DOCUMENTATIE

TEMA 2

STUDENT NAME: MURESAN ALEXANDRU DORIAN

GRUPA: 30229

# CUPRINS:

[1. Obiectivul temei 2](file:///C:\Users\User\Desktop\CTI\AN%20II\SEM%202\TP\QueManagementAssigment\QueManagementAssigment\Doc.docx#_Toc128043139)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 2](file:///C:\Users\User\Desktop\CTI\AN%20II\SEM%202\TP\QueManagementAssigment\QueManagementAssigment\Doc.docx#_Toc128043140)

[3. Proiectare 4](file:///C:\Users\User\Desktop\CTI\AN%20II\SEM%202\TP\QueManagementAssigment\QueManagementAssigment\Doc.docx#_Toc128043141)

[4. Implementare 6](file:///C:\Users\User\Desktop\CTI\AN%20II\SEM%202\TP\QueManagementAssigment\QueManagementAssigment\Doc.docx#_Toc128043142)

[5. Rezultate 12](file:///C:\Users\User\Desktop\CTI\AN%20II\SEM%202\TP\QueManagementAssigment\QueManagementAssigment\Doc.docx#_Toc128043143)

[6. Concluzii 13](file:///C:\Users\User\Desktop\CTI\AN%20II\SEM%202\TP\QueManagementAssigment\QueManagementAssigment\Doc.docx#_Toc128043144)

[7. Bibliografie 13](file:///C:\Users\User\Desktop\CTI\AN%20II\SEM%202\TP\QueManagementAssigment\QueManagementAssigment\Doc.docx#_Toc128043145)

# Obiectivul Temei

Crearea unei aplicatii de gestionare a cozilor bazata pe simulare, care atribuie clientii la cozi intr-un mod care minimizeaza timpul de asteptare. Aplicatia ar trebui sa simuleze sosirea a N clienti, intrarea lor in Q cozi, asteptarea, servirea si parasirea cozilor. Timpul de asteptare pentru fiecare client trebuie sa fie urmarit, iar timpul mediu de asteptare trebuie calculat. Utilizatorul ar trebui sa poata introduce numarul de clienti, numarul de cozi, intervalul de simulare si timpul minim/maxim de sosire si de servire. Aplicatia ar trebui sa atribuie fiecarui client coada cu timpul minim de asteptare atunci cand timpul de sosire al clientului este mai mare sau egal cu timpul de simulare.

Pentru simplicitate avem acest exemplu: Graphical user interface, text, application

Description automatically generated

# Analiza problemei, modelare, scenario, cazuri de utilizare

Prin analiza problemei ne referim la procesul de identificare, definire și înțelegere a unei probleme în vederea găsirii unei soluții sau a unui set de soluții. Avem un avantaj clar deoarece folosim un limbaj orientat obiect, ceea ce ne permite să lucrăm la un nivel conceptual superior, fără a fi limitați de munca tehnică.

Această strategie este cunoscută și sub denumirea de proiectare de jos în sus. Are un mare avantaj deoarece putem găsi ușor structuri care sunt conectate într-un mod cu lumea reală (obiecte). Este clar că acest concept nu are doar avantaje, ci vine și cu un nivel ridicat de complexitate.

Pentru a rezolva această problemă, aplicația trebuie să aibă un timp de simulare (interval de timp) și să urmărească timpul de sosire, timpul de servire și timpul de așteptare pentru fiecare client. Aplicația trebuie să țină evidența și a numărului de cozi și a numărului de clienți din fiecare coadă. Utilizatorul ar trebui să poată introduce numărul de clienți, numărul de cozi, intervalul de simulare și timpul minim/maxim de sosire și de servire.

Pentru a atribui clienții la cozi, aplicația ar trebui să folosească un algoritm de programare care ia în considerare timpul de sosire și timpul de servire al fiecărui client. O abordare ar fi să atribuim fiecărui client coada cu cel mai scurt timp de așteptare, care poate fi calculat prin luarea fiecărei cozi/servitoare disponibile și obținerea numărului de clienți care sunt serviti în momentul respectiv.

Aplicația ar trebui să calculeze, de asemenea, timpul mediu de așteptare pentru toți clienții și să îl afișeze utilizatorului la sfârșitul simulării. Acest lucru poate fi calculat prin însumarea timpului de așteptare pentru fiecare client și împărțind la numărul total de clienți.

**Cazuri de utilizare :**

- Furnizor de servicii: Aplicația poate fi utilizată de un furnizor de servicii pentru a gestiona cozile clienților și pentru a minimiza timpii de așteptare.

-Gestionarea evenimentelor: Aplicația poate fi folosită de organizatorii de evenimente pentru a gestiona cozile și pentru a optimiza timpii de așteptare pentru participanți. Câteva exemple: linii de bilete, linii de magazine etc.

