TK1143 Program Design

QUEUE

Section A:

1. Write a java statement to declare and create:

a) An object variable timeQueue that represent a data of arrival time of a passenger.

```
Queue <String> timeQueue = new Queue <String> ( ) ;
```

b) An object variable author with data type BOOK class.

```
Queue <String> author = new Queue <String> ( ) ;
```

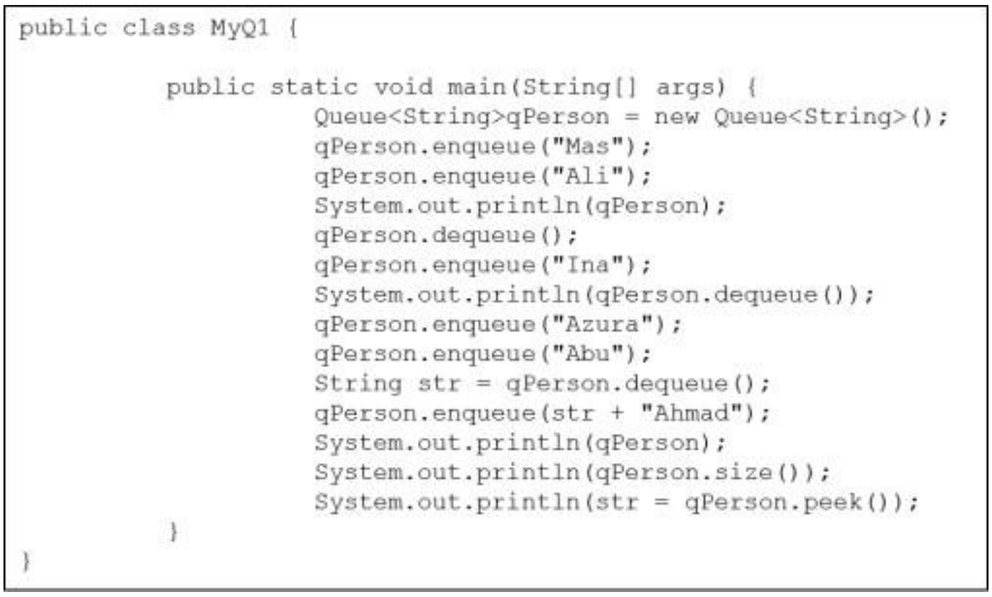
2. Complete the following table with the description of each method in Queue.

Method	Description
isEmpty( )	To check if the queue is empty
enqueue( )	To add item into the queue
dequeue( )	To remove the front item from the queue
peek()	To retrieve the front item without removing it
size( )	To get the number of items in the queue

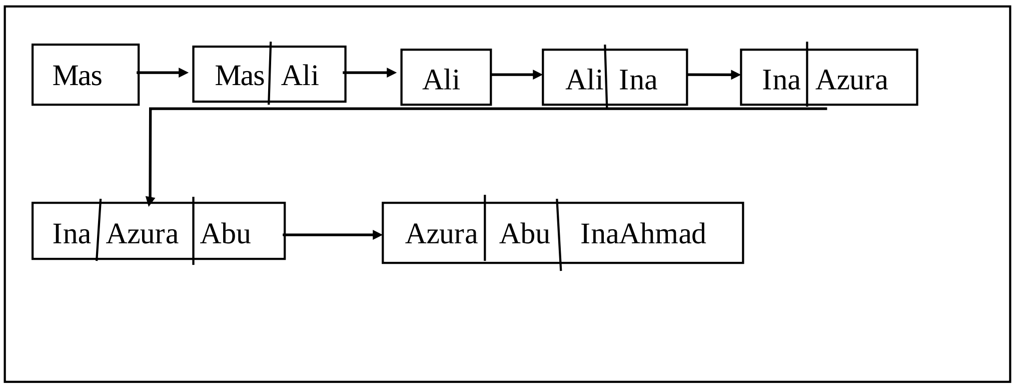
3. Complete the following table that shows a series of stack operations and their effects on an initially empty queue Q of character.

Method	**Return Value**	**First←Q←Last**
enqueue(p)	-	[p]
enqueue(r)	-	[p, r]
size( )	2	[p, r]
first( )	p	[p, r]
enqueue(o)	-	[p, r, o]
isEmpty( )	false	[p, r, o]
dequeue( )	p	[r, o]
dequeue( )	r	[o]
size( )	1	[o]
dequeue( )	o	[]
isEmpty( )	true	[]
enqueue(g)	-	[g]
enqueue(h)	-	[g,h]
first( )	g	[g,h]
dequeue( )	g	[h]
size( )	1	[h]
first( )	h	[h]

4. Refer to the given program. Draw a diagram to represent the following queue operations, step-by-step as the program executes. Show the output, if any.



```
public class MyQ1 {

        public static void main(String[] args) {
                Queue<String>qPerson = new Queue<String>();
                qPerson.enqueue("Mas");
                qPerson.enqueue("Ali");
                System.out.println(qPerson);
                qPerson.dequeue();
                qPerson.enqueue("Ina");
                System.out.println(qPerson.dequeue());
                qPerson.enqueue("Azura");
                qPerson.enqueue("Abu");
                String str = qPerson.dequeue();
                qPerson.enqueue(str + "Ahmad");
                System.out.println(qPerson);
                System.out.println(qPerson.size());
                System.out.println(str = qPerson.peek());
        }
}
```



