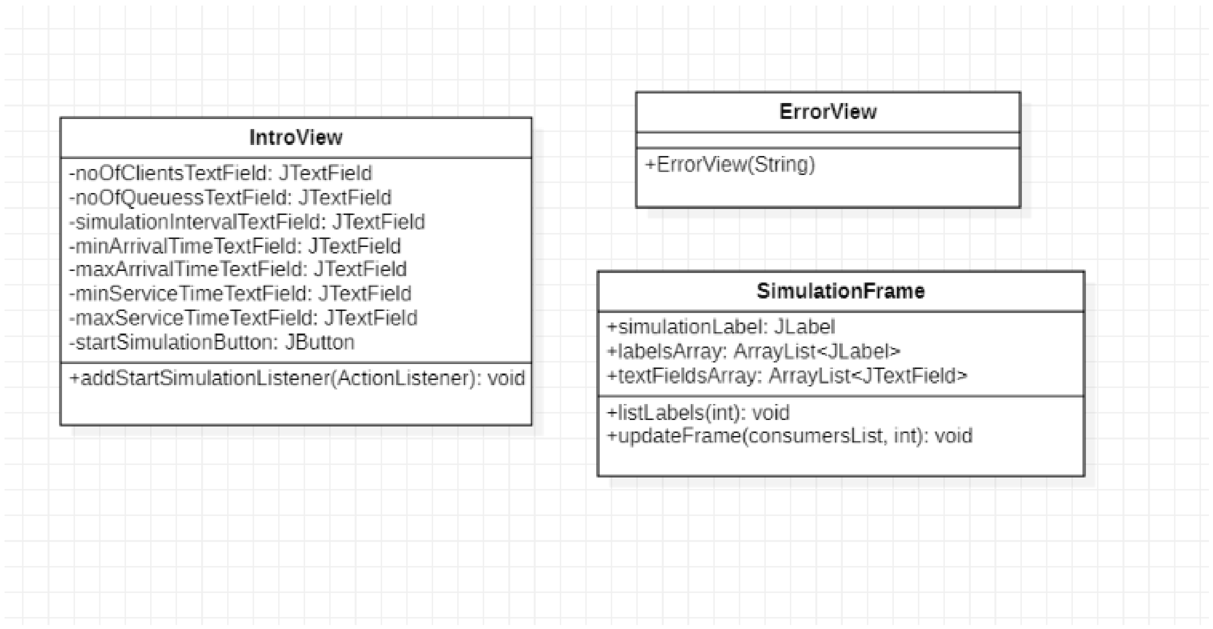
Clasa **Producer**: extinde la randul sau clasa Thread, si reprezinta o clasa esentiala a acestei aplicatii. Acest fir de executie se ocupa cu generarea radom a unei liste de clienti (metoda *generateClients(int,int,int,int)*), crearea unei liste de obiecte Consumer (metoda *generateConsumers(int)*) si pornirea lor cu ajutorul metodei *start()***,** intrucat ele reprezinta fire de executie. In metoda *run()* a acestei clase se face plasarea clientilor in cozile consumatorilor, la fiecare moment de timp se adauga clientii care au timpul de sosire mai mic sau egal cu timpul curent de simulare. Tot in aceasta metoda se face actualizarea interfetei grafice si scrierea intr-un fisier Log a continutului cozilor, la fiecare moment de timp. De asemenea se face evaluarea numarului de clienti din cozi cu fiecare secunda ce trece cu ajutorul metodei *computePeekHour()*, pentru ca in final sa se extraga valoarea timpului la care in cozi se afla cei mai multi clienti – adica ora de varf (eng. peak hour). La expirarea timpului de simulare se face evaluarea timpului de asteptare mediu prin intermediul metodei *computeAverageWaiting(),* iar la generarea clientilor se face calculul timpului mediu de servire (metoda *computeAverageServiceTime()*). Timpul mediu de asteptare se defineste ca raportul dintre suma timpilor de asteptare a tututor clientilor si numarul de clienti. Timpul de asteptare pentru un client se defineste ca fiind suma timpilor de servire a clientilor aflati inaintea sa in coada unui consumator. Tot asa se defineste si timpul mediu de servire ca fiind raportul dintre suma timpilor de servire a tuturor clientilor si numarul de clienti. Metoda *stopSimulation()* se ocupa cu oprirea Thread-urilor Consumer in momentul in care timpul curent de simulare devine egal cu timpul de simulare stabilit de utilizator din interfata grafica. Acest lucru se realizeaza prin setarea variabilei running a fiecarui obiect de tip Consumer la valoarea false.

❖ Pachetul Controller contine o singura clasa si anume: SimulationController.



Clasa **SimulationController**: face legatura intre fereastra din interfata grafica unde i se solicita utilizatorului sa introduca detaliile simularii: numarul de clienti, numarul de cozi, timpul de simulare, timpul minim si maxim de sosire, timpul minim si maxim de servire si thread-ul principal al aplicatiei: clasa Producer. Tot aici se afla ascultatorul pentru butonul „Start Simulare" care creeaza o instanta o obiectului Producer, ce va genera o lista de clienti si o lista de consumeri si le va da start acestora din urma, iar mai apoi va apela el insusi metoda *start()* intrand in starea *ready* (pregatit de executie).

❖ Pachetul View contine clasele: ErrorView, IntroView si SimulationFrame.


IntroView
-noOfClientsTextField: JTextField
-noOfQueuessTextField: JTextField
-simulationIntervalTextField: JTextField
-minArrivalTimeTextField: JTextField
-maxArrivalTimeTextField: JTextField
-minServiceTimeTextField: JTextField
-maxServiceTimeTextField: JTextField
-startSimulationButton: JButton
+addStartSimulationListener(ActionListener): void
ErrorView
+ErrorView(String)
SimulationFrame
+simulationLabel: JLabel
+labelsArray: ArrayList<JLabel>
+textFieldsArray: ArrayList<JTextField>
+listLabels(int): void
+updateFrame(consumersList, int): void

Clasa **IntroView**: se ocupa cu crearea ferestrei in care utilizatorul va introduce datele simularii: numarul de clienti, numarul de cozi, timpul de simulare, timpul minim si maxim de sosire, timpul minim si maxim de servire. TextField-urile pentru aceste date au metode de get() pentru a putea prelua informatia introdusa de utilizator in aceste campuri si a-o transmite mai departe clasei Producer care se ocupa de generarea clientilor si a consumatorilor, prin intermediul clasei

SimulationController care face legatura intre aceste doua. Tot in aceasta fereastra regasim butonul „Start Simulare” a-l carui ascultator se afla in pachetul controller ce l-am mentionat mai sus.

Clasa **ErrorView:** este fereastra care se ocupa cu atentionarea utilizatorului prin mesaje corespunzatoare in cazul in care datele simularii nu au fost introduse corect (nu respecta constrangerile impuse: sa fie numere naturale, iar intervalele sa fie precizate in secunde).

Clasa **SimulationFrame**: prezinta fereastra de simulare, si contine un label care indica timpul curent de simulare precum si un numar de cozi egal cu numarul specificat de utilizator prin intermediul ferestrei IntroView, cozi a caror continut se modifica la fiecare moment de timp, cu fiecare client nou ce soseste in aceste cozi sau le paraseste, prin intermediul metodei *updateFrame (List<Consumers>, int).* Fiecare client din coada este reprezentat printr-un caracter special UNICODE 👤, fiind simulata in timp real asezarea clientilor in cozi si parasirea cozilor de catre clienti. Daca o coada nu are niciun client la un moment de timp se afiseaza mesajul empty.

Structuri de date

Ca si structuri de date am folosit interfata *List<T>* si clasa *ArrayList<T>*
a clasei Collections. In instantele clasei ArrayList<T> stochez lista de clienti cu date aleatorii generata de catre clasa Producer, lista de obicte Consumer, iar in fiecare clasa Consumer am cate o lista de clienti asupra careia am apelat metoda syncronizedList() tot din clasa Collections pentru a obtine o lista sinzronizata (thread-safe), intrucat aceasta lista va fi accesata atat de catre thread-ul Producer pentru a adauga clienti cat si de catre thread-ul Consumer pentru a scoate clienti.

