Prince Mohammad bin Fahd University

Data Structures - GEIT 2421

Section 101

Spring 2020

Instructed by Dr. Loay Alzubaidi

Assignment 2

by Abdulrahman Emad S Aleid

PMU#201800290

Assignment 2

Suppose there is only one customer service available in SAMBA Bank in Thursday morning, in every 4 minutes a new customer arrives at the end of waiting line, each customer will need 7 minutes for the service

Write a program to print out the information after the first 60 minutes

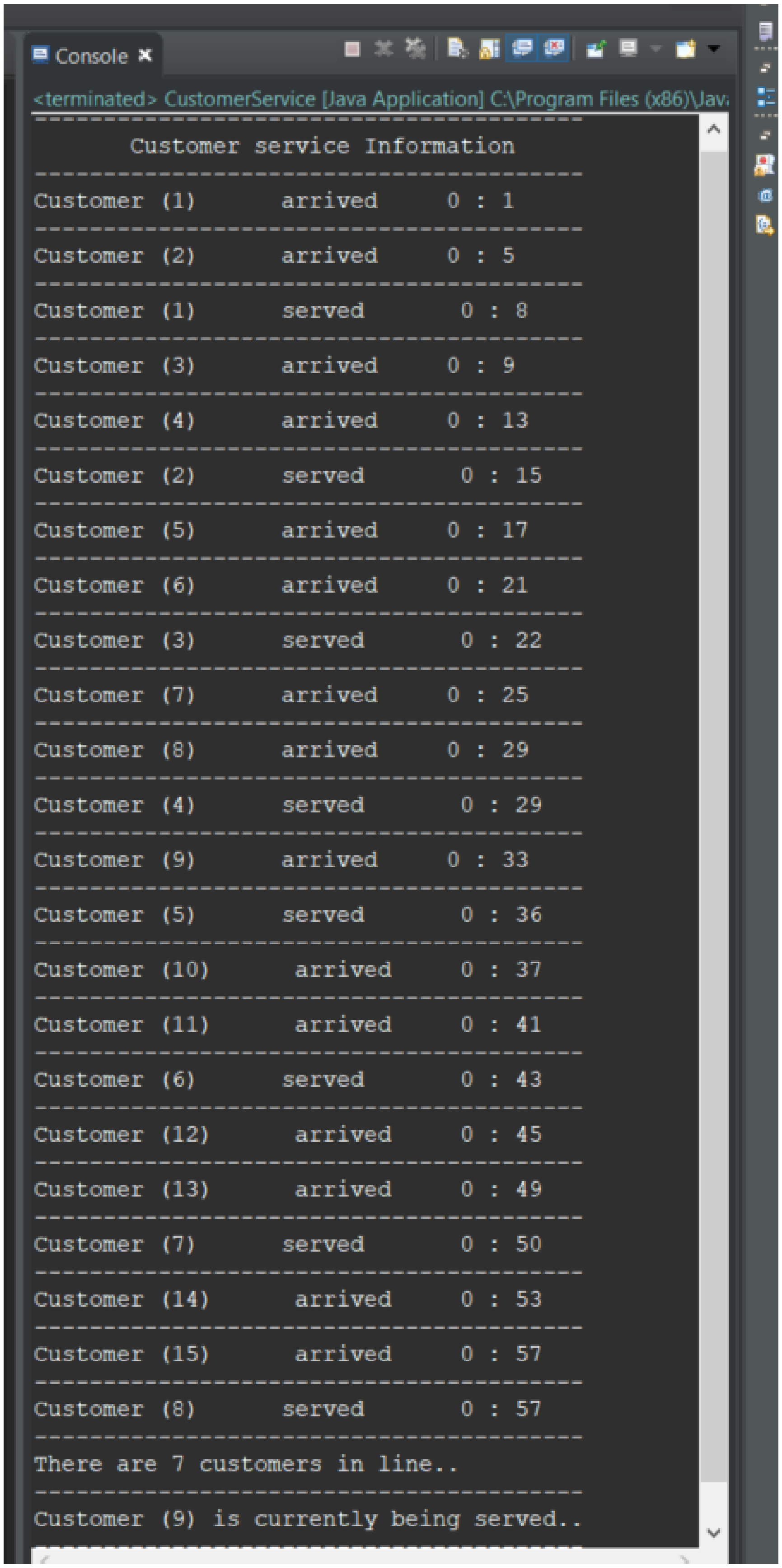
- **The time of arriving for each customer**
- **The time of leaving for each customer**
- **How many customers are in the line?**
- **Who is the current serving customer?**

```
public interface QueueInterface {

     public boolean isEmpty();
     public boolean isFull();
     public Object front();
     public Object back();
     public Object pop();
     public void push(Object item);
     public int size();
     public void printCustomersInfo();
```

```
}



public class Queue implements QueueInterface {

     private static int DEFAULT_CAPACItY = 100;
     private Object queue[];
     private int rear, front;
     private int time = 0;

     public Queue() {
          this(DEFAULT_CAPACItY);
     }

     public Queue(int capacity) {
          if(capacity < 2)
               throw new IllegalArgumentException("Capacity
Must be > 1");
          queue = new Object[capacity];
          rear = front = 0;
     }

     public int size(){
          if(rear >= front)
               return (rear - front);
          else
               return (rear - front + queue.length);
     }

     public boolean isEmpty() {
          return (rear == front);
     }

     public boolean isFull() {
          return (size() == queue.length - 1);
     }

     public void push(Object item) {
          if(isFull())
               throw new IllegalStateException("Queue is
full");
          queue[rear] = item;
          rear = (rear + 1) % queue.length;
     }

     public Object pop() {
          if(isEmpty())
               throw new IllegalStateException("Queue is 
empty");
          Object frontItem = queue[front];
          queue[front] = null;
          front = (front + 1) % queue.length;
          return frontItem;

     }

     public Object front() {
          if(isEmpty())
               throw new IllegalStateException("Queue is 
empty");
          return queue[front];
     }

     public Object back() {
          return (queue[rear - 1]);
     }

     public void printCustomersInfo() {
          int id = 1;
          int timeOfArrival = 1;
          int timeOfLeaving = 8;

          System.out.println("-------------------------------
----------");
          System.out.println("       Customer service 
Information");
          System.out.println("-------------------------------
----------");

          for (int time = 0; time <= 60; time++) {
               if (time == timeOfArrival) {
                    push(new Coustomer(id, timeOfArrival, 
0));

                    String printCustInfo = ((Coustomer) 
back()).printArrival();
                    System.out.println(printCustInfo);
                    timeOfArrival += 4;
                    id++;
               }
               if (time == timeOfLeaving) {
                    ((Coustomer)
front()).setTimeOfLeaving(timeOfLeaving);

                    String printCustInfo = ((Coustomer)
pop()).printLeaving();
                    System.out.println(printCustInfo);

                    timeOfLeaving += 7;
               }
          }

          int numOfCustInLine = size();
          System.out.println("There are "
                   + numOfCustInLine
                    + " customers in line..");
          System.out.println("---------------"
                    + "-------------------------");