Output :

Mas , Ali
Ali
Azura , Abu , InaAhmad
3
Azura

Section B:

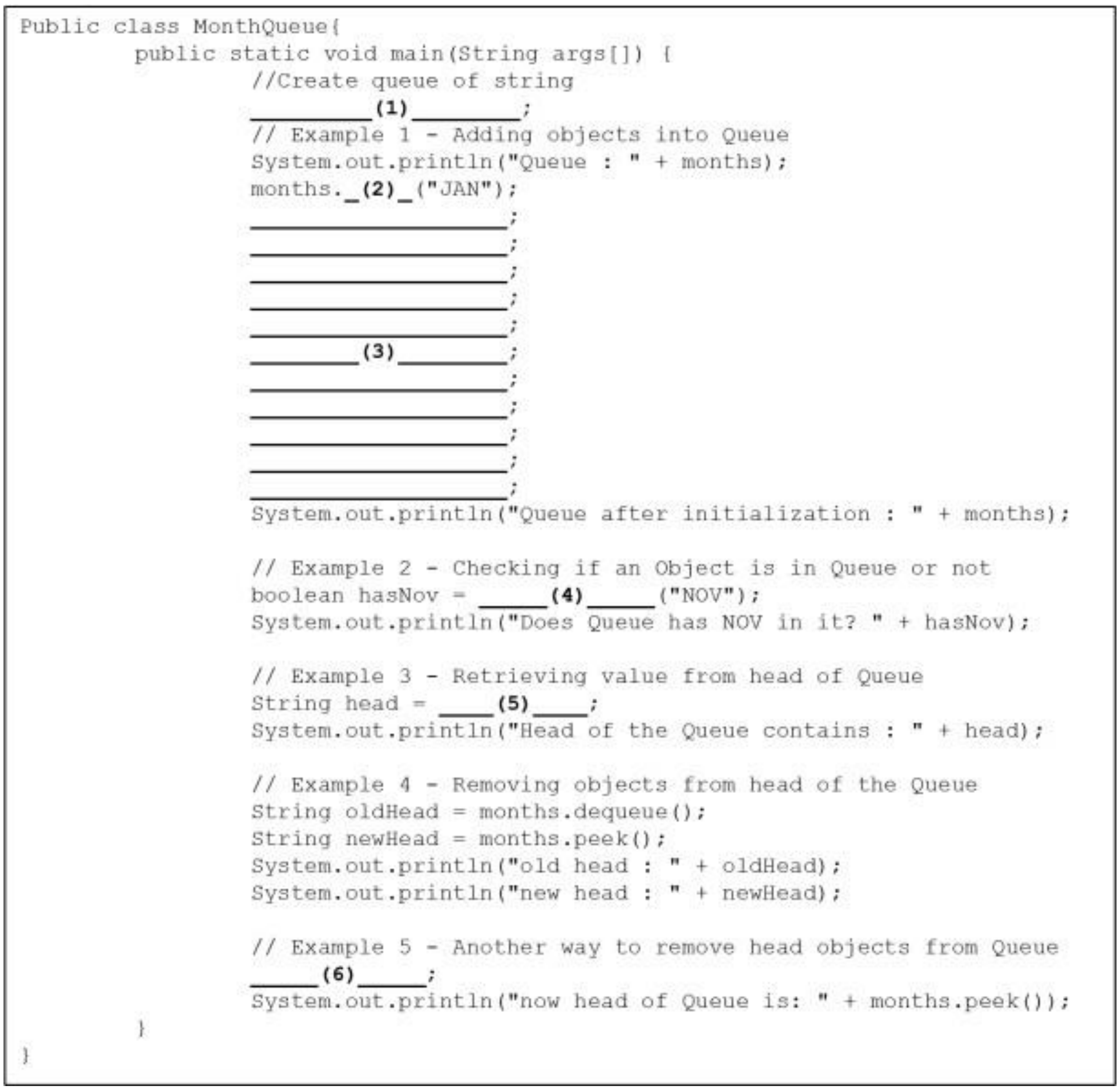
1. What is the output of the following program? Draw the content of vChar.

```java
public class Vchar {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Queue <Character> vChar = new Queue<Character>();
		String text = " I love to learn TK1143 Program Design !";

		for(int i=0; i< text.length(); i++) {
			char c = text.charAt(i);
			if(c=='a' || c=='e' || c=='i' || c=='o' || c=='u')
				vChar.enqueue(c);
		}
		System.out.println(vChar.size());

		while(!vChar.isEmpty())
			System.out.print(vChar.dequeue() + " ");
	}
}
```

