НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«Київський політехнічний інститут імені Ігоря Сікорського» Фaкультет інформaтики тa обчислювaльної техніки

Кaфедрa технічної кібернетики

Звіти до комп'ютерних прaктикумів з кредитного модуля “Моделювaння систем”

Виконaв

Студенти групи ІТ-02 Терешкович М.О.

Перевірив:

Київ – 2023

**Завдання:**

Зображення, що містить текст, Шрифт, знімок екрана, папір

Автоматично згенерований опис

**Зображення, що містить текст, Шрифт, знімок екрана, число

Автоматично згенерований опис**

**Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис**

**Завдання1:**package Bank;  
  
import java.text.DecimalFormat;  
import java.util.ArrayList;  
  
public class Main {  
  
 public static int *MAX\_QUEUE* = 3;  
 public static double *CASHIER1\_DELAY* = 0.3;  
 public static double *CASHIER2\_DELAY* = 0.3;  
 public static double *CREATOR\_DELAY* = 0.5;  
  
 public static int *TIME\_SIMULATION* = 10000;  
  
  
  
 private static final DecimalFormat *df* = new DecimalFormat("0.00");  
  
 public static void main(String[] args) {  
 *bankTask*();  
 }  
  
 public static void bankTask() {  
 Create creator = new Create("CREATOR", *CREATOR\_DELAY*);  
 Cashier cashier1 = new Cashier("CASHIER1", *CASHIER1\_DELAY*, *MAX\_QUEUE* );  
 Cashier cashier2 = new Cashier("CASHIER2", *CASHIER2\_DELAY*, *MAX\_QUEUE*);  
 Despose despose = new Despose("DESPOSE");  
  
 creator.setPriorityCashier(cashier1);  
 creator.setNonPriorityCashier(cashier2);  
  
 cashier1.setNextElement(despose);  
 cashier1.setAnotherCashier(cashier2);  
  
 cashier2.setNextElement(despose);  
 cashier2.setAnotherCashier(cashier1);  
  
 Model model = new Model(new ArrayList<>(){{  
 add(creator);  
 add(cashier1);  
 add(cashier2);  
 add(despose);}});  
  
 model.simulate(*TIME\_SIMULATION*);  
  
 System.*out*.println("1) Avarage load" + "\n\tCash№1: " + *df*.format(cashier1.getMeanLoad() / *TIME\_SIMULATION*) + ";" + "\n\tCash№2: " + *df*.format(cashier2.getMeanLoad() / *TIME\_SIMULATION*) + ";");  
 System.*out*.println("2)Avarage client in bank: " + *df*.format((cashier1.getMeanLoad() + cashier1.getMeanQueue() + cashier2.getMeanLoad() + cashier2.getMeanQueue()) / *TIME\_SIMULATION*));  
 System.*out*.println("3) Average between customers leaving the windows: " + *df*.format(*TIME\_SIMULATION* / (double)creator.getQuantity()));  
 System.*out*.println("4) Avarage time in bank: " + *df*.format(despose.getAverageTimeClientStayInBank()) + "ms;");  
 System.*out*.println("5) Avarage client in" + "\n\tQueue №1: " + *df*.format(cashier1.getMeanQueue() / *TIME\_SIMULATION*) + ";" + "\n\tQueue №2: " + *df*.format(cashier2.getMeanQueue() / *TIME\_SIMULATION*) + ";");  
 System.*out*.println("6)Rejection: " + *df*.format((creator.getFailure() / (double)creator.getQuantity()) \* 100) + "%;");  
 System.*out*.println("7)Road swap:" + "\n\tFrom №1 to №2: " + cashier1.getAmountOfSwitchesToAnotherCashier() + ";" + "\n\tFrom №2 to №1: " + cashier2.getAmountOfSwitchesToAnotherCashier() + ";");  
 }  
}

**Результат:   
Зображення, що містить текст, знімок екрана, програмне забезпечення, Мультимедійне програмне забезпечення

Автоматично згенерований опис**

**Завдання 2:**

package Durka;  
  
import java.text.DecimalFormat;  
import java.util.ArrayList;  
  
public class Main {  
  
 public static int *TIME\_SIMULATION* = 10000;  
  
 public static int *WORKERS\_RECEPTION* = 2;  
 public static int *WORKERS\_ROOM*= 2;  
 public static int *WORKERS\_REGISTRY* = 3;  
 public static int *WORKERS\_LAB* = 2;  
 private static final DecimalFormat *df* = new DecimalFormat("0.00");  
 public static void main(String[] args) {  
 *durkaTask*();  
 }  
  
 public static void durkaTask() {  
 Create creator = new Create("CREATOR");  
 Reception reception = new Reception("RECEPTION", *WORKERS\_RECEPTION*);  
 Process room = new Process("ROOM", *WORKERS\_ROOM*);  
 Process walkToRegistry = new Process("WALK\_TO\_REGISTRY");  
 Process registry = new Process("REGISTRY", *WORKERS\_REGISTRY*);  
 Lab lab = new Lab("LAB", *WORKERS\_LAB*);  
 Process walkToReception = new Process("WALK\_TO\_RECEPTION");  
 Despose despose = new Despose("DESPOSE");  
  
  
 creator.Exp\_Distribution(15);  
 room.Unif\_Distribution(3, 8);  
 walkToRegistry.Unif\_Distribution(2, 5);  
 registry.Erlang\_Distribution(4.5, 3);  
 lab.Erlang\_Distribution(4, 2);  
 walkToReception.Unif\_Distribution(2, 5);  
  
 creator.setNextElement(reception);  
 reception.setWalkToRegistry(walkToRegistry);  
 reception.setRoom(room);  
 room.setNextElement(despose);  
 walkToRegistry.setNextElement(registry);  
 registry.setNextElement(lab);  
 lab.setWalkToReception(walkToReception);  
 lab.setDespose(despose);  
 walkToReception.setNextElement(reception);  
  
 Model model = new Model(new ArrayList<>(){{  
 add(creator);  
 add(reception);  
 add(room);  
 add(walkToRegistry);  
 add(registry);  
 add(lab);  
 add(walkToReception);  
 add(despose);  
 }});  
  
 model.simulate(*TIME\_SIMULATION*);  
  
 System.*out*.println("(PATIENT) Time in system: " + *df*.format(despose.getAverageTimePatientStayInDurka()));  
 System.*out*.println("\tType1: " + *df*.format(despose.getAverageTimePatientStayInDurkaByType(1)));  
 System.*out*.println("\tType2: " + *df*.format(despose.getAverageTimePatientStayInDurkaByType(2)));  
 System.*out*.println("\tType3: " + *df*.format(despose.getAverageTimePatientStayInDurkaByType(3)));  
  
 System.*out*.println("Arrivals interval at the laboratory: " + *df*.format(*TIME\_SIMULATION* / (double)creator.getQuantity()));  
 }  
}

**Результат:**

**Зображення, що містить текст, знімок екрана

Автоматично згенерований опис**

**Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис**

**Висновок:**

У результаті виконання даної лабароторної роботи було створено дві системи імітації масового обслуговування – Банк та Госпіталь в резульатті чого ми змогли дослідити їх роботу та визначити оптимальні параметри для роботи кожної з них.