OBJECT ORIENTED PROGRAMMING WEEK – 03 ASSIGNMENT

Darshana pubudu keerthirathna

ICM 106   OR23106564

Question 01

```
class PriorityQueue{
	private Node front;

	public void enQueue(int data){
		Node n1 = new Node(data);
		if(isEmpty()){
			front = n1;
		}else{
			Node lastNode = front;
			while(lastNode.next!=null){
				lastNode=lastNode.next;
			}
			lastNode.next = n1;
		}
	}

	public void deQueue(){
		front = front.next;
	}

	public void printQueue(){
		Node temp = front;
		System.out.print("[");
		while(temp!=null){
			System.out.print(temp.data+", ");
			temp=temp.next;
		}
		System.out.println("\b\b]");
	}

	private boolean isEmpty(){
		return front==null;
```

```
        }



}


class Node{
        int data;
        Node next;
        Node(int data){
                this.data=data;
        }
}



class Demo{
        public static void main(String args[]){
                PriorityQueue pq=new PriorityQueue();
                pq.enQueue(12);
                pq.enQueue(90);
                pq.enQueue(16);
                pq.enQueue(45);
                pq.enQueue(96);
                pq.enQueue(23);
                pq.printQueue(); //[96, 16, 12, 90, 45, 23]
                pq.deQueue();
                pq.printQueue(); //[90, 16, 23, 45, 12]
                pq.deQueue();
                pq.printQueue(); //[45, 16, 23, 12]
        }
}
```

Question 02

```
class PatientQueue{
        private Node front;

        public void enQueue(Patient patient){
                Node n1 = new Node(patient);
                if(isEmpty()){
                        front = n1;
                }else{
                        Node lastNode = front;
                        while(lastNode.next!=null){
                                lastNode=lastNode.next;
                        }
                        lastNode.next = n1;
                }
        }

        public Patient deQueue(){
                Node temp = front;
                front = front.next;
                return temp.patient;
        }

        public void printQueue(){
                Node temp = front;
                System.out.print("{");
                while(temp!=null){
                        System.out.print("["+temp.patient.num+"-"+temp.patient.name+"], ");
                        temp=temp.next;
                }
                System.out.println(isEmpty()?"empty}":"\b\b]");
        }
```

```
    private boolean isEmpty(){

        return front==null;

    }


    public int size(){

        Node temp = front;

        int count = 0;

        while (temp!=null){

            count++;

            temp=temp.next;

        }

        return count;

    }


    public void clear(){

        front = null;

    }


}


class Node{

    Patient patient;

    Node next;

    Node(Patient patient){

        this.patient=patient;

    }

}


class Patient{

    int num;

    String name;

    Patient(int num, String name){

        this.num= num;
```

```
		this.name= name;
	}
	public String getPatientDetail(){
		String number = String.valueOf(num);
		return "["+num+"-"+name+"]";
	}
}


class Demo{
	public static void main(String args[]){
		PatientQueue queue=new PatientQueue();
		queue.enQueue(new Patient(101,"Amal"));
		queue.enQueue(new Patient(102,"Nimal"));
		queue.enQueue(new Patient(103,"Ramal"));
		queue.enQueue(new Patient(104,"Bimal"));
		queue.printQueue(); //{[101-Amal], [102-Niaml], [103-Ramal], [104-Bimal]}
		Patient firstPatient= queue.deQueue();
		System.out.println(firstPatient.getPatientDetail()); //[1001-Amal]
		queue.printQueue(); //{[102-Niaml], [103-Ramal], [104-Bimal]}
		System.out.println("No of patient of the queue : "+queue.size()); //3
		queue.clear();
		queue.printQueue(); //{Empty}
		System.out.println("No of patient of the queue : "+queue.size()); //0
	}
}
```