Output:

```
9
o e o e a o a e i
```

2. Complete the following program, trace and give the output.



```
Public class MonthQueue{
        public static void main(String args[]) {
                //Create queue of string
                _________(1)________;
                // Example 1 - Adding objects into Queue
                System.out.println("Queue : " + months);
                months._(2)_("JAN");
                ___________________;
                ___________________;
                ___________________;
                ___________________;
                ___________________;
                ________(3)________;
                ___________________;
                ___________________;
                ___________________;
                ___________________;
                ___________________;
                System.out.println("Queue after initialization : " + months);

                // Example 2 - Checking if an Object is in Queue or not
                boolean hasNov = _____(4)____("NOV");
                System.out.println("Does Queue has NOV in it? " + hasNov);

                // Example 3 - Retrieving value from head of Queue
                String head = ____(5)____;
                System.out.println("Head of the Queue contains : " + head);

                // Example 4 - Removing objects from head of the Queue
                String oldHead = months.dequeue();
                String newHead = months.peek();
                System.out.println("old head : " + oldHead);
                System.out.println("new head : " + newHead);

                // Example 5 - Another way to remove head objects from Queue
                _____(6)_____;
                System.out.println("now head of Queue is: " + months.peek());
        }
}
```

(1) Queue<String>months=new Queue<String>();

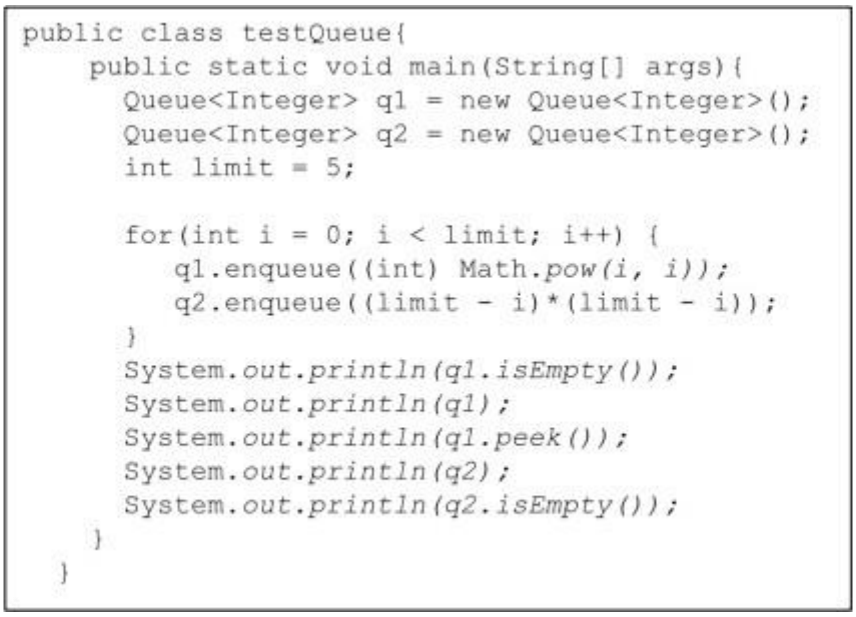
(2) enqueue

(3) months.enqueue("FEB") ;
    months.enqueue("MAR") ;
    months.enqueue("APR") ;
    months.enqueue("MAY") ;
    months.enqueue("JUN") ;
    months.enqueue("JUL") ;
    months.enqueue("AUG") ;
    months.enqueue("SEP") ;
    months.enqueue("OCT") ;
    months.enqueue("NOV") ;
    months.enqueue("DEC") ;

(4) months.contains
(5) months.peek();
(6) months.poll();

```
Output:

Queue:
Queue after initialization: JAN FEB MAR APR MAY JUN
JUL AUG SEP OCT NOV DEC
Does Queue has NOV in it :true
Head of the Queue contains: JAN
Old head: JAN
New head: FEB
now head of Queue is: MAR
```