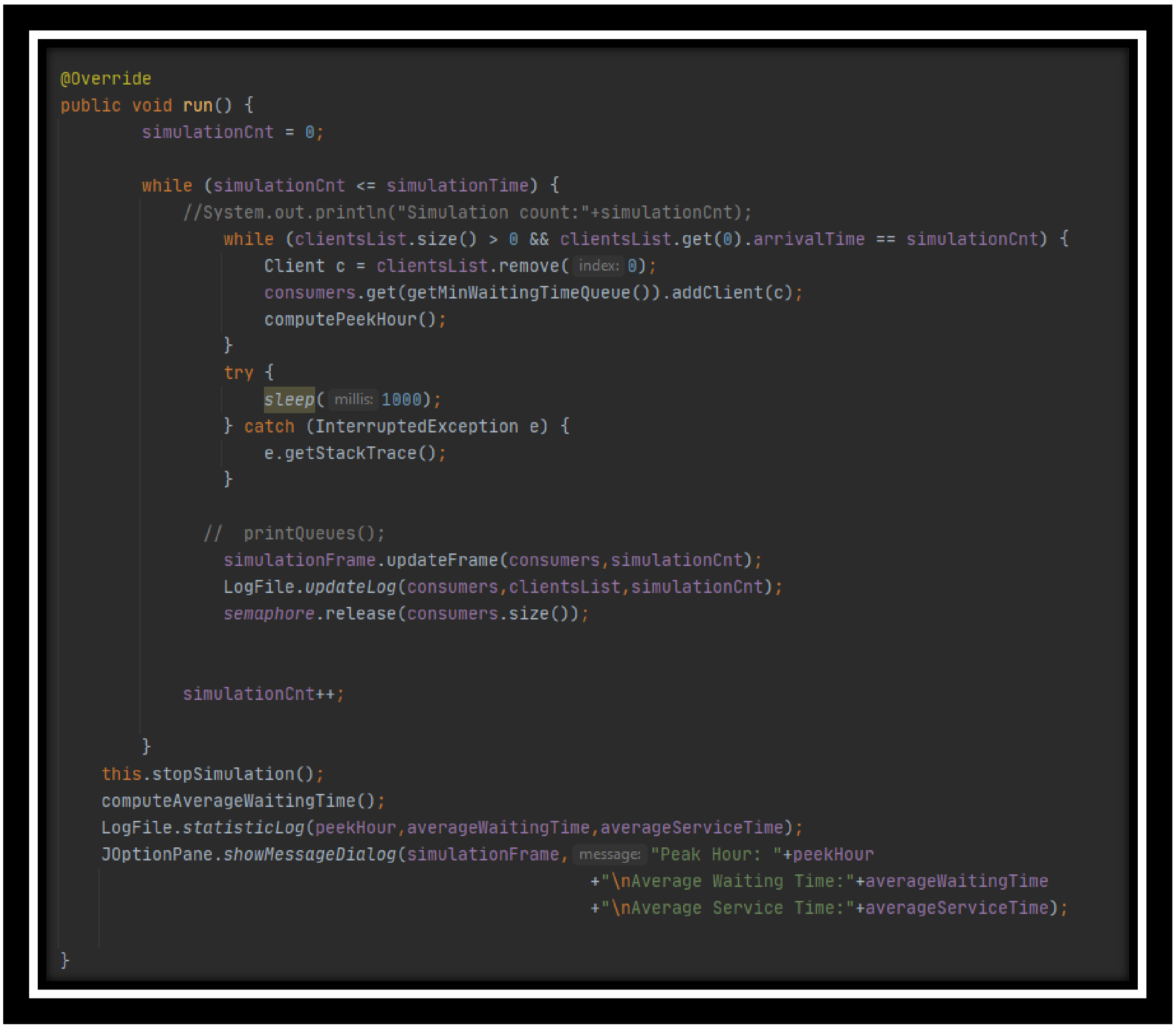
Interfata Utilizator

In ceea ce priveste interfata grafica cu care interactioneaza utilizatorul, am mai multe ferestre, una in care se introduc datele simularii, alta in care se vizualizeaza continutul cozilor si timpul curent de simulare in timp real, iar la incheierea simularii se pot vizualiza statistic: timpul mediu de asteptare, timpul mediu de servire si ora de varf printr-o fereasta pop-up.

Implementare

- Pentru implementarea aplicatiei am folosit mediul de dezvoltare IntelliJ IDEA si limbajul Java care ofera suport pentru Multithreading.

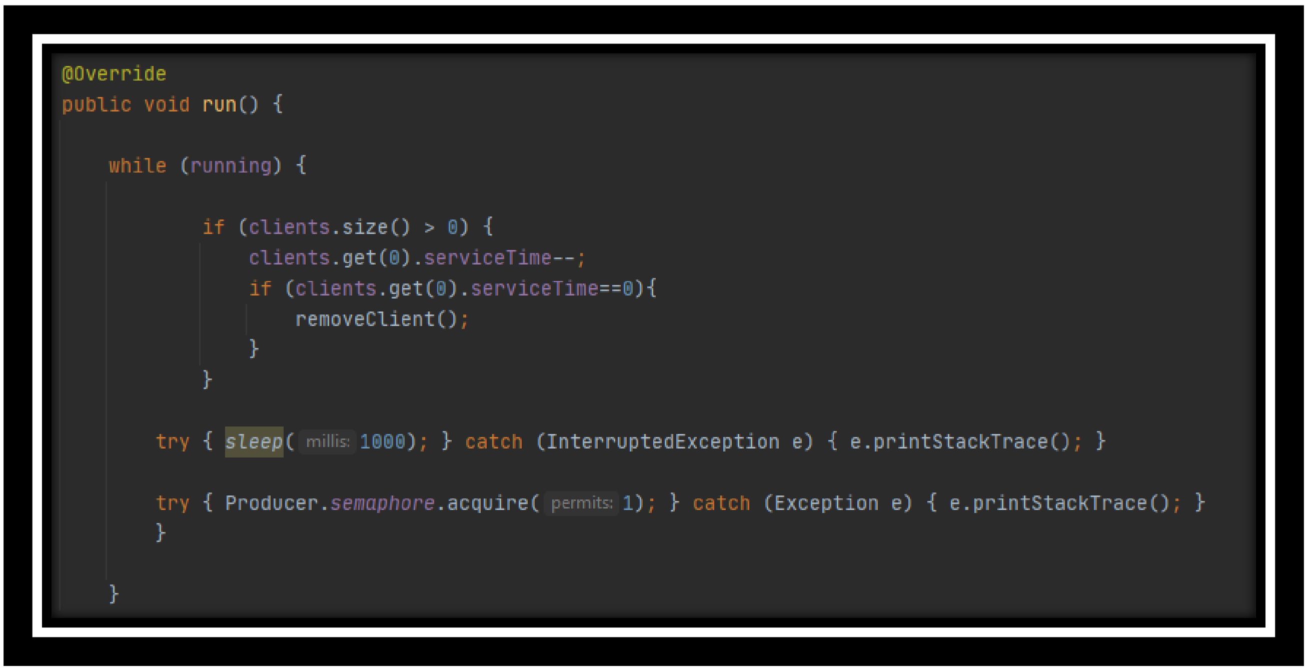
  ➔ Implementarea metodei *run()* din **clasa Producer**



```
@Override
public void run() {
        simulationCnt = 0;

        while (simulationCnt <= simulationTime) {
            //System.out.println("Simulation count:"+simulationCnt);
                while (clientsList.size() > 0 && clientsList.get(0).arrivalTime == simulationCnt) {
                    Client c = clientsList.remove( index: 0);
                    consumers.get(getMinWaitingTimeQueue()).addClient(c);
                    computePeekHour();
                }
                try {
                    sleep( millis: 1000);
                } catch (InterruptedException e) {
                    e.getStackTrace();
                }

              //  printQueues();
                simulationFrame.updateFrame(consumers,simulationCnt);
                LogFile.updateLog(consumers,clientsList,simulationCnt);
                semaphore.release(consumers.size());


            simulationCnt++;

        }
    this.stopSimulation();
    computeAverageWaitingTime();
    LogFile.statisticLog(peekHour,averageWaitingTime,averageServiceTime);
    JOptionPane.showMessageDialog(simulationFrame, message: "Peak Hour: "+peekHour
                                                +"\nAverage Waiting Time:"+averageWaitingTime
                                                +"\nAverage Service Time:"+averageServiceTime);

}
```

In aceasta metoda se face adaugarea clientilor in cozi (se adauga acei client care au timpul de sosire egal cu timpul current de simulare), actualizarea interfetei grafice cu continutul cozilor si timpul curent, scrierea in fisierul Log a continutului curent al cozilor, al clientilor care sunt inca in asteptare pentru a fi adaugati in cozi, precum si evaluarea numarului de clienti din toate cozile pentru calculul orei de varf (eng. peak hour).

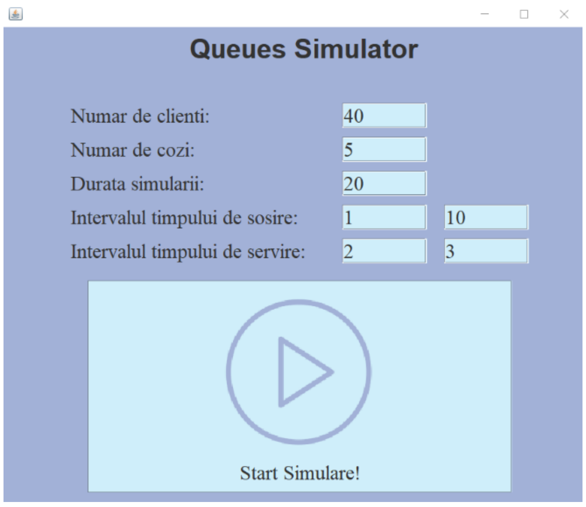
➔ Implementarea metodei *run()* din **clasa Consumer**



```java
@Override
public void run() {

    while (running) {

            if (clients.size() > 0) {
                clients.get(0).serviceTime--;
                if (clients.get(0).serviceTime==0){
                    removeClient();
                }
            }

        try { sleep( millis: 1000); } catch (InterruptedException e) { e.printStackTrace(); }

        try { Producer.semaphore.acquire( permits: 1); } catch (Exception e) { e.printStackTrace(); }
        }

    }
```

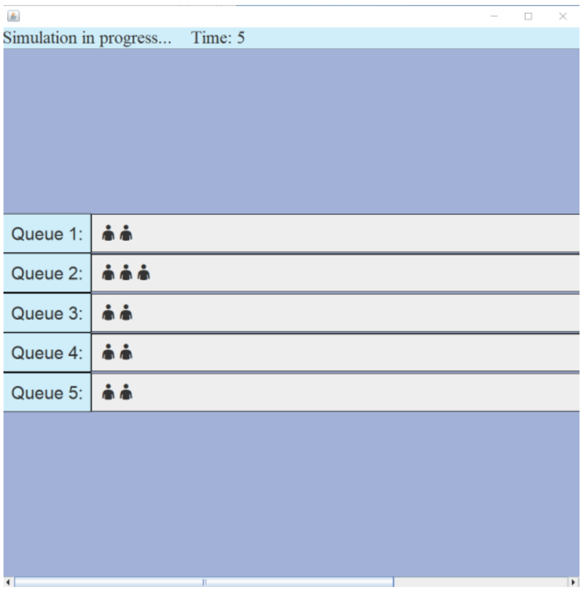
In aceasta metoda se face decrementarea timpului de servire al unui client daca acesta nu a ajuns la valoarea 0, respectiv eliminarea clientului din coada cand timpul sau de service are valoarea 0.
Pentru a putea realiza sincronizarea intre Thread-urile Consumer si Thread-ul Producer am folosit un semafor. Aceasta sincronizare presupune ca Thread-urile Consumer nu vor incerca sa scoata din cozi clienti sau sa decrementeze timpul acestora de service inainte ca Thread-ul Producer sa adauge clienti in cozi, iar fiecare dintre aceste Thread-uri nu va actiona decat asupra unui client pe secunda. Astfel Thread-ul Producer il putem asemana cu un bodyguard sau un paznic, care dupa ce si-a executat codul ofera un numar de permisiuni egale cu numarul de consumatori (*semaphore.release(consumers.size())*), iar fiecare dintre consumatori va cere si va obtine o singura permisiune de la acest fir de executie (*semaphore.acquire(1)*), asigurandu-se in acest fel siguranta datelor adaugate sau scoase din cozi.