# Proiectare

Dependențe, proiectare, structură, variabile și metode ale claselor:

A picture containing diagram, sketch, drawing, plan

Description automatically generated

Clasa View -> Este responsabilă pentru interfața de utilizare furnizată în aplicație. Are un ascultător la un buton astfel încât atunci când utilizatorul apasă butonul, simularea începe. De asemenea, oferă șapte textLabels care memorează intrările cu care ar trebui să funcționeze aplicația.

Clasa SimulationManager -> Este într-o relație puternică cu toate celelalte Clase furnizate în acest program. Această clasă este „managerul” aplicației.

Clasa Task -> Este pur și simplu o clasă care oferă informații despre un client individual, cum ar fi: id, ora la care ajunge la coadă și timpul necesar pentru a satisface acest client.

Clasa Server -> Este o clasă care este responsabilă pentru multithreading cozile. Similar cu clasa client, oferă informații de care coada are nevoie și oferă, de asemenea, o metodă care permite accesarea clasei de mai multe ori în paralel.

Clasa Scheduler -> Clasa Scheduler este într-o relație puternică cu clasa Server. Acesta creează numărul de servere/cozi care sunt furnizate de intrarea de la SimulationManager. De asemenea, are o metodă care adaugă clientul într-una dintre cozile care are numărul minim de clienți.

# Implementare

**View**

A screenshot of a computer program

Description automatically generated with medium confidence

View Screen este pur și simplu o clasă care este responsabilă pentru interfața (UI) a aplicației. Toată logica pentru interfață este pusă în constructorul clasei.

A screenshot of a computer

Description automatically generated

De asemenea, oferă o metodă clară, care este setată atunci când butonul a fost apăsat. Șterge toate intrările furnizate în UI. De asemenea, are o metodă submit Listener care va oferi un ascultător clasei de controler, atunci când butonul a fost apăsat.

**Main Class**

A screenshot of a computer

Description automatically generated

Pentru ca aplicația să pornească, are nevoie în metoda Main o referință la View Class și SimulationManager Class (care este managerul aplicației). Un obiect view este inițializat, ceea ce rezultă cu interfața afișată pe afișaj atunci când se rulează aplicația. Un obiect de tip controler este apoi inițializat, iar când intrările sunt furnizate pe interfață, de către utilizator, și butonul a fost apăsat, o variabilă booleană din Clasa SimulationManager este setată la adevărat -> firul principal de pe simulare este activat .

**Task Class**

A screenshot of a computer

Description automatically generated

Clasa Task este responsabilă pentru informațiile unui anumit client. Memorează un id pentru fiecare client diferit, o variabilă time Arrival care spune când clientul ar trebui să intre într-una dintre cozile disponibile și o variabilă time Service care specifică cât timp trebuie să petreacă pentru ca un client să fie servit.

**Server Class**

Această clasă are o responsabilitate similară cu Clasa Task. Acesta oferă informațiile necesare pentru fiecare coadă specifică.

A screenshot of a computer

Description automatically generated

Oferă un timp de sosire, un serviciu de timp, un timp total și un id. Folosim Atomic Integer în loc de int pentru că mai multe fire de execuție pot accesa și modifica în siguranță valoarea unei variabile întregi fără riscul de condiții de cursă sau de inconsecvență a datelor. Id-ul este folosit numai în scopuri de afișare pentru a specifica ce coadă este afișată.

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Clasa este extinsă dintr-o clasă Thread care ne va permite să facem multithreading (însemnând că fiecare coadă este un fir diferit). În timp ce acest fir de execuție este activ și coada are câțiva clienți, firul de execuție nu va fi accesat până când clientul curent a fost deservit și părăsit. Apoi elimină clientul din coadă.

**The Scheduler Class**

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Clasa Scheduler oferă o listă de cozi. Acesta creează numărul de cozi/ fire care a fost specificat când a fost furnizată intrarea pe interfață și pornește toate acele fire

**Controller Class**

A picture containing text, screenshot, software, multimedia software

Description automatically generated

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Când constructorul clasei este numit o variabilă writer de tip File Writer este inițializată, astfel încât rezultatul aplicației va fi afișat într-un anumit fișier text.

**Simulation Manager**

A computer screen shot of a program

Description automatically generated with low confidence

Când este apelată metoda de citire, fiecare intrare care a fost specificată pe interfață este salvată pe o variabilă globală pentru a fi mai ușor de lucrat cu ele. A picture containing text, screenshot, software, multimedia software

Description automatically generated

Această metodă este folosită pentru a genera o listă aleatorie de clienți cu ora de sosire și de serviciu. Metoda nu ia parametri de intrare și folosește variabile de instanță pentru a seta valorile minime și maxime pentru ora de sosire și timpul de service al clienților.

Metoda creează mai întâi un nou obiect Random pentru a genera numere aleatorii. Apoi trece prin numberOfClients de ori pentru a crea numărul specificat de clienți.