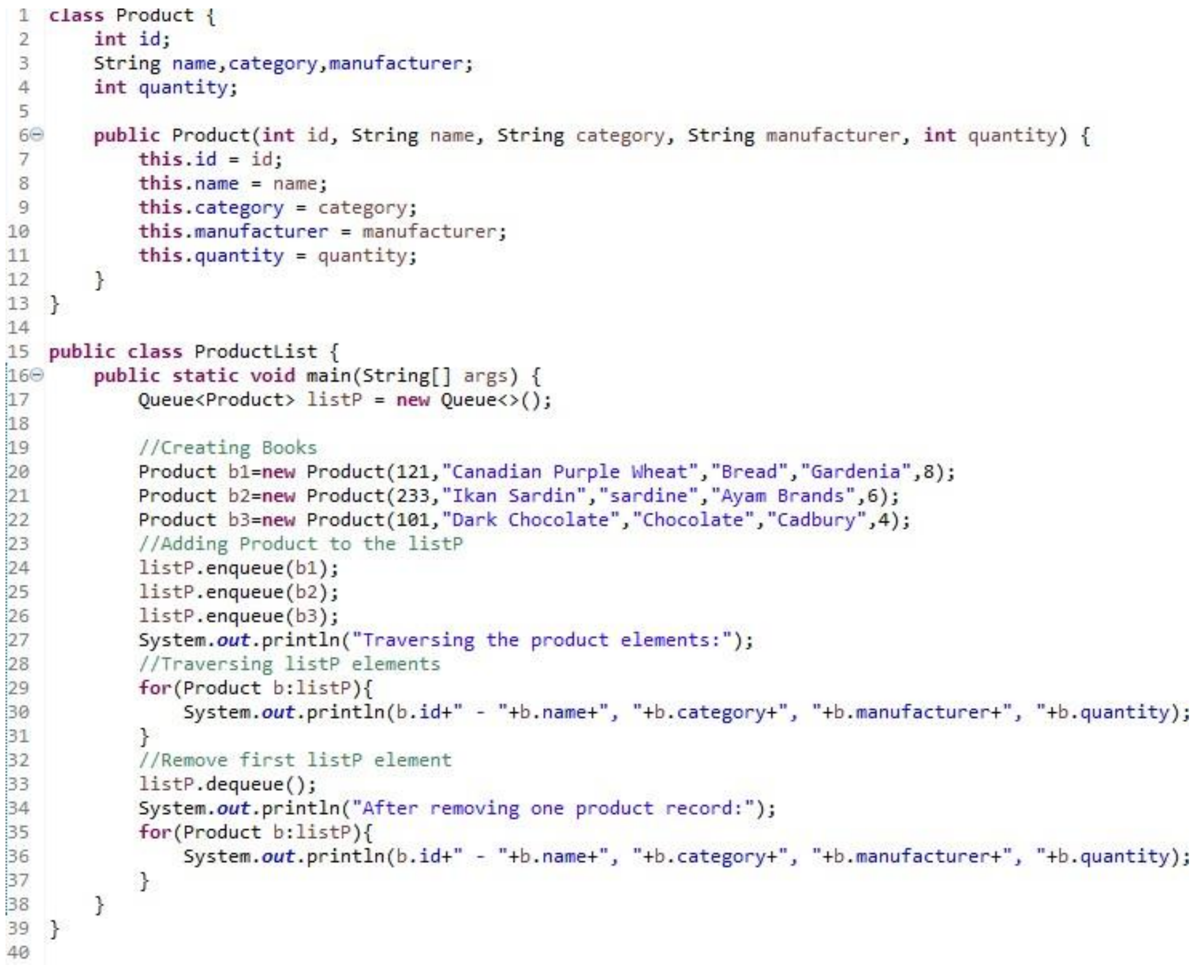
3. What is the output of the following program?



```
public class testQueue{
    public static void main(String[] args){
      Queue<Integer> q1 = new Queue<Integer>();
      Queue<Integer> q2 = new Queue<Integer>();
      int limit = 5;

      for(int i = 0; i < limit; i++) {
         q1.enqueue((int) Math.pow(i, i));
         q2.enqueue((limit - i)*(limit - i));
      }
      System.out.println(q1.isEmpty());
      System.out.println(q1);
      System.out.println(q1.peek());
      System.out.println(q2);
      System.out.println(q2.isEmpty());
    }
  }
```

```
Output:

false
1 1 4 27 256
1
25 16 9 4 1
false
```

4. Trace the given program and give the output.



```java
class Product {
    int id;
    String name,category,manufacturer;
    int quantity;

    public Product(int id, String name, String category, String manufacturer, int quantity) {
        this.id = id;
        this.name = name;
        this.category = category;
        this.manufacturer = manufacturer;
        this.quantity = quantity;
    }
}

public class ProductList {
    public static void main(String[] args) {
        Queue<Product> listP = new Queue<>();

        //Creating Books
        Product b1=new Product(121,"Canadian Purple Wheat","Bread","Gardenia",8);
        Product b2=new Product(233,"Ikan Sardin","sardine","Ayam Brands",6);
        Product b3=new Product(101,"Dark Chocolate","Chocolate","Cadbury",4);
        //Adding Product to the listP
        listP.enqueue(b1);
        listP.enqueue(b2);
        listP.enqueue(b3);
        System.out.println("Traversing the product elements:");
        //Traversing listP elements
        for(Product b:listP){
            System.out.println(b.id+" - "+b.name+", "+b.category+", "+b.manufacturer+", "+b.quantity);
        }
        //Remove first listP element
        listP.dequeue();
        System.out.println("After removing one product record:");
        for(Product b:listP){
            System.out.println(b.id+" - "+b.name+", "+b.category+", "+b.manufacturer+", "+b.quantity);
        }
    }
}
```

Output:

```
Traversing the product elements:
121 - Canadian Purple Wheat - Bread - Gardenia - 8
233 - Ikan Sardin - sardine - Ayam Brands - 6
101 - Dark Chocolate - Chocolate - Cadbury - 4
After removing one product record:
233 - Ikan Sardin - sardine - Ayam Brands - 6
101 - Dark Chocolate - Chocolate - Cadbury - 4
```

5. Complete the following program as input and output given.

Sample I/O :