Rezultate

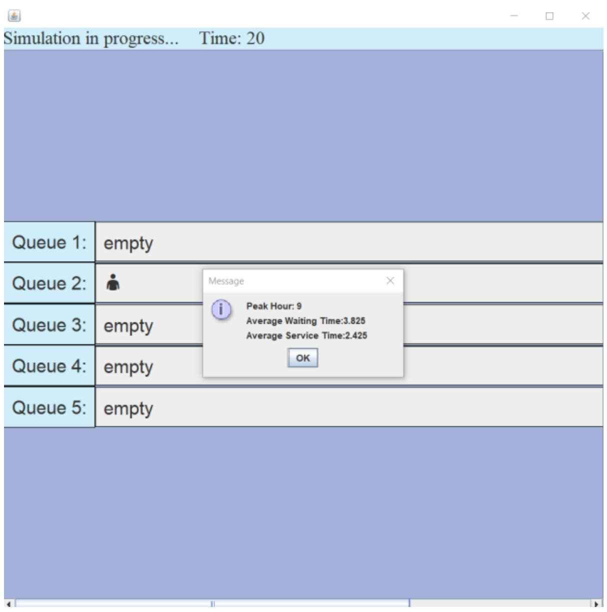


Numar de clienti: 40
Numar de cozi: 5
Durata simularii: 20
Intervalul timpului de sosire: 1 10
Intervalul timpului de servire: 2 3

➔ Se introduc datele de simulare si se apasa butonul “Start Simualare”

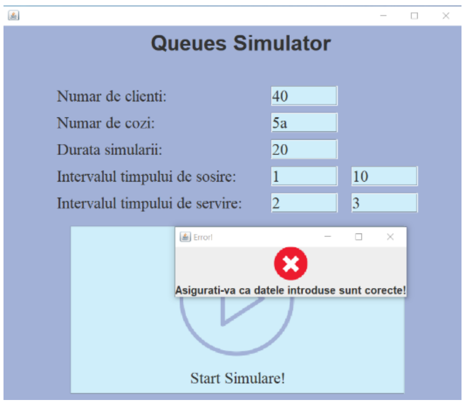

Simulation in progress... Time: 5

Queue 1:
Queue 2:
Queue 3:
Queue 4:
Queue 5:


Simulation in progress... Time: 20

Queue 1: empty
Queue 2:
Queue 3: empty
Queue 4: empty
Queue 5: empty

Peak Hour: 9
Average Waiting Time:3.825
Average Service Time:2.425



Numar de clienti: 40
Numar de cozi: 5a
Durata simularii: 20
Intervalul timpului de sosire: 1 10
Intervalul timpului de servire: 2 3

Asigurati-va ca datele introduse sunt corecte!

➔ Afisarea mesajelor de eroare

Concluzii

Dezvoltarea acestei aplicatii m-a ajutat sa inteleg niste concepte cu totul noi si anume: Thread-urile. Am invatat de ce este nevoie de sincronizare in programarea concurenta, si cum niciodata nu vom stii in ce ordine sunt planificate Thread-urile spre executie, aceest lucru realizandu-se in mod transparent pentru utilizator de catre JVM. De asemenea am observant consecinta directa a folosirii mai multor fire de executie, si anume imbunatatirea semnificativa a performantei aplicatiei.

Ca si posibile dezvoltari ulterioare s-ar putea aduce imbunatatiri pe parte de interfata grafica, s-ar putea folosi algoritmi care sa ne reduca timpul de asteptare, sau selectia altei politici de adaugare a clientilor in cozi, s-ar putea calcula statistici mai avansate, etc.

Bibliografie

- FUNDAMENTAL PROGRAMMING TECHNIQUES (ASSIGNMENT 2 SUPPORT PRESENTATION)
- http://docs.oracle.com/javase/tutorial/essential/concurrency/index.html
- http://www.tutorialspoint.com/java/util/timer_schedule_period.htm
- -http://www.javacodegeeks.com/2013/01/java-thread-pool-example-using-executors-and-threadpoolexecutor.html
- https://en.wikipedia.org/wiki/Queueing_theory
- Java Multithreading - https://www.youtube.com/playlist?list=PLBB24CFB073F1048E

Anexa - Loguri

Test 1:

N=4, Q=2, $t_{simualtion}^{MAX} = 60\ seconds$, $[t_{arrival}^{MIN}, t_{arrival}^{MAX}]$=[2,30], $[t_{service}^{MIN}, t_{service}^{MAX}]$=[2,4]

Time 0 - 5 Waiting clients:(1,6,3) (3,12,3) (4,18,2) (2,21,2) Queue 1:empty Queue 2:empty	Time 6 Waiting clients:(3,12,3) (4,18,2) (2,21,2) Queue 1:(1,6,3) Queue 2:empty
Time 7 Waiting clients:(3,12,3) (4,18,2) (2,21,2) Queue 1:(1,6,2) Queue 2:empty	Time 8 Waiting clients:(3,12,3) (4,18,2) (2,21,2) Queue 1:(1,6,1) Queue 2:empty
Time 9 - 11 Waiting clients:(3,12,3) (4,18,2) (2,21,2) Queue 1:empty Queue 2:empty	Time 12 Waiting clients:(4,18,2) (2,21,2) Queue 1:(3,12,3) Queue 2:empty
Time 13 Waiting clients:(4,18,2) (2,21,2) Queue 1:(3,12,2) Queue 2:empty	Time 14 Waiting clients:(4,18,2) (2,21,2) Queue 1:(3,12,1) Queue 2:empty
Time 15 - 17 Waiting clients:(4,18,2) (2,21,2) Queue 1:empty Queue 2:empty	Time 18 Waiting clients:(2,21,2) Queue 1:(4,18,2) Queue 2:empty
Time 19 Waiting clients:(2,21,2) Queue 1:(4,18,1) Queue 2:empty	Time 20 Waiting clients:(2,21,2) Queue 1:empty Queue 2:empty
Time 21 Waiting clients: Queue 1:(2,21,2) Queue 2:empty	Time 22 Waiting clients: Queue 1:(2,21,1) Queue 2:empty
Time 23 Waiting clients: Queue 1:empty Queue 2:empty	
Statistics: Peak Hour: 6 Average Waiting Time: 0.0 Average Service Time: 2.5	

Test 2:

N=50, Q=5, $t_{simualtion}^{MAX} = 60\ seconds$, $[t_{arrival}^{MIN}, t_{arrival}^{MAX}]$=[2,40], $[t_{service}^{MIN}, t_{service}^{MAX}]$=[1,7]

Time 0 - 1 Waiting clients:(48,2,5) (37,3,6) (14,4,4) (20,4,6) (47,4,2) (34,6,4) (36,6,4) (2,7,4) (4,7,4) (38,7,1) (42,7,6) (8,8,3) (39,8,4) (13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:empty Queue 2:empty Queue 3:empty Queue 4:empty Queue 5:empty	Time 2 Waiting clients:(37,3,6) (14,4,4) (20,4,6) (47,4,2) (34,6,4) (36,6,4) (2,7,4) (4,7,4) (38,7,1) (42,7,6) (8,8,3) (39,8,4) (13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(48,2,5) Queue 2:empty Queue 3:empty Queue 4:empty Queue 5:empty	Time 3 Waiting clients:(14,4,4) (20,4,6) (47,4,2) (34,6,4) (36,6,4) (2,7,4) (4,7,4) (38,7,1) (42,7,6) (8,8,3) (39,8,4) (13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(48,2,4) Queue 2:(37,3,6) Queue 3:empty Queue 4:empty Queue 5:empty
Time 4 Waiting clients:(34,6,4) (36,6,4) (2,7,4) (4,7,4) (38,7,1) (42,7,6) (8,8,3) (39,8,4) (13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(48,2,3) Queue 2:(37,3,5) Queue 3:(14,4,4) Queue 4:(20,4,6) Queue 5:(47,4,2)	Time 5 Waiting clients:(34,6,4) (36,6,4) (2,7,4) (4,7,4) (38,7,1) (42,7,6) (8,8,3) (39,8,4) (13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(48,2,2) Queue 2:(37,3,4) Queue 3:(14,4,3) Queue 4:(20,4,5) Queue 5:(47,4,1)	Time 6 Waiting clients:(2,7,4) (4,7,4) (38,7,1) (42,7,6) (8,8,3) (39,8,4) (13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(48,2,1) (36,6,4) Queue 2:(37,3,3) Queue 3:(14,4,2) Queue 4:(20,4,4) Queue 5:(34,6,4)
Time 7 Waiting clients:(8,8,3) (39,8,4) (13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(36,6,4) Queue 2:(37,3,2) (4,7,4) Queue 3:(14,4,1) (2,7,4) Queue 4:(20,4,3) (38,7,1) Queue 5:(34,6,3) (42,7,6)	Time 8 Waiting clients:(13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(36,6,3) (8,8,3) Queue 2:(37,3,1) (4,7,4) Queue 3:(2,7,4) Queue 4:(20,4,2) (38,7,1) (39,8,4) Queue 5:(34,6,2) (42,7,6)	Time 9 Waiting clients:(13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(36,6,2) (8,8,3) Queue 2:(4,7,4) Queue 3:(2,7,3) Queue 4:(20,4,1) (38,7,1) (39,8,4) Queue 5:(34,6,1) (42,7,6)
Time 10 Waiting clients:(13,11,5) (19,11,7) (35,11,3) (3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(36,6,1) (8,8,3) Queue 2:(4,7,3) Queue 3:(2,7,2)	Time 11 Waiting clients:(3,12,7) (50,12,2) (30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(8,8,3) (35,11,3) Queue 2:(4,7,2) (19,11,7) Queue 3:(2,7,1) (13,11,5)	Time 12 Waiting clients:(30,13,1) (41,13,1) (24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(8,8,2) (35,11,3) Queue 2:(4,7,1) (19,11,7) Queue 3:(13,11,5)