În interiorul buclei, metoda verifică dacă timpul maxim de sosire este mai mare decât timpul minim de sosire și dacă timpul maxim de serviciu este mai mare decât timpul minim de serviciu. Dacă această condiție este adevărată, metoda generează sosire aleatoare și timpi de serviciu pentru client folosind metoda nextInt() a clasei Random, care generează numere întregi aleatoare în intervalul specificat.

Metoda creează apoi un nou obiect Client cu orele de sosire și service generate și îl adaugă la lista de așteptare Clienți. În cele din urmă, metoda adaugă timpul de serviciu generat la variabila totalTimeService.

Dacă condiția este falsă și valorile maxime și minime nu sunt setate corect, metoda afișează un mesaj de eroare.

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Această metodă este utilizată pentru a calcula timpul mediu de așteptare pentru clienți în toate cozile din planificator. Nu ia parametri de intrare și returnează o valoare întreagă reprezentând timpul mediu de așteptare.

Metoda inițializează mai întâi o variabilă waitTime la zero, care va fi folosită pentru a stoca timpul total de așteptare în toate cozile. Apoi parcurge fiecare obiect ClientQueue din lista de cozi a programatorului.

În interiorul buclei, metoda adaugă atributul totalTime al fiecărei cozi (care reprezintă timpul total de așteptare pentru toți clienții din coadă) la variabila waitTime folosind metoda intValue() pentru a converti atributul totalTime într-o valoare întreagă.

După trecerea în buclă prin toate cozile, metoda returnează timpul mediu de așteptare prin împărțirea variabilei waitTime la atributul numberOfQueues al planificatorului, care reprezintă numărul total de cozi.

A picture containing text, screenshot, software, multimedia software

Description automatically generated

Această metodă este folosită pentru a scrie un șir într-un fișier și pe consolă. Ia un șir ca parametru de intrare și nu returnează nimic.

Metoda folosește un obiect writer pentru a scrie șirul de intrare într-un fișier. Se presupune că obiectul writer este inițializat înainte de apelarea acestei metode, cel mai probabil prin deschiderea unui fișier folosind un FileWriter sau o clasă similară.Text

Description automatically generated

The simulation of the application ends when the real time of the simulation is smaller or equal than the maximum simulation time provided from the input and when there are no more clients to be served or waiting.

Firstly we iterate trough the list of clients that are waiting. When the time Arrival of a client is smaller or equal than the simulation time, the client is added in the queue to be served and removed from the waiting list.

Text

Description automatically generated

After the clients that needed to be added were added and then removed from the waiting list, we display the remaining waiting clients. We take a Boolean variable that is set to false when there are no more waiting clients. We calculate then the peak hour, by updating an variable with the maximum number of clients in a queue.

Text

Description automatically generated

Repetăm prin cozi. Dacă nu mai există cozi, setăm o variabilă booleană la false. De asemenea, dacă nu mai sunt clienți în așteptare și toate cozile sunt goale, simularea se va opri.Text

Description automatically generated

Pentru finalul simulării afișăm așteptarea medie a unui client pentru a intra într-o coadă pentru a fi servit, ora de vârf și timpul mediu de service.

# Rezultate

Rezultatele date pentru informațiile de intrare furnizate:Text, letter

Description automatically generated

Aceste rezultate pot fi utilizate pentru a analiza performanța sistemului de așteptare, inclusiv măsuri precum timpul mediu de așteptare pentru clienți, rata de utilizare a resurselor de servicii și numărul de clienți care au fost deserviți în perioada de timp. O analiză ulterioară poate fi făcută și prin simularea sistemului de așteptare în diferite condiții, cum ar fi variarea numărului de resurse de servicii sau modificarea ratei de sosire a clienților. Aceste informații pot fi utile

îmbunătățirea eficienței și eficacității sistemului de așteptare.Text

Description automatically generated

Restul rezultatelor sunt puse în depozit pe cele trei fișiere text: test1.txt, test2.txt, test3.txt;

# Concluzii

În concluzie, specificația furnizată descrie o aplicație de gestionare a cozilor care simulează un sistem în care clienții sunt alocați la cozi pentru a minimiza timpul de așteptare. Aplicația ar trebui să urmărească timpul petrecut de fiecare client în cozi și să calculeze timpul mediu de așteptare. Interfața cu utilizatorul ar trebui să permită introducerea numărului de clienți, a numărului de cozi, a intervalului de simulare, a timpului minim și maxim de sosire și a timpului minim și maxim de serviciu.

O posibilă extensie ar putea fi să permită crearea și ștergerea dinamică a cozilor pe baza încărcării curente a sistemului. În plus, aplicația ar putea fi extinsă pentru a include o componentă de vizualizare pentru a afișa cozile și clienții pe măsură ce se deplasează prin sistem.

# Bibliography

<https://www.youtube.com/watch?v=r_MbozD32eo>

<https://www.youtube.com/watch?v=jiWNk9S46j4>

https://stackoverflow.com/