Input	Output
2 java proggramming is fun end i love UKM end	4 : java programming is fun 3 : i love UKM

```
public class CountWord {

    public static void main(String[] args) {
            // TODO Auto-generated method stub

        Queue <String> myQueue = new Queue<String>();

        Scanner sc = new Scanner(System.in);
        //case number
        int num = sc.nextInt();
        sc.nextLine();

        int i=0;

        //insert string
        while(i<num) {
            String str = sc.nextLine();
            //break the string using stringtokenizer and insert into myQueue
            StringTokenizer st = new StringTokenizer(str);
            while (st.hasMoreTokens()) {
                                    A

            // display the number of words in myQueue
                                    B
            //display the words in myQueue
            while(!myQueue.isEmpty()) {
                                    C
            }
            System.out.println();
        }
    }
}
```

A

```
String str= sc.nextLine();
StringTokenizer st= new StringTokenizer(str);
while (st.hasMoreTokens()) {
        String tem=st.nextToken();
        if(!(tem.equals("end")))
                myQueue.enqueue (tem);
}
```

B

```
System.out.print(myQueue.size() + "");
```

C

```
while(ImyQueue.isEmpty()) {
         System.out.print(myQueue.dequeue() + " ");
```

```
Output :

4 : java programming is fun
3 : I love UKM
```

Section C:

There are four (4) questions given in this section. Your task is to write the program to implement data structure Stack for each case.

ODD EVEN NUMBER	
Input	Standard Input
Output	Standard Output
Programming Elements	Loop
Data Structure	Queue

Problem Description
Even Numbers are integers that are exactly divisible by 2, whereas an odd number cannot be exactly divided by 2. Example of even numbers are 2,4,6,8 and odd numbers are 1, 3, 5, 7, 9.

Input
The first line input contains an integer, which determines the number of test cases. Each of following lines represent sequence of integers that ends with 0.

```
2
34 1 8 5 22 0
10 7 16 -2 0
```

Output
For each test case, the output will present the size of oddQueue and evenQueue following with the integers of odd and even numbers.

```
oddQueue 2: 1 5
evenQueue 3: 34 8 22
oddQueue 1: 7
evenQueue 3: 10 16 -2
```

Your task is to write a program that will read the input and identify whether it is an even number or odd number.

Self-Study
1. Explain the implementation of Queue in this case study.

PALINDROME

Input	Standard Input
Output	Standard Output
Programming Elements	Loop
Data Structure	Stack & Queue

Problem Description

Palindrome is a sequence of characters, a word, phrase, number or sequence of words, which reads the same backward as forward. Example of palindrome words are katak, civic and anna.

Input

The first line input contains an integer, which determines the number of test cases. Each of following lines represent words.

```
3
racecar
java
madam
```

Output

For each test case, the output contains respond either 'It is a Palindrome' or 'Not a palindrome'.

```
It is a Palindrome
Not a palindrome
It is a Palindrome
```

Your task is to write a program that will read the word and identify whether it is a palindrome or not. You must use stack and queue in your program.

Self Study

1. Explain the implementation of Stack and Queue in this case study.

ABC WASH MACHINE

Input	Randomly generated
Output	Standard Output
Programming Elements	Selection, Looping
Data Structure	Queue
Additional Function	class Clock, rand()

Problem Description
ABC Wash Machine provides a self-wash car for its customer. The customer must queue up in order to get the service. The customer will immediately be served if the queue is empty. If the machine is free, then it will serve the front customer in the queue. The machine takes 5 minutes to complete one service. The machine operation starts at 8.00 am and ends at 8.30 am. However, it will continue serve customer who arrive before or by 8.30 am.

The owner of the ABC Wash Machine would like to know some statistical information so that he can improve the machine in the future. The information are:

- Number of customer that arrives by 8.30 am.
- Longest customer waiting time.
- Average customer waiting time.
-

Assume that all customers are an ethical person and determined to get the service, also the machine is ideal. Input for customer arrival times will randomly generated by rand() function. Assume that the next customer is likely to arrive within 9 minutes after the current customer.