Queue 4:(38,7,1) (39,8,4) Queue 5:(42,7,6)	Queue 4:(39,8,4) Queue 5:(42,7,5)	Queue 4:(39,8,3) (3,12,7) Queue 5:(42,7,4) (50,12,2)
Time 13 Waiting clients:(24,14,7) (29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(8,8,1) (35,11,3) (30,13,1) Queue 2:(19,11,7) Queue 3:(13,11,4) (41,13,1) Queue 4:(39,8,2) (3,12,7) Queue 5:(42,7,3) (50,12,2)	Time 14 Waiting clients:(29,15,7) (27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(35,11,3) (30,13,1) (24,14,7) Queue 2:(19,11,6) Queue 3:(13,11,3) (41,13,1) Queue 4:(39,8,1) (3,12,7) Queue 5:(42,7,2) (50,12,2)	Time 15 Waiting clients:(27,16,4) (11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(35,11,2) (30,13,1) (24,14,7) Queue 2:(19,11,5) Queue 3:(13,11,2) (41,13,1) (29,15,7) Queue 4:(3,12,7) Queue 5:(42,7,1) (50,12,2)
Time 16 Waiting clients:(11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(35,11,1) (30,13,1) (24,14,7) Queue 2:(19,11,4) Queue 3:(13,11,1) (41,13,1) (29,15,7) Queue 4:(3,12,6) Queue 5:(50,12,2) (27,16,4)	Time 17 Waiting clients:(11,18,2) (18,18,1) (6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(30,13,1) (24,14,7) Queue 2:(19,11,3) Queue 3:(41,13,1) (29,15,7) Queue 4:(3,12,5) Queue 5:(50,12,1) (27,16,4)	Time 18 Waiting clients:(6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(24,14,7) Queue 2:(19,11,2) (11,18,2) (18,18,1) Queue 3:(29,15,7) Queue 4:(3,12,4) Queue 5:(27,16,4)
Time 19 Waiting clients:(6,20,2) (26,20,4) (40,20,4) (1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(24,14,6) Queue 2:(19,11,1) (11,18,2) (18,18,1) Queue 3:(29,15,6) Queue 4:(3,12,3) Queue 5:(27,16,3)	Time 20 Waiting clients:(1,21,1) (7,21,6) (25,21,1) (5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(24,14,5) Queue 2:(11,18,2) (18,18,1) (40,20,4) Queue 3:(29,15,5) Queue 4:(3,12,2) (6,20,2) Queue 5:(27,16,2) (26,20,4)	Time 21 Waiting clients:(5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(24,14,4) (7,21,6) Queue 2:(11,18,1) (18,18,1) (40,20,4) Queue 3:(29,15,4) (25,21,1) Queue 4:(3,12,1) (6,20,2) (1,21,1) Queue 5:(27,16,1) (26,20,4)
Time 22 Waiting clients:(5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(24,14,3) (7,21,6) Queue 2:(18,18,1) (40,20,4) Queue 3:(29,15,3) (25,21,1) Queue 4:(6,20,2) (1,21,1) Queue 5:(26,20,4)	Time 23 Waiting clients:(5,24,3) (12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(24,14,2) (7,21,6) Queue 2:(40,20,4) Queue 3:(29,15,2) (25,21,1) Queue 4:(6,20,1) (1,21,1) Queue 5:(26,20,3)	Time 24 Waiting clients:(12,25,4) (16,25,5) (43,25,5) (23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(24,14,1) (7,21,6) Queue 2:(40,20,3) Queue 3:(29,15,1) (25,21,1) Queue 4:(1,21,1) (5,24,3) Queue 5:(26,20,2)
Time 25 Waiting clients:(23,26,2) (45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4)	Time 26 Waiting clients:(45,27,5) (10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4)	Time 27 Waiting clients:(10,28,4) (32,28,3) (17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4)

Queue 1:(7,21,6) Queue 2:(40,20,2) (43,25,5) Queue 3:(25,21,1) (12,25,4) Queue 4:(5,24,3) Queue 5:(26,20,1) (16,25,5)	Queue 1:(7,21,5) Queue 2:(40,20,1) (43,25,5) Queue 3:(12,25,4) Queue 4:(5,24,2) (23,26,2) Queue 5:(16,25,5)	Queue 1:(7,21,4) Queue 2:(43,25,5) Queue 3:(12,25,3) (45,27,5) Queue 4:(5,24,1) (23,26,2) Queue 5:(16,25,4)
Time 28 Waiting clients:(17,29,5) (28,29,1) (44,29,3) (15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(7,21,3) (32,28,3) Queue 2:(43,25,4) Queue 3:(12,25,2) (45,27,5) Queue 4:(23,26,2) (10,28,4) Queue 5:(16,25,3)	Time 29 Waiting clients:(15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(7,21,2) (32,28,3) Queue 2:(43,25,3) (28,29,1) (44,29,3) Queue 3:(12,25,1) (45,27,5) Queue 4:(23,26,1) (10,28,4) Queue 5:(16,25,2) (17,29,5)	Time 30 Waiting clients:(15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(7,21,1) (32,28,3) Queue 2:(43,25,2) (28,29,1) (44,29,3) Queue 3:(45,27,5) Queue 4:(10,28,4) Queue 5:(16,25,1) (17,29,5)
Time 31 Waiting clients:(15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(32,28,3) Queue 2:(43,25,1) (28,29,1) (44,29,3) Queue 3:(45,27,4) Queue 4:(10,28,3) Queue 5:(17,29,5)	Time 32 Waiting clients:(15,33,2) (9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(32,28,2) Queue 2:(28,29,1) (44,29,3) Queue 3:(45,27,3) Queue 4:(10,28,2) Queue 5:(17,29,4)	Time 33 Waiting clients:(9,34,5) (21,34,6) (49,34,1) (33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(32,28,1) (15,33,2) Queue 2:(44,29,3) Queue 3:(45,27,2) Queue 4:(10,28,1) Queue 5:(17,29,3)
Time 34 Waiting clients:(33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(15,33,2) (49,34,1) Queue 2:(44,29,2) Queue 3:(45,27,1) (21,34,6) Queue 4:(9,34,5) Queue 5:(17,29,2)	Time 35 Waiting clients:(33,36,6) (22,38,5) (31,40,1) (46,40,4) Queue 1:(15,33,1) (49,34,1) Queue 2:(44,29,1) Queue 3:(21,34,6) Queue 4:(9,34,4) Queue 5:(17,29,1)	Time 36 Waiting clients:(22,38,5) (31,40,1) (46,40,4) Queue 1:(49,34,1) Queue 2:(33,36,6) Queue 3:(21,34,5) Queue 4:(9,34,3) Queue 5:empty
Time 37 Waiting clients:(22,38,5) (31,40,1) (46,40,4) Queue 1:empty Queue 2:(33,36,5) Queue 3:(21,34,4) Queue 4:(9,34,2) Queue 5:empty	Time 38 Waiting clients:(31,40,1) (46,40,4) Queue 1:(22,38,5) Queue 2:(33,36,4) Queue 3:(21,34,3) Queue 4:(9,34,1) Queue 5:empty	Time 39 Waiting clients:(31,40,1) (46,40,4) Queue 1:(22,38,4) Queue 2:(33,36,3) Queue 3:(21,34,2) Queue 4:empty Queue 5:empty
Time 40 Waiting clients: Queue 1:(22,38,3) Queue 2:(33,36,2) Queue 3:(21,34,1) Queue 4:(31,40,1) Queue 5:(46,40,4)	Time 41 Waiting clients: Queue 1:(22,38,2) Queue 2:(33,36,1) Queue 3:empty Queue 4:empty Queue 5:(46,40,3)	Time 42 Waiting clients: Queue 1:(22,38,1) Queue 2:empty Queue 3:empty Queue 4:empty Queue 5:(46,40,2)

Time 43 Waiting clients: Queue 1:empty Queue 2:empty Queue 3:empty Queue 4:empty Queue 5:(46,40,1)	Time 44 Waiting clients: Queue 1:empty Queue 2:empty Queue 3:empty Queue 4:empty Queue 5:empty	
Statistics: Peak Hour: 21	Average Waiting Time: 1.96	Average Service Time: 3.76

Test 3:

N=1000, Q=20, $t_{simualtion}^{MAX} = 200\ seconds$, [$t_{arrival}^{MIN}, t_{arrival}^{MAX}$]=[10,100], [$t_{service}^{MIN}, t_{service}^{MAX}$]=[3,9]