          String currentCust = ((Coustomer)
front()).printCustomer();
          System.out.println(currentCust);
     }
```

```
public class Coustomer {

     private int timeOfArrival;
     private int timeOfLeaving;
     private int id;

     public Coustomer(int id, int timeOfArrival, int
timeOfLeaving) {
          this.id = id;
          this.timeOfArrival = timeOfArrival;
          this.setTimeOfLeaving(timeOfLeaving);
     }

     int getTimeOfLeaving() {
          return timeOfLeaving;
     }

     void setTimeOfLeaving(int timeOfLeaving) {
          this.timeOfLeaving = timeOfLeaving;
     }

     public String printArrival() {
          String str = "";
          str += "Customer (" + this.id
                   + ")      arrived     0 : "
                   + this.timeOfArrival
                  + "\n------------------"
                  + "--------------------";
          return str;
     }

     public String printLeaving() {
          String str = "";
          str += "Customer (" + this.id
                   + ")      served       0 : "
                  + this.getTimeOfLeaving()
                 + "\n------------------"
                 + "--------------------";
          return str;
     }

     public String printCustomer() {
        String str = "";
        str += "Customer (" + this.id
                 + ") is currently being served.."
                + "\n-------------------"
                + "---------------------";
        return str;
    }

}
```

```
public class CustomerService {

    public static void main(String[] args) {

        QueueInterface queue = new Queue(16);
        queue.printCustomersInfo();
   }
}
```



```
<terminated> CustomerService [Java Application] C:\Program Files (x86)\Jav
----------------------------------------
       Customer service Information
----------------------------------------
Customer (1)      arrived     0 : 1
----------------------------------------
Customer (2)      arrived     0 : 5
----------------------------------------
Customer (1)      served       0 : 8
----------------------------------------
Customer (3)      arrived     0 : 9
----------------------------------------
Customer (4)      arrived     0 : 13
----------------------------------------
Customer (2)      served       0 : 15
----------------------------------------
Customer (5)      arrived     0 : 17
----------------------------------------
Customer (6)      arrived     0 : 21
----------------------------------------
Customer (3)      served       0 : 22
----------------------------------------
Customer (7)      arrived     0 : 25
----------------------------------------
Customer (8)      arrived     0 : 29
----------------------------------------
Customer (4)      served       0 : 29
----------------------------------------
Customer (9)      arrived     0 : 33
----------------------------------------
Customer (5)      served       0 : 36
----------------------------------------
Customer (10)      arrived     0 : 37
----------------------------------------
Customer (11)      arrived     0 : 41
----------------------------------------
Customer (6)      served       0 : 43
----------------------------------------
Customer (12)      arrived     0 : 45
----------------------------------------
Customer (13)      arrived     0 : 49
----------------------------------------
Customer (7)      served       0 : 50
----------------------------------------
Customer (14)      arrived     0 : 53
----------------------------------------
Customer (15)      arrived     0 : 57
----------------------------------------
Customer (8)      served       0 : 57
----------------------------------------
There are 7 customers in line..
----------------------------------------
Customer (9) is currently being served..
----------------------------------------
```