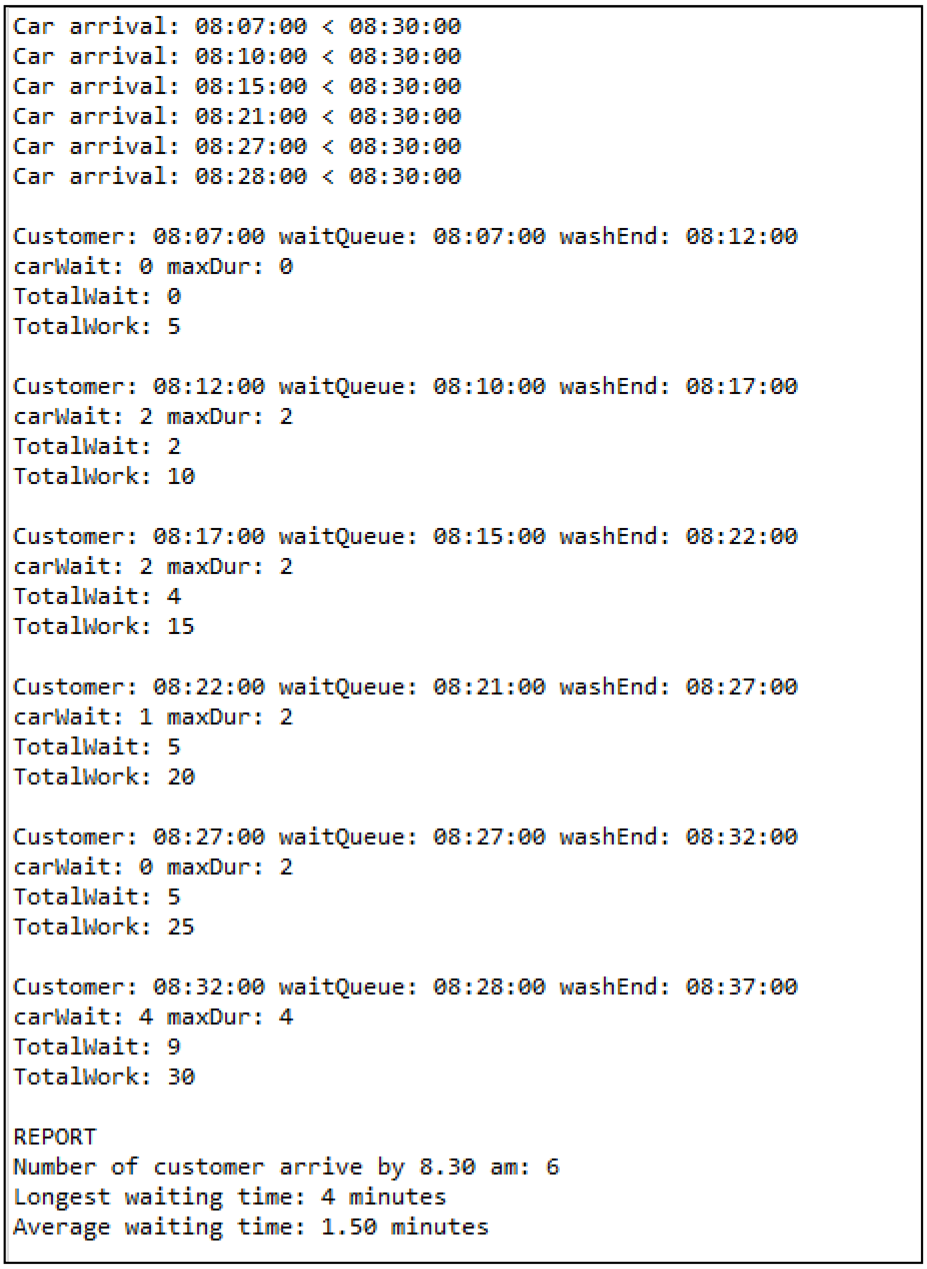
Your task is to modify the given program that will print out the report information as below.

Input
Randomly generated.

Sample Output
<List of car arrival time>
Car arrival : <car arrival time> < <ABC wash machine end time>

< list of car waiting time information>
Customer: <Start_served> waitQueue: <Car_arrival_time> washEnd: <Servise_has_ended>
carwait: <Waiting_time_per_customer> maxDur: < Longest_waiting_time>
Totalwait: <Sum_of_waiting_time>
Totalwork: <Sum_of_machine_works>

REPORT
Number of customer that arrives by 8.30 am: <Number_ customer _arrives_by 8.30>
Longest waiting time: < Longest_waiting_time>
Average waiting time: < Average_waiting_time>



```
Car arrival: 08:07:00 < 08:30:00
Car arrival: 08:10:00 < 08:30:00
Car arrival: 08:15:00 < 08:30:00
Car arrival: 08:21:00 < 08:30:00
Car arrival: 08:27:00 < 08:30:00
Car arrival: 08:28:00 < 08:30:00

Customer: 08:07:00 waitQueue: 08:07:00 washEnd: 08:12:00
carWait: 0 maxDur: 0
TotalWait: 0
TotalWork: 5

Customer: 08:12:00 waitQueue: 08:10:00 washEnd: 08:17:00
carWait: 2 maxDur: 2
TotalWait: 2
TotalWork: 10

Customer: 08:17:00 waitQueue: 08:15:00 washEnd: 08:22:00
carWait: 2 maxDur: 2
TotalWait: 4
TotalWork: 15

Customer: 08:22:00 waitQueue: 08:21:00 washEnd: 08:27:00
carWait: 1 maxDur: 2
TotalWait: 5
TotalWork: 20

Customer: 08:27:00 waitQueue: 08:27:00 washEnd: 08:32:00
carWait: 0 maxDur: 2
TotalWait: 5
TotalWork: 25

Customer: 08:32:00 waitQueue: 08:28:00 washEnd: 08:37:00
carWait: 4 maxDur: 4
TotalWait: 9
TotalWork: 30

REPORT
Number of customer arrive by 8.30 am: 6
Longest waiting time: 4 minutes
Average waiting time: 1.50 minutes
```