Time 0 - 9

Waiting clients:(66,10,9) (219,10,5) (246,10,3) (272,10,3) (300,10,4) (411,10,5) (515,10,6) (521,10,6) (747,10,4) (908,10,8) (108,11,4) (120,11,5) (181,11,5) (258,11,8) (395,11,5) (456,11,7) (465,11,8) (668,11,7) (814,11,3) (856,11,8) (978,11,9) (992,11,8) (84,12,9) (111,12,5) (259,12,7) (313,12,7) (538,12,7) (670,12,4) (727,12,7) (728,12,9) (751,12,8) (775,12,7) (825,12,7) (844,12,5) (876,12,5) (940,12,3) (985,12,3) (62,13,4) (79,13,7) (779,13,8) (808,13,9) (818,13,7) (935,13,8) (269,14,4) (423,14,8) (742,14,9) (763,14,8) (795,14,5) (49,15,4) (170,15,4) (178,15,7) (180,15,7) (261,15,7) (271,15,8) (310,15,3) (329,15,4) (346,15,4) (347,15,8) (447,15,6) (556,15,5) (557,15,4) (565,15,9) (589,15,9) (633,15,6) (680,15,4) (145,16,5) (172,16,8) (270,16,5) (424,16,7) (580,16,9) (585,16,4) (688,16,9) (713,16,6) (729,16,3) (782,16,3) (793,16,5) (899,16,6) (971,16,6) (991,16,3) (101,17,5) (106,17,9) (202,17,4) (212,17,9) (221,17,3) (445,17,7) (505,17,4) (541,17,3) (652,17,4) (693,17,9) (731,17,7) (745,17,9) (953,17,8) (8,18,9) (91,18,3) (177,18,7) (205,18,8) (216,18,6) (257,18,6) (377,18,8) (426,18,9) (457,18,7) (523,18,6) (628,18,7) (650,18,7) (756,18,6) (765,18,5) (30,19,3) (44,19,3) (70,19,8) (223,19,6) (237,19,8) (399,19,3) (507,19,9) (762,19,3) (807,19,6) (880,19,5) (994,19,7) (38,20,6) (133,20,8) (155,20,4) (208,20,3) (308,20,8) (466,20,4) (648,20,3) (663,20,3) (789,20,7) (855,20,6) (941,20,7) (956,20,8) (16,21,7) (40,21,4) (85,21,7) (427,21,4) (449,21,4) (581,21,7) (684,21,6) (810,21,7) (882,21,6) (19,22,9) (22,22,9) (34,22,7) (134,22,7) (160,22,8) (163,22,3) (251,22,6) (254,22,5) (318,22,7) (365,22,9) (454,22,8) (654,22,6) (691,22,6) (750,22,6) (869,22,7) (977,22,4) (46,23,6) (245,23,8) (500,23,9) (532,23,8) (588,23,5) (647,23,7) (651,23,9) (679,23,5) (769,23,3) (783,23,7) (943,23,7) (7,24,3) (82,24,9) (167,24,4) (176,24,6) (190,24,4) (234,24,3) (304,24,7) (312,24,7) (362,24,5) (474,24,9) (627,24,3) (824,24,4) (862,24,5) (967,24,6) (135,25,5) (189,25,4) (333,25,6) (436,25,6) (566,25,5) (623,25,4) (635,25,5) (643,25,3) (722,25,4) (879,25,4) (910,25,7) (923,25,5) (942,25,4) (976,25,3) (137,26,5) (294,26,9) (314,26,8) (366,26,9) (513,26,3) (776,26,5) (841,26,7) (897,26,4) (104,27,3) (199,27,8) (475,27,8) (512,27,6) (780,27,4) (920,27,6) (968,27,8) (74,28,5) (263,28,7) (268,28,6) (437,28,4) (620,28,4) (826,28,9) (827,28,8) (35,29,6) (41,29,9) (55,29,5) (76,29,5) (119,29,6) (327,29,8) (381,29,8) (551,29,4) (640,29,9) (657,29,6) (671,29,5) (736,29,6) (743,29,6) (919,29,8) (921,29,6) (958,29,3) (97,30,5) (211,30,3) (215,30,3) (235,30,9) (316,30,9) (324,30,6) (341,30,9) (385,30,4) (391,30,6) (413,30,5) (444,30,8) (497,30,9) (563,30,8) (903,30,9) (973,30,6) (993,30,3) (60,31,4) (112,31,4) (247,31,4) (359,31,4) (415,31,8) (509,31,7) (572,31,4) (602,31,3) (605,31,4) (667,31,6) (681,31,9) (701,31,6) (836,31,3) (916,31,6) (986,31,8) (122,32,5) (288,32,7) (467,32,5) (477,32,9) (494,32,8) (584,32,8) (698,32,9) (749,32,7) (999,32,9) (4,33,4) (51,33,4) (98,33,6) (188,33,7) (404,33,4) (414,33,6) (453,33,9) (525,33,3) (561,33,9) (592,33,9) (787,33,8) (37,34,4) (266,34,7) (297,34,6) (298,34,8) (431,34,9) (502,34,4) (547,34,6) (834,34,4) (984,34,7) (396,35,5) (428,35,3) (508,35,8) (534,35,5) (617,35,8) (792,35,7) (854,35,4) (295,36,7) (311,36,5) (402,36,7) (553,36,9) (573,36,3) (624,36,7) (661,36,5) (758,36,7) (843,36,8) (851,36,9) (129,37,4) (131,37,8) (146,37,9) (275,37,9) (389,37,5) (527,37,7) (542,37,9) (548,37,7) (567,37,9) (874,37,6) (950,37,9) (983,37,3) (67,38,4) (240,38,7) (380,38,6) (615,38,5) (700,38,7) (786,38,3) (797,38,6) (839,38,7) (864,38,8) (867,38,6) (922,38,8) (933,38,9) (963,38,7) (996,38,6) (997,38,8) (47,39,3) (68,39,3) (353,39,8) (367,39,6) (393,39,7) (408,39,3) (510,39,7) (529,39,6) (571,39,4) (676,39,6) (704,39,5) (820,39,4) (885,39,7) (917,39,7) (45,40,8) (128,40,5) (267,40,8) (299,40,5) (401,40,7) (476,40,7) (483,40,8) (865,40,9) (959,40,5) (127,41,9) (230,41,5) (248,41,6) (296,41,3) (349,41,4) (516,41,7) (520,41,5) (708,41,4) (754,41,7) (174,42,3) (197,42,9) (274,42,4) (283,42,9) (317,42,7) (639,42,7) (709,42,4) (760,42,9) (778,42,4) (817,42,4) (875,42,5) (998,42,4) (90,43,5) (185,43,8) (322,43,7) (373,43,8) (434,43,9) (501,43,4) (519,43,4) (658,43,7) (790,43,7) (798,43,6) (883,43,8) (926,43,3) (927,43,5) (936,43,4) (6,44,4) (179,44,3) (201,44,9) (305,44,6) (320,44,3) (321,44,6) (328,44,9) (478,44,9) (649,44,6) (669,44,8) (813,44,4) (29,45,3) (118,45,5) (141,45,6) (144,45,7) (157,45,3) (206,45,4) (637,45,3) (714,45,5) (800,45,6) (969,45,5) (988,45,6) (43,46,6) (107,46,5) (109,46,9) (403,46,4) (440,46,5) (540,46,9) (575,46,6) (730,46,7) (791,46,6) (828,46,3) (901,46,8) (924,46,9) (59,47,7) (183,47,5) (292,47,8) (348,47,6) (686,47,4) (858,47,9) (895,47,5) (3,48,4) (12,48,6) (301,48,7) (672,48,9) (678,49,4) (705,49,6) (707,49,8) (955,49,8) (193,50,7) (394,50,6) (438,50,5) (473,50,9) (554,50,5) (568,50,3) (625,50,9) (677,50,9) (689,50,3) (812,50,4) (829,50,5) (840,50,6) (966,50,8) (162,51,5) (165,51,8) (231,51,5) (459,51,3) (504,51,8) (511,51,6) (518,51,8) (641,51,6) (930,51,8) (13,52,3) (73,52,6) (100,52,6) (114,52,7) (598,52,3) (690,52,5) (702,52,8) (741,52,9) (884,52,9) (125,53,8) (204,53,8) (207,53,3) (289,53,6) (462,53,8) (496,53,4)

```
(498,53,9) (524,53,7) (552,53,8) (601,53,4) (673,53,3) (734,53,4) (748,53,3) (26,54,5) (56,54,9) (72,54,8) (96,54,7) (102,54,4) (239,54,6)
(319,54,9) (343,54,9) (420,54,5) (485,54,9) (555,54,7) (560,54,5) (699,54,3) (873,54,4) (945,54,5) (220,55,4) (228,55,8) (330,55,3)
(352,55,7) (368,55,5) (384,55,9) (450,55,7) (492,55,8) (577,55,9) (599,55,9) (604,55,9) (773,55,3) (980,55,8) (64,56,3) (99,56,3) (117,56,3)
(217,56,4) (253,56,5) (382,56,9) (433,56,4) (435,56,6) (484,56,5) (710,56,8) (848,56,7) (948,56,7) (14,57,8) (50,57,3) (116,57,8) (564,57,9)
(596,57,8) (69,58,7) (159,58,5) (171,58,7) (249,58,8) (276,58,3) (291,58,9) (340,58,7) (421,58,6) (451,58,4) (479,58,6) (802,58,8)
(842,58,4) (850,58,7) (866,58,8) (1000,58,9) (265,59,9) (290,59,4) (370,59,3) (626,59,4) (716,59,8) (739,59,8) (894,59,6) (5,60,5)
(166,60,3) (600,60,3) (695,60,4) (816,60,8) (845,60,3) (868,60,5) (889,60,5) (909,60,5) (939,60,3) (946,60,3) (57,61,6) (77,61,3) (200,61,5)
(417,61,4) (448,61,9) (591,61,5) (752,61,5) (806,61,5) (837,61,8) (11,62,4) (36,62,4) (88,62,7) (154,62,4) (282,62,3) (425,62,8) (544,62,4)
(735,62,4) (768,62,5) (957,62,3) (192,63,9) (242,63,8) (354,63,9) (410,63,9) (606,63,3) (646,63,6) (815,63,8) (859,63,8) (877,63,3)
(121,64,7) (126,64,3) (150,64,4) (323,64,6) (472,64,5) (578,64,5) (771,64,7) (886,64,4) (914,64,4) (928,64,6) (934,64,5) (148,65,3)
(149,65,5) (195,65,4) (196,65,6) (280,65,8) (355,65,5) (374,65,8) (378,65,9) (429,65,4) (443,65,4) (493,65,3) (574,65,7) (582,65,9)
