Institute of Computer Engineering Technology

ASSIGNMENT

Assignement	WEEK 02 - Object Oriented Programming
Name	Data Structures - Stack and Queue
Ass. Date	02nd February 2024

01. Enhance the following class Stack, as to dynamically grow in capacity