Solution

The algorithm for ABC Wash Machine simulation:

```
Set the operation time.
Generate all car arrivals, and push them onto arrival queue.
Set the initial state for the wash machine (status, startWash, endWash)
Set current event.
    While there is still an event
        If car arrival event occurs:
          Push car onto waiting queue.
        If machine finish washing
          Update machine state.
        If machine available and there is a car waiting:
          Front car in waiting queue will be washed.
          Remove this car from waiting queue.
          Update the machine state and statistical information.
        If machine available and there is no car waiting:
          Update endWash to beyond operation time.
        Jump to the next earliest event that is either car arrival or machine finish washing.

Produce report
```

Basic Structure:

```
int main(){
        Clock carArrival;
        Queue <Clock> waitQueue;
        Queue <Clock> arrivalQueue;

        enum status {free, busy} machineWash;

        //observation data
        Clock totalWaitTime, maxWaitTime;
        Clock totalServiceTime;
        Clock startTime, endTime;
        Clock washEnd, washStart;

        Clock i;
        int nextArrival;



        for (i=startTime;(i.lessThan(endTime)||(!waitQueue.empty())||(!arrivalQueue.empty()));)
        {


        }


        return 0;
}
```

Full structure of the worked-example program: ABC Wash Machine Simulation

```java
import java.util.*;

public class ABCWashMachine {
    public enum status {free, busy}
    static int maxDur=0;
    static int numCar=0;
    static int totalWait=0;
    static int totalWork=0;

    public static void main(String[] args) {
        Clock carArrival;
        Queue <Clock> waitQueue= new Queue <>();
        Queue <Clock> arrivalQueue = new Queue<>();

        //enum status {free, busy} machineWash;
        status machineWash;

        //observation data
        Clock totalWaitTime= new Clock();
        Clock maxWaitTime = new Clock();
        Clock totalServiceTime = new Clock();
        Clock startTime= new Clock();
        Clock endTime= new Clock();
        Clock washEnd = new Clock();
        Clock washStart = new Clock();

        Clock i;
        int nextArrival=0;
        Random rand = new Random();

        startTime.setTime(8,0,0);
        endTime.setTime(8,30,0); // can change to 12 pm

        for (i=startTime.getCopy();i.lessThan(endTime); ){
            nextArrival = rand.nextInt(10); //nextArrival in the range 0 to 9
            i.addTimeMinute(nextArrival);
            if(i.lessThan(endTime)) {
                arrivalQueue.enqueue(i.getCopy());
                System.out.println("car arrival: " + i.toString() + " < " +
                        endTime.toString());
            }
        }

        //start the simulation
        machineWash=status.free;
        if (!arrivalQueue.isEmpty()) {
            startTime=arrivalQueue.peek();
            washEnd=startTime.getCopy();
            washEnd.addTimeMinute(5);
        } else
            startTime=endTime.getCopy();

        Clock del;
        for
        (i=startTime;(i.lessThan(endTime)||(!waitQueue.isEmpty())||(!arrivalQueue.isEmpty()););)
        {
            if (!arrivalQueue.isEmpty())
                if (i.equalTime(arrivalQueue.peek())) {
                    waitQueue.enqueue(i.getCopy());
                    del=arrivalQueue.dequeue();
                }

            if ((machineWash==status.busy) && (i.equalTime(washEnd))) {
                washEnd.setTime(14,0,0);
                machineWash=status.free;
            }

            if ((machineWash==status.free) && !(waitQueue.isEmpty())) {
                washStart=i.getCopy();
                washEnd=i.getCopy(); washEnd.addTimeMinute(5);
                doAnalysis(i,waitQueue.peek(),washEnd); // call doAnalysis method
                del=waitQueue.dequeue();
                machineWash=status.busy;
            }
75
76              if ((machineWash==status.free) && (waitQueue.isEmpty()))
77                  washEnd.setTime(14,0,0);
78
79              //jump to next event.
80              if (!arrivalQueue.isEmpty())
81                  if (washEnd.lessThan(arrivalQueue.peek())) {
82                      i=washEnd.getCopy();
83                  }
84                  else {
85                      i=arrivalQueue.peek().getCopy();
86                  }
87              else {
88                  i=washEnd.getCopy();
89              }
90          }
91
92          //report
93          System.out.print("\nREPORT\n");
94          System.out.print("Number of customer arrive by 8.30 am: " + numCar + "\n");
95          System.out.print("Longest waiting time: " + maxDur + " minutes\n");
96          System.out.print(String.format("Average waiting time: %.2f minutes",
97                  totalWait/(float)numCar));
98      }
99
100
101     public static void doAnalysis(Clock waitStop, Clock start, Clock washStop) {
102
103         int carWait,machineWork;
104
105         carWait=start.durationSec(waitStop.getCopy());
106         machineWork=waitStop.durationSec(washStop.getCopy());
107         numCar++;
108         totalWait+= carWait;
109         totalWork+=machineWork;
110         if (maxDur<carWait)
111             maxDur=carWait;
112     }
113 }
114
```

Self-activity

1. What is the statistical information needed in this problem?
2. Let say, the wash machine operation hours is from 8.00am to 9.00am and the machine takes 10 minutes to complete one service. The car arrival times are as follows:
   i. 8.05am, 8.09am, 8.12am, 8.25am, and 8.55am.
3. Calculate the statistical information in this case.
4. Determine the simulation characteristic of the ABC Wash Machine.