(696,65,6) (823,65,6) (10,66,8) (63,66,5) (86,66,3) (186,66,7) (531,66,4) (576,66,5) (610,66,5) (723,66,7) (784,66,4) (821,66,3) (861,66,6)
(870,66,8) (888,66,6) (911,66,7) (918,66,4) (33,67,6) (65,67,5) (103,67,8) (140,67,3) (250,67,5) (255,67,8) (461,67,7) (666,67,5) (674,67,3)
(737,67,7) (887,67,7) (932,67,5) (961,67,6) (124,68,5) (175,68,3) (233,68,4) (543,68,9) (622,68,8) (863,68,8) (872,68,6) (975,68,7)
(81,69,8) (244,69,5) (332,69,5) (357,69,8) (361,69,9) (539,69,3) (611,69,3) (613,69,6) (664,69,5) (665,69,6) (738,69,6) (801,69,3)
(902,69,6) (937,69,9) (944,69,5) (970,69,6) (286,70,5) (309,70,8) (376,70,9) (379,70,4) (439,70,8) (463,70,3) (533,70,7) (618,70,8)
(636,70,6) (718,70,6) (847,70,9) (987,70,6) (105,71,4) (184,71,5) (279,71,8) (281,71,5) (345,71,8) (458,71,8) (583,71,8) (630,71,7)
(900,71,8) (915,71,7) (153,72,8) (164,72,9) (306,72,4) (549,72,4) (229,73,4) (232,73,4) (706,73,8) (719,73,3) (805,73,8) (965,73,3)
(182,74,7) (326,74,6) (528,74,5) (597,74,9) (632,74,9) (715,74,8) (964,74,5) (979,74,8) (990,74,6) (20,75,4) (42,75,9) (71,75,7) (173,75,5)
(252,75,6) (264,75,9) (278,75,9) (339,75,5) (405,75,6) (482,75,5) (550,75,3) (595,75,3) (619,75,7) (744,75,5) (759,75,8) (803,75,5)
(32,76,9) (227,76,3) (302,76,6) (400,76,4) (621,76,4) (724,76,9) (853,76,3) (947,76,9) (58,77,6) (303,77,8) (334,77,4) (471,77,9) (594,77,8)
(712,77,9) (857,77,4) (907,77,3) (9,78,3) (28,78,3) (78,78,8) (169,78,3) (307,78,8) (388,78,3) (481,78,3) (562,78,5) (781,78,7) (904,78,8)
(123,79,6) (136,79,4) (151,79,3) (358,79,9) (579,79,6) (645,79,7) (687,79,5) (720,79,6) (833,79,7) (48,80,8) (158,80,7) (168,80,7)
(187,80,3) (225,80,4) (238,80,3) (241,80,8) (375,80,7) (464,80,8) (526,80,9) (660,80,7) (767,80,9) (819,80,5) (974,80,6) (92,81,6) (95,81,8)
(213,81,9) (256,81,7) (387,81,6) (603,81,5) (609,81,5) (659,81,6) (675,81,4) (717,81,4) (721,81,3) (881,81,3) (951,81,7) (147,82,6)
(243,82,4) (372,82,7) (446,82,7) (480,82,6) (634,82,7) (732,82,7) (755,82,5) (770,82,5) (23,83,6) (93,83,3) (209,83,3) (369,83,4) (397,83,7)
(398,83,3) (469,83,7) (503,83,8) (740,83,5) (746,83,9) (785,83,6) (835,83,7) (906,83,4) (24,84,3) (142,84,4) (218,84,9) (363,84,4)
(470,84,3) (537,84,3) (614,84,4) (683,84,5) (766,84,7) (799,84,4) (809,84,6) (912,84,7) (929,84,3) (949,84,9) (52,85,6) (285,85,5)
(386,85,6) (392,85,8) (488,85,6) (522,85,7) (559,85,5) (612,85,9) (777,85,8) (846,85,8) (905,85,8) (952,85,8) (981,85,7) (17,86,4)
(113,86,7) (198,86,3) (210,86,5) (226,86,9) (236,86,5) (287,86,3) (335,86,5) (406,86,6) (486,86,7) (491,86,4) (655,86,7) (692,86,9)
(694,86,6) (733,86,9) (61,87,3) (115,87,4) (138,87,5) (262,87,6) (336,87,4) (530,87,4) (703,87,9) (832,87,5) (94,88,6) (277,88,5) (325,88,3)
(338,88,4) (455,88,5) (460,88,3) (545,88,4) (570,88,5) (631,88,4) (697,88,7) (711,88,3) (753,88,3) (794,88,5) (898,88,8) (27,89,9) (75,89,3)
(132,89,7) (224,89,5) (331,89,7) (360,89,5) (419,89,6) (441,89,8) (638,89,5) (682,89,8) (838,89,9) (891,89,7) (25,90,7) (89,90,8) (214,90,8)
(452,90,5) (468,90,4) (487,90,4) (593,90,9) (608,90,9) (725,90,3) (764,90,7) (788,90,4) (849,90,4) (960,90,5) (83,91,4) (161,91,3)
(293,91,9) (342,91,4) (351,91,3) (383,91,4) (499,91,9) (558,91,6) (804,91,7) (852,91,9) (938,91,9) (31,92,5) (39,92,3) (662,92,6) (726,92,7)
(860,92,8) (896,92,5) (203,93,8) (273,93,3) (356,93,6) (430,93,8) (514,93,3) (536,93,8) (587,93,4) (653,93,3) (53,94,3) (139,94,8)
(350,94,7) (409,94,3) (569,94,4) (616,94,7) (629,94,8) (796,94,6) (931,94,6) (222,95,5) (344,95,4) (364,95,8) (390,95,7) (442,95,3)
(490,95,7) (644,95,7) (656,95,6) (774,95,7) (822,95,5) (830,95,3) (890,95,8) (972,95,9) (87,96,5) (110,96,6) (130,96,7) (143,96,6)
(156,96,5) (194,96,5) (489,96,9) (506,96,8) (517,96,9) (590,96,9) (757,96,4) (811,96,9) (871,96,3) (892,96,5) (989,96,4) (2,97,8) (21,97,7)
(152,97,4) (416,97,7) (535,97,9) (546,97,8) (831,97,5) (893,97,4) (954,97,4) (982,97,4) (80,98,9) (191,98,9) (418,98,4) (422,98,6)
(432,98,3) (607,98,5) (685,98,6) (761,98,7) (878,98,4) (925,98,8) (995,98,4) (1,99,7) (15,99,7) (54,99,7) (284,99,8) (315,99,9) (337,99,4)
(371,99,4) (495,99,4) (642,99,6) (913,99,7) (18,100,5) (260,100,4) (407,100,5) (412,100,4) (586,100,9) (772,100,4) (962,100,7)

Queue 1:empty

Queue 2:empty

Queue 3:empty

Queue 4:empty

Queue 5:empty

Queue 6:empty

Queue 7:empty

Queue 8:empty

Queue 9:empty
Queue 10:empty

Queue 11:empty

Queue 12:empty

Queue 13:empty

Queue 14:empty

Queue 15:empty

Queue 16:empty

Queue 17:empty

Queue 18:empty

Queue 19:empty

Queue 20:empty


Time 10

Waiting clients:(108,11,4) (120,11,5) (181,11,5) (258,11,8) (395,11,5) (456,11,7) (465,11,8) (668,11,7) (814,11,3) (856,11,8) (978,11,9)
(992,11,8) (84,12,9) (111,12,5) (259,12,7) (313,12,7) (538,12,7) (670,12,4) (727,12,7) (728,12,9) (751,12,8) (775,12,7) (825,12,7)
(844,12,5) (876,12,5) (940,12,3) (985,12,3) (62,13,4) (79,13,7) (779,13,8) (808,13,9) (818,13,7) (935,13,8) (269,14,4) (423,14,8) (742,14,9)
(763,14,8) (795,14,5) (49,15,4) (170,15,4) (178,15,7) (180,15,7) (261,15,7) (271,15,8) (310,15,3) (329,15,4) (346,15,4) (347,15,8)
(447,15,6) (556,15,5) (557,15,4) (565,15,9) (589,15,9) (633,15,6) (680,15,4) (145,16,5) (172,16,8) (270,16,5) (424,16,7) (580,16,9)
(585,16,4) (688,16,9) (713,16,6) (729,16,3) (782,16,3) (793,16,5) (899,16,6) (971,16,6) (991,16,3) (101,17,5) (106,17,9) (202,17,4)
(212,17,9) (221,17,3) (445,17,7) (505,17,4) (541,17,3) (652,17,4) (693,17,9) (731,17,7) (745,17,9) (953,17,8) (8,18,9) (91,18,3) (177,18,7)
(205,18,8) (216,18,6) (257,18,6) (377,18,8) (426,18,9) (457,18,7) (523,18,6) (628,18,7) (650,18,7) (756,18,6) (765,18,5) (30,19,3) (44,19,3)
(70,19,8) (223,19,6) (237,19,8) (399,19,3) (507,19,9) (762,19,3) (807,19,6) (880,19,5) (994,19,7) (38,20,6) (133,20,8) (155,20,4) (208,20,3)
(308,20,8) (466,20,4) (648,20,3) (663,20,3) (789,20,7) (855,20,6) (941,20,7) (956,20,8) (16,21,7) (40,21,4) (85,21,7) (427,21,4) (449,21,4)
(581,21,7) (684,21,6) (810,21,7) (882,21,6) (19,22,9) (22,22,9) (34,22,7) (134,22,7) (160,22,8) (163,22,3) (251,22,6) (254,22,5) (318,22,7)
(365,22,9) (454,22,8) (654,22,6) (691,22,6) (750,22,6) (869,22,7) (977,22,4) (46,23,6) (245,23,8) (500,23,9) (532,23,8) (588,23,5)
(647,23,7) (651,23,9) (679,23,5) (769,23,3) (783,23,7) (943,23,7) (7,24,3) (82,24,9) (167,24,4) (176,24,6) (190,24,4) (234,24,3) (304,24,7)
(312,24,7) (362,24,5) (474,24,9) (627,24,3) (824,24,4) (862,24,5) (967,24,6) (135,25,5) (189,25,4) (333,25,6) (436,25,6) (566,25,5)
(623,25,4) (635,25,5) (643,25,3) (722,25,4) (879,25,4) (910,25,7) (923,25,5) (942,25,4) (976,25,3) (137,26,5) (294,26,9) (314,26,8)
(366,26,9) (513,26,3) (776,26,5) (841,26,7) (897,26,4) (104,27,3) (199,27,8) (475,27,8) (512,27,6) (780,27,4) (920,27,6) (968,27,8)
(74,28,5) (263,28,7) (268,28,6) (437,28,4) (620,28,4) (826,28,9) (827,28,8) (35,29,6) (41,29,9) (55,29,5) (76,29,5) (119,29,6) (327,29,8)
