Object Oriented Programming – Assignment 03 (Data Structures-Linked List)

CMJD90

Malshini Hansika

Rajarata University of Sri Lanka

Q1

```
class PriorityQueue{
   private Node front;


   public void enQueue(int data){
      Node n1=new Node(data);
      n1.next=front;
      front=n1;
      setPriority();
   }
   public void setPriority(){
      int max=front.data;
      int count=1;
      Node temp=front;
      while(temp!=null){
         if(temp.data>max){
            max=temp.data;
            count++;
            temp.data=front.data;
         }
         temp=temp.next;
      }
      front.data=max;
```

```
    }
    public void printQueue(){
        Node temp=front;
        System.out.print("[");
        while(temp!=null){
            System.out.print(temp.data+", ");
            temp=temp.next;
        }
        System.out.println(front==null ? "empty]":"\b\b]");
    }
    public void deQueue(){
        front=front.next;
        setPriority();
    }
    class Node{
       private int data;
       private 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]
    }
}
```

Q2

```
class Patient{
   private int data;
        private String name;
        Patient(int data, String name){
                    this.data=data;
      this.name=name;
             }
 public String getPatientDetail(){
   return "["+data+"-"+name+"]";
 }
}
class PatientQueue{
  private Node front;
  public void enQueue(Patient patient){
             Node n1=new Node(patient);
             if(front==null){
                    front=n1;
             }else{
                    Node lastNode=front;
                    while(lastNode.next!=null){
                            lastNode=lastNode.next;
                    }
                    lastNode.next=n1;
             }
  }
  public void printQueue(){
             System.out.print("{");
             Node temp=front;
             while(temp!=null){
                    System.out.print(temp.patient.getPatientDetail()+", ");
                    temp=temp.next;
             }
             System.out.println(front==null ? "Empty}":"\b\b}");
 }

  public int size(){
             Node temp=front;
             int count=0;
             while(temp!=null){
```

```
                        count++;
                        temp=temp.next;
                }
                return count;
        }
        public Patient deQueue(){
          Node temp=front;
          Patient p = temp.patient;
          if(front!=null){
                  front=front.next;
                }
                return p;
        }
        public void clear(){
                front=null;
        }
        class Node{
          private Patient patient;
          private Node next;
          Node(Patient patient){
                 this.patient=patient;
         }
        }
}

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
  }
}
```

Q3

```
class Student{
    private int code;
    private String name;
    public Student(int code, String name){
      this.code=code;
      this.name=name;
    }
    public String getStudentDetails(){
      return code+"-"+name;
    }
    public boolean equals(Student s1){
        return this.code==s1.code;
     }
 }
class StudentList{
    private Node start;

    public void add(Student st){
        Node n1=new Node(st);
        if(start==null){
            start=n1;
        }else{
            Node temp=start;
            while(temp.next!=null){
                temp=temp.next;
            }
            temp.next=n1;
        }
    }
    public void printList(){
     System.out.print("[");
        Node temp=start;
        while(temp!=null){
            System.out.print(temp.st.getStudentDetails()+", ");
            temp=temp.next;
        }
        System.out.println(start==null ? "Empty}":"\b\b]");
    }
```

```
public Student get(int index){
        if(start!=null){
                for (int i = 0; i < index; i++) {
                   start=start.next;
          }
        }
        Node temp=start;
        Student st=temp.st;
        return st;
}
public int search(Student st){
  if(start!=null){
          int i=0;
          Node temp=start;
          while(temp!=null){
                 if(st.equals(temp.st)){
                         return i;
                  }
                  i++;
          }
        }
        return -1;
}
public Student remove(int index){
 Node temp=start;
        Student st=temp.st;
```

```
    if(start!=null){
        for (int i = 0; i < index; i++) {
           start=start.next;
         }
    }
    return st;
}
public void add(int index,Student st){
    Node n1=new Node(index,st);
    if(start==null){
        start=n1;
    }else{
        Node temp=start;
        while(temp.next!=null){
            temp=temp.next;
        }
        temp.next=n1;
    }
}
public int size(){
    Node temp=start;
    int count=0;
    while(temp!=null){
        count++;
        temp=temp.next;
    }
    return count;
```

```
            }
            public Student remove(Student s){
                if(start!=null){
                            start=start.next;
                    }
            }
    class Node{
              private Student st;
              private Node next;
              private int index;
              Node(Student st){
                      this.st=st;
             }
            }
}
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();

   Student s1=stList.get(2);
   System.out.println("Student of index 2 : "+s1.getStudentDetails());
```

```
    Student s2=stList.remove(1);
    System.out.println("Last removed student : "+s1.getStudentDetails());
    stList.printList();

    stList.add(1,new Student(1000,"Gnanapala"));
    stList.printList();

    int index=stList.search(new Student(1003,"Somapala"));
    System.out.println("Index of 1003-Somapala : "+index);

    index=stList.search(new Student(1111,"Somasiri"));
    System.out.println("Index of 1111-Somasiri : "+index);

      Student s3=stList.remove(new Student(1000,"Gnanapala"));
      System.out.println("Last removed student : "+s3.getStudentDetails());
      stList.printList();
  }
}
```