Tutorial Activity

1. How the program get the input of customer's arrival?
2. List the variables of type Clock.
3. What is the data type used to determine the average waiting time and number of customer. Why?
4. List out the standard function and user defined function used in the given program.
5. Identify main and supportive identifiers in the program
6. Which line in the program shows the following?
   - Generate all car arrivals, and push them onto arrival queue.
   - Set the initial state for the wash machine

- Jump to the next earliest event that is either car arrival or machine finish washing.
- Update the statistical information.

7. If we were to change the machine working hours from 8am -9am, which line of code need to be change?

Hands-on Activity

Implement the worked-example of ABC Wash Machine program. Try to understand the output printed by the program.

DEF WASH MACHINE

Input	Randomly generated
Output	Standard Output
Programming Elements	Selection, Looping
Data Structure	Queue
Additional Function	class Clock, rand()

Problem Description

The DEF Wash Machine problem will use the same problem description as the worked-example of ABC Wash Machine with regard the DEF Wash Machine provides two types of services:

Type 1: normal wash (takes 6 minutes for every service)
Type 2: wash and polish (takes 10 minutes for every service).

The owner of the DEF Wash Machine would like to know the statistical information as below:

- Total customers: <total_customer>
- Number of customers for service of type 1: <Number_ customer _type_1>
- Number of customers for service of type 2: <Number_ customer _type_2>
- Longest customer waiting time: < Longest_waiting_time>
- Average customer waiting time: < Average_waiting_time>

Your task is to modify the worked-example of ABC Wash Machine and print out the above statistical information.

Input
Input are randomly generated for both - car arrival and services type.

Sample Output

```
Total customers:
Number of customers for service of type 1: <Number_ customer _type_1>
Number of customers for service of type 2: <Number_ customer _type_2>
Longest customer waiting time: < Longest_waiting_time>
Average customer waiting time: < Average_waiting_time>
```

Tips

1. Queue in DEF Wash Machine problem holds two information: arrival time and service type. Therefore, we would like to suggest you to use class Customer as follows:

```
public class Customer {
        Clock time = new Clock ();
        int serviceType;

        public Customer (Clock tm, int sType)
        {
                time.setTime(tm.getHours(),tm.getMinutes(),tm.getSeconds());
                serviceType = sType;
        }

        public Customer()
        {
                time.setTime(0,0,0);
                serviceType =0;
        }

        public int getSeviceType()
        {
                int temp;
                temp=this.serviceType;
                return  temp;
        }

        public int setSeviceType()
        {
                int temp;
                temp=this.serviceType;
                return  temp;
        }

        public Customer CopyCustomer()
        {
                Customer temp = new Customer();
                temp.time = time.getCopy();
                temp.serviceType= getSeviceType();
                temp.toString();

                return temp;
        }

        public String toString()
        {
                String str;
                str = this.time.toString();
                str = str + " (" + serviceType + ") ";

                return str;
        }
}
```

2. Be careful with your variable WashEnd. (Why?)
3. Manipulation of a queue, which hold single information is simple. However, queue which hold complex information such as class Customer is not easy to handle. Hence, we give you code example on how to manipulate this type of queue.

```
Clock i = new Clock();
Customer car = new Customer();
int nextArrival, carService;
Queue <Customer> myQueue = new Queue<>();
nextArrival = rand.nextInt(10); // nextArrival in the range 0 to 9
i.addTimeMinute(nextArrival);
car.time = i.getCopy();
car.serviceType = 2;
myQueue.enqueue(car.CopyCustomer());
if(arrivalQueue.peek().serviceType==2)
System.out.print("service type is 2- Wash Only" );
```

DEF Wash Machine simulation algorithm:

```
Set the operation time.
Generate all car arrivals and its service type, and push them onto arrival queue.
Set the initial state for the wash machine (status, startWash, endWash)
Set current event.
While there is still an event
    If car arrival event occurs:
        Push car onto waiting queue.
    If machine finish washing
        Update machine state.
    If machine available and there is a car waiting:
        Front car in waiting queue will be washed.
        Remove this car from waiting queue.
        Update the machine state and statistical information.
    If machine available and there is no car waiting:
        Update endWash to beyond operation time.
    Jump to the next earliest event that is either car arrival or machine finish washing.

Produce report.
```

Tutorial Activity

1. Determine the simulation characteristics of the DEF Wash Machine simulation.
2. What is the difference between ABC Wash Machine algorithm and DEF Wash machine algorithm?
3. What is the different between queue in ABC Wash Machine and queue in DEF Wash machine?
4. What are the modifications needed from the ABC Wash Machine code/program?