(381,29,8) (551,29,4) (640,29,9) (657,29,6) (671,29,5) (736,29,6) (743,29,6) (919,29,8) (921,29,6) (958,29,3) (97,30,5) (211,30,3)
(215,30,3) (235,30,9) (316,30,9) (324,30,6) (341,30,9) (385,30,4) (391,30,6) (413,30,5) (444,30,8) (497,30,9) (563,30,8) (903,30,9)
(973,30,6) (993,30,3) (60,31,4) (112,31,4) (247,31,4) (359,31,4) (415,31,8) (509,31,7) (572,31,4) (602,31,3) (605,31,4) (667,31,6)
(681,31,9) (701,31,6) (836,31,3) (916,31,6) (986,31,8) (122,32,5) (288,32,7) (467,32,5) (477,32,9) (494,32,8) (584,32,8) (698,32,9)
(749,32,7) (999,32,9) (4,33,4) (51,33,4) (98,33,6) (188,33,7) (404,33,4) (414,33,6) (453,33,9) (525,33,3) (561,33,9) (592,33,9) (787,33,8)
(37,34,4) (266,34,7) (297,34,6) (298,34,8) (431,34,9) (502,34,4) (547,34,6) (834,34,4) (984,34,7) (396,35,5) (428,35,3) (508,35,8)
(534,35,5) (617,35,8) (792,35,7) (854,35,4) (295,36,7) (311,36,5) (402,36,7) (553,36,9) (573,36,3) (624,36,7) (661,36,5) (758,36,7)
(843,36,8) (851,36,9) (129,37,4) (131,37,8) (146,37,9) (275,37,9) (389,37,5) (527,37,7) (542,37,9) (548,37,7) (567,37,9) (874,37,6)
(950,37,9) (983,37,3) (67,38,4) (240,38,7) (380,38,6) (615,38,5) (700,38,7) (786,38,3) (797,38,6) (839,38,7) (864,38,8) (867,38,6)
(922,38,8) (933,38,9) (963,38,7) (996,38,6) (997,38,8) (47,39,3) (68,39,3) (353,39,8) (367,39,6) (393,39,7) (408,39,3) (510,39,7) (529,39,6)
(571,39,4) (676,39,6) (704,39,5) (820,39,4) (885,39,7) (917,39,7) (45,40,8) (128,40,5) (267,40,8) (299,40,5) (401,40,7) (476,40,7)
(483,40,8) (865,40,9) (959,40,5) (127,41,9) (230,41,5) (248,41,6) (296,41,3) (349,41,4) (516,41,7) (520,41,5) (708,41,4) (754,41,7)
(174,42,3) (197,42,9) (274,42,4) (283,42,9) (317,42,7) (639,42,7) (709,42,4) (760,42,9) (778,42,4) (817,42,4) (875,42,5) (998,42,4)
(90,43,5) (185,43,8) (322,43,7) (373,43,8) (434,43,9) (501,43,4) (519,43,4) (658,43,7) (790,43,7) (798,43,6) (883,43,8) (926,43,3)
(927,43,5) (936,43,4) (6,44,4) (179,44,3) (201,44,9) (305,44,6) (320,44,3) (321,44,6) (328,44,9) (478,44,9) (649,44,6) (669,44,8) (813,44,4)
(29,45,3) (118,45,5) (141,45,6) (144,45,7) (157,45,3) (206,45,4) (637,45,3) (714,45,5) (800,45,6) (969,45,5) (988,45,6) (43,46,6) (107,46,5)
(109,46,9) (403,46,4) (440,46,5) (540,46,9) (575,46,6) (730,46,7) (791,46,6) (828,46,3) (901,46,8) (924,46,9) (59,47,7) (183,47,5)
(292,47,8) (348,47,6) (686,47,4) (858,47,9) (895,47,5) (3,48,4) (12,48,6) (301,48,7) (672,48,9) (678,49,4) (705,49,6) (707,49,8) (955,49,8)
(193,50,7) (394,50,6) (438,50,5) (473,50,9) (554,50,5) (568,50,3) (625,50,9) (677,50,9) (689,50,3) (812,50,4) (829,50,5) (840,50,6)
```

```
(966,50,8) (162,51,5) (165,51,8) (231,51,5) (459,51,3) (504,51,8) (511,51,6) (518,51,8) (641,51,6) (930,51,8) (13,52,3) (73,52,6) (100,52,6)
(114,52,7) (598,52,3) (690,52,5) (702,52,8) (741,52,9) (884,52,9) (125,53,8) (204,53,8) (207,53,3) (289,53,6) (462,53,8) (496,53,4)
(498,53,9) (524,53,7) (552,53,8) (601,53,4) (673,53,3) (734,53,4) (748,53,3) (26,54,5) (56,54,9) (72,54,8) (96,54,7) (102,54,4) (239,54,6)
(319,54,9) (343,54,9) (420,54,5) (485,54,9) (555,54,7) (560,54,5) (699,54,3) (873,54,4) (945,54,5) (220,55,4) (228,55,8) (330,55,3)
(352,55,7) (368,55,5) (384,55,9) (450,55,7) (492,55,8) (577,55,9) (599,55,9) (604,55,9) (773,55,3) (980,55,8) (64,56,3) (99,56,3) (117,56,3)
(217,56,4) (253,56,5) (382,56,9) (433,56,4) (435,56,6) (484,56,5) (710,56,8) (848,56,7) (948,56,7) (14,57,8) (50,57,3) (116,57,8) (564,57,9)
(596,57,8) (69,58,7) (159,58,5) (171,58,7) (249,58,8) (276,58,3) (291,58,9) (340,58,7) (421,58,6) (451,58,4) (479,58,6) (802,58,8)
(842,58,4) (850,58,7) (866,58,8) (1000,58,9) (265,59,9) (290,59,4) (370,59,3) (626,59,4) (716,59,8) (739,59,8) (894,59,6) (5,60,5)
(166,60,3) (600,60,3) (695,60,4) (816,60,8) (845,60,3) (868,60,5) (889,60,5) (909,60,5) (939,60,3) (946,60,3) (57,61,6) (77,61,3) (200,61,5)
(417,61,4) (448,61,9) (591,61,5) (752,61,5) (806,61,5) (837,61,8) (11,62,4) (36,62,4) (88,62,7) (154,62,4) (282,62,3) (425,62,8) (544,62,4)
(735,62,4) (768,62,5) (957,62,3) (192,63,9) (242,63,8) (354,63,9) (410,63,9) (606,63,3) (646,63,6) (815,63,8) (859,63,8) (877,63,3)
(121,64,7) (126,64,3) (150,64,4) (323,64,6) (472,64,5) (578,64,5) (771,64,7) (886,64,4) (914,64,4) (928,64,6) (934,64,5) (148,65,3)
(149,65,5) (195,65,4) (196,65,6) (280,65,8) (355,65,5) (374,65,8) (378,65,9) (429,65,4) (443,65,4) (493,65,3) (574,65,7) (582,65,9)
(696,65,6) (823,65,6) (10,66,8) (63,66,5) (86,66,3) (186,66,7) (531,66,4) (576,66,5) (610,66,5) (723,66,7) (784,66,4) (821,66,3) (861,66,6)
(870,66,8) (888,66,6) (911,66,7) (918,66,4) (33,67,6) (65,67,5) (103,67,8) (140,67,3) (250,67,5) (255,67,8) (461,67,7) (666,67,5) (674,67,3)
(737,67,7) (887,67,7) (932,67,5) (961,67,6) (124,68,5) (175,68,3) (233,68,4) (543,68,9) (622,68,8) (863,68,8) (872,68,6) (975,68,7)
(81,69,8) (244,69,5) (332,69,5) (357,69,8) (361,69,9) (539,69,3) (611,69,3) (613,69,6) (664,69,5) (665,69,6) (738,69,6) (801,69,3)
(902,69,6) (937,69,9) (944,69,5) (970,69,6) (286,70,5) (309,70,8) (376,70,9) (379,70,4) (439,70,8) (463,70,3) (533,70,7) (618,70,8)
(636,70,6) (718,70,6) (847,70,9) (987,70,6) (105,71,4) (184,71,5) (279,71,8) (281,71,5) (345,71,8) (458,71,8) (583,71,8) (630,71,7)
(900,71,8) (915,71,7) (153,72,8) (164,72,9) (306,72,4) (549,72,4) (229,73,4) (232,73,4) (706,73,8) (719,73,3) (805,73,8) (965,73,3)
(182,74,7) (326,74,6) (528,74,5) (597,74,9) (632,74,9) (715,74,8) (964,74,5) (979,74,8) (990,74,6) (20,75,4) (42,75,9) (71,75,7) (173,75,5)
(252,75,6) (264,75,9) (278,75,9) (339,75,5) (405,75,6) (482,75,5) (550,75,3) (595,75,3) (619,75,7) (744,75,5) (759,75,8) (803,75,5)
(32,76,9) (227,76,3) (302,76,6) (400,76,4) (621,76,4) (724,76,9) (853,76,3) (947,76,9) (58,77,6) (303,77,8) (334,77,4) (471,77,9) (594,77,8)
(712,77,9) (857,77,4) (907,77,3) (9,78,3) (28,78,3) (78,78,8) (169,78,3) (307,78,8) (388,78,3) (481,78,3) (562,78,5) (781,78,7) (904,78,8)
(123,79,6) (136,79,4) (151,79,3) (358,79,9) (579,79,6) (645,79,7) (687,79,5) (720,79,6) (833,79,7) (48,80,8) (158,80,7) (168,80,7)
(187,80,3) (225,80,4) (238,80,3) (241,80,8) (375,80,7) (464,80,8) (526,80,9) (660,80,7) (767,80,9) (819,80,5) (974,80,6) (92,81,6) (95,81,8)
(213,81,9) (256,81,7) (387,81,6) (603,81,5) (609,81,5) (659,81,6) (675,81,4) (717,81,4) (721,81,3) (881,81,3) (951,81,7) (147,82,6)
(243,82,4) (372,82,7) (446,82,7) (480,82,6) (634,82,7) (732,82,7) (755,82,5) (770,82,5) (23,83,6) (93,83,3) (209,83,3) (369,83,4) (397,83,7)
(398,83,3) (469,83,7) (503,83,8) (740,83,5) (746,83,9) (785,83,6) (835,83,7) (906,83,4) (24,84,3) (142,84,4) (218,84,9) (363,84,4)
(470,84,3) (537,84,3) (614,84,4) (683,84,5) (766,84,7) (799,84,4) (809,84,6) (912,84,7) (929,84,3) (949,84,9) (52,85,6) (285,85,5)
(386,85,6) (392,85,8) (488,85,6) (522,85,7) (559,85,5) (612,85,9) (777,85,8) (846,85,8) (905,85,8) (952,85,8) (981,85,7) (17,86,4)