Question 03

```
class StudentList{
	private Node front;

	public void add(Student student){
		Node n1 = new Node(student);
		if(isEmpty()){
			front = n1;
		}else{
			Node lastNode = front;
			while(lastNode.next!=null){
				lastNode=lastNode.next;
			}
			lastNode.next = n1;
		}
	}

	public void add(int index,Student student){
		if(index>=0 && index<size()){
			Node temp=front;
			Node n1 = new Node(student);
			int count=0;
			while(count<index-1){
				temp=temp.next;
				count++;
			}
			n1.next=temp.next;
			temp.next=n1;

		}
	}

	public Student get(int index){
        if (index>=0 && index<size()){
                Node temp = front;
                int count = 0;
                while(count<index){
                        temp=temp.next;
                        count++;
                }
                return temp.student;
        }
        return null;
}

public Student remove(){
        Node temp = front;
        front = front.next;
        return temp.student;
}

public Student remove(int index){
        if (index>=0 && index<size()){
                Node temp = front;
                int count = 0;
                while(count<index-1){
                        temp=temp.next;
                        count++;
                }
                Node prvObj=temp;
                while(count<index){
                        temp=temp.next;
                        count++;
                }
                Node curObj = temp;
                prvObj.next=temp.next;
                return curObj.student;
            }
            return null;
    }

    public Student remove(Student student){
            if(student!=null){
                Node temp = front;
                int stuIndex = search(student);
                if (stuIndex!=-1){
                    Student stuObj = remove(stuIndex);
                    return stuObj;
                }
            }
            return null;
    }

    public int search(Student student){
            Node temp = front;
            int count =0;
            while(temp!=null){
                if(temp.student.code==student.code){
                    return count;
                }else{
                    temp=temp.next;
                    count++;
                }
            }
            return -1;
    }

    public void printList(){
            Node temp = front;
```

```
        System.out.print("{");
        while(temp!=null){
            System.out.print("["+temp.student.code+"-"+temp.student.name+"], ");
            temp=temp.next;
        }
        System.out.println(isEmpty()?"empty}":"\b\b}");
    }

    private boolean isEmpty(){
        return front==null;
    }

    public int size(){
        Node temp = front;
        int count = 0;
        while (temp!=null){
            count++;
            temp=temp.next;
        }
        return count;
    }

    public void clear(){
        front = null;
    }

}

class Node{
    Student student;
    Node next;
    Node(Student student){
        this.student=student;
```

```
	}
}

class Student{
	int code;
	String name;
	Student(int code, String name){
		this.code= code;
		this.name= name;
	}
	public String getStudentDetails(){
		String number = String.valueOf(code);
		return "["+code+"-"+name+"]";
	}
}


class Demo{
	public static void main(String args[]){
		StudentList stList=new StudentList();
		stList.add(new Student(1001,"Danapala"));
		stList.add(new Student(1002,"Gunapala"));
		stList.add(new Student(1003,"Somapala"));
		stList.add(new Student(1004,"Amarapala"));
		stList.add(new Student(1005,"Siripala"));
		stList.printList(); //{[1001-Danapala], [1002-Gunapala], [1003-Somapala], [1004-Amarapala], [1005-Siripala]}
		Student s1=stList.get(2);
		System.out.println("Student of index 2: "+s1.getStudentDetails()); //[1003-Somapala]

		Student s2= stList.remove(1);
		System.out.println("Last Removed Student: "+s2.getStudentDetails()); //[1002-Gunapala]
		stList.printList();//{[1001-Danapala], [1003-Somapala], [1004-Amarapala], [1005-Siripala]}
```

```
                stList.add(1,new Student(1000,"Gunapala"));

                stList.printList();//{[1001-Danapala],[1000-Gunapala], [1003-Somapala], [1004-Amarapala], [1005-Siripala]}


                int index= stList.search(new Student(1003,"Somapala"));

                System.out.println("Index of 1003 Somapala: "+index); //2


                index= stList.search(new Student(1111,"Somasiri"));

                System.out.println("Index of 1111,Somasiri "+index); //-1


                Student s3= stList.remove(new Student(1000,"Gunapala"));

                System.out.println("Last Removed Student: "+s3.getStudentDetails()); //[1000-Gunapala]

                stList.printList();//{[1001-Danapala], [1003-Somapala], [1004-Amarapala], [1005-Siripala]}


        }

}
```