(113,86,7) (198,86,3) (210,86,5) (226,86,9) (236,86,5) (287,86,3) (335,86,5) (406,86,6) (486,86,7) (491,86,4) (655,86,7) (692,86,9)
(694,86,6) (733,86,9) (61,87,3) (115,87,4) (138,87,5) (262,87,6) (336,87,4) (530,87,4) (703,87,9) (832,87,5) (94,88,6) (277,88,5) (325,88,3)
(338,88,4) (455,88,5) (460,88,3) (545,88,4) (570,88,5) (631,88,4) (697,88,7) (711,88,3) (753,88,3) (794,88,5) (898,88,8) (27,89,9) (75,89,3)
(132,89,7) (224,89,5) (331,89,7) (360,89,5) (419,89,6) (441,89,8) (638,89,5) (682,89,8) (838,89,9) (891,89,7) (25,90,7) (89,90,8) (214,90,8)
(452,90,5) (468,90,4) (487,90,4) (593,90,9) (608,90,9) (725,90,3) (764,90,7) (788,90,4) (849,90,4) (960,90,5) (83,91,4) (161,91,3)
(293,91,9) (342,91,4) (351,91,3) (383,91,4) (499,91,9) (558,91,6) (804,91,7) (852,91,9) (938,91,9) (31,92,5) (39,92,3) (662,92,6) (726,92,7)
(860,92,8) (896,92,5) (203,93,8) (273,93,3) (356,93,6) (430,93,8) (514,93,3) (536,93,8) (587,93,4) (653,93,3) (53,94,3) (139,94,8)
(350,94,7) (409,94,3) (569,94,4) (616,94,7) (629,94,8) (796,94,6) (931,94,6) (222,95,5) (344,95,4) (364,95,8) (390,95,7) (442,95,3)
(490,95,7) (644,95,7) (656,95,6) (774,95,7) (822,95,5) (830,95,3) (890,95,8) (972,95,9) (87,96,5) (110,96,6) (130,96,7) (143,96,6)
(156,96,5) (194,96,5) (489,96,9) (506,96,8) (517,96,9) (590,96,9) (757,96,4) (811,96,9) (871,96,3) (892,96,5) (989,96,4) (2,97,8) (21,97,7)
(152,97,4) (416,97,7) (535,97,9) (546,97,8) (831,97,5) (893,97,4) (954,97,4) (982,97,4) (80,98,9) (191,98,9) (418,98,4) (422,98,6)
(432,98,3) (607,98,5) (685,98,6) (761,98,7) (878,98,4) (925,98,8) (995,98,4) (1,99,7) (15,99,7) (54,99,7) (284,99,8) (315,99,9) (337,99,4)
(371,99,4) (495,99,4) (642,99,6) (913,99,7) (18,100,5) (260,100,4) (407,100,5) (412,100,4) (586,100,9) (772,100,4) (962,100,7)

Queue 1:(66,10,9)

Queue 2:(219,10,5)

Queue 3:(246,10,3)

Queue 4:(272,10,3)

Queue 5:(300,10,4)

Queue 6:(411,10,5)

Queue 7:(515,10,6)

Queue 8:(521,10,6)
Queue 9:(747,10,4)

Queue 10:(908,10,8)

Queue 11:empty

Queue 12:empty

Queue 13:empty

Queue 14:empty

Queue 15:empty

Queue 16:empty

Queue 17:empty

Queue 18:empty

Queue 19:empty

Queue 20:empty


.................................................................................................................................................................................................

Time 200

Waiting clients:

Queue 1:(255,67,8) (902,69,6) (458,71,8) (71,75,7) (621,76,4) (388,78,3) (645,79,7) (609,81,5) (755,82,5) (363,84,4) (559,85,5) (406,86,6)
(545,88,4) (638,89,5) (788,90,4) (31,92,5) (409,94,3) (656,95,6) (871,96,3) (893,97,4) (878,98,4) (407,100,5)

Queue 2:(461,67,7) (664,69,5) (718,70,6) (229,73,4) (173,75,5) (32,76,9) (687,79,5) (819,80,5) (951,81,7) (470,84,3) (285,85,5) (210,86,5)
(703,87,9) (593,90,9) (587,93,4) (364,95,8) (21,97,7) (284,99,8)

Queue 3:(666,67,5) (332,69,5) (379,70,4) (583,71,8) (252,75,6) (302,76,6) (562,78,5) (241,80,8) (243,82,4) (785,83,6) (612,85,9) (832,87,5)
(331,89,7) (161,91,3) (39,92,3) (653,93,3) (796,94,6) (506,96,8) (685,98,6) (772,100,4)

Queue 4:(10,66,2) (961,67,6) (937,69,9) (153,72,8) (339,75,5) (58,77,6) (720,79,6) (256,81,7) (397,83,7) (386,85,6) (486,86,7) (631,88,4)
(89,90,8) (662,92,6) (931,94,6) (517,96,9) (925,98,8)

Queue 5:(870,66,5) (357,69,8) (105,71,4) (164,72,9) (595,75,3) (724,76,9) (158,80,7) (717,81,4) (398,83,3) (24,84,3) (949,84,9) (733,86,9)
(214,90,8) (726,92,7) (390,95,7) (892,96,5) (432,98,3) (315,99,9)

Queue 6:(582,65,2) (124,68,5) (665,69,6) (184,71,5) (232,73,4) (264,75,9) (857,77,4) (358,79,9) (675,81,4) (770,82,5) (537,84,3) (392,85,8)
(61,87,3) (460,88,3) (898,88,8) (293,91,9) (222,95,5) (143,96,6) (954,97,4) (995,98,4) (412,100,4)

Queue 7:(33,67,5) (361,69,9) (630,71,7) (528,74,5) (619,75,7) (481,78,3) (833,79,7) (659,81,6) (469,83,7) (488,85,6) (491,86,4) (94,88,6)
(682,89,8) (938,91,9) (774,95,7) (152,97,4) (607,98,5) (913,99,7)

Queue 8:(65,67,4) (975,68,7) (533,70,7) (706,73,8) (744,75,5) (907,77,3) (781,78,7) (974,80,6) (372,82,7) (683,84,5) (17,86,4) (655,86,7)
(27,89,9) (383,91,4) (514,93,3) (569,94,4) (890,95,8) (982,97,4) (1,99,7)

Queue 9:(674,67,3) (863,68,8) (618,70,8) (326,74,6) (405,75,6) (9,78,3) (904,78,8) (387,81,6) (23,83,6) (766,84,7) (226,86,9) (75,89,3)
(452,90,5) (342,91,4) (356,93,6) (442,95,3) (87,96,5) (416,97,7) (337,99,4) (962,100,7)

Queue 10:(888,66,3) (872,68,6) (286,70,5) (900,71,8) (278,75,9) (28,78,3) (123,79,6) (660,80,7) (446,82,7) (799,84,4) (952,85,8) (277,88,5)
(360,89,5) (725,90,3) (499,91,9) (490,95,7) (989,96,4) (418,98,4) (371,99,4)
```

```
Queue 11:(911,66,4) (81,69,8) (847,70,9) (597,74,9) (303,77,8) (168,80,7) (721,81,3) (93,83,3) (835,83,7) (981,85,7) (138,87,5) (132,89,7)
(849,90,4) (860,92,8) (822,95,5) (757,96,4) (80,98,9)

Queue 12:(737,67,7) (738,69,6) (279,71,8) (632,74,9) (334,77,4) (136,79,4) (187,80,3) (92,81,6) (480,82,6) (614,84,4) (777,85,8) (262,87,6)
(419,89,6) (960,90,5) (430,93,8) (972,95,9) (422,98,6) (18,100,5)

Queue 13:(186,66,1) (887,67,7) (944,69,5) (281,71,5) (719,73,3) (715,74,8) (853,76,3) (169,78,3) (579,79,6) (95,81,8) (503,83,8) (846,85,8)
(336,87,4) (697,88,7) (764,90,7) (536,93,8) (110,96,6) (535,97,9) (586,100,9)

Queue 14:(378,65,1) (932,67,5) (611,69,3) (309,70,8) (306,72,4) (964,74,5) (759,75,8) (151,79,3) (48,80,8) (881,81,3) (209,83,3) (906,83,4)
(52,85,6) (236,86,5) (325,88,3) (711,88,3) (838,89,9) (896,92,5) (616,94,7) (590,96,9) (15,99,7)

Queue 15:(103,67,7) (801,69,3) (439,70,8) (805,73,8) (803,75,5) (78,78,8) (375,80,7) (147,82,6) (142,84,4) (522,85,7) (692,86,9) (891,89,7)
(558,91,6) (350,94,7) (156,96,5) (546,97,8) (260,100,4)

Queue 16:(723,66,2) (175,68,3) (539,69,3) (970,69,6) (915,71,7) (979,74,8) (947,76,9) (225,80,4) (603,81,5) (634,82,7) (809,84,6)
(198,86,3) (694,86,6) (753,88,3) (25,90,7) (804,91,7) (629,94,8) (811,96,9) (495,99,4)

Queue 17:(861,66,2) (233,68,4) (613,69,6) (987,70,6) (965,73,3) (990,74,6) (227,76,3) (471,77,9) (464,80,8) (732,82,7) (912,84,7)
(287,86,3) (115,87,4) (570,88,5) (468,90,4) (83,91,4) (203,93,8) (830,95,3) (194,96,5) (831,97,5) (54,99,7)

Queue 18:(918,66,2) (543,68,9) (636,70,6) (549,72,4) (20,75,4) (482,75,5) (594,77,8) (238,80,3) (213,81,9) (746,83,9) (113,86,7) (338,88,4)
(224,89,5) (608,90,9) (53,94,3) (344,95,4) (130,96,7) (191,98,9)

Queue 19:(140,67,2) (622,68,8) (463,70,3) (345,71,8) (42,75,9) (712,77,9) (526,80,9) (369,83,4) (218,84,9) (335,86,5) (455,88,5) (441,89,8)
(852,91,9) (644,95,7) (2,97,8) (642,99,6)

Queue 20:(250,67,4) (244,69,5) (376,70,9) (182,74,7) (550,75,3) (400,76,4) (307,78,8) (767,80,9) (740,83,5) (929,84,3) (905,85,8)
(530,87,4) (794,88,5) (487,90,4) (351,91,3) (273,93,3) (139,94,8) (489,96,9) (761,98,7)
```

```
Statistics:

Peak Hour: 100

Average Waiting Time: 103.606

Average Service Time: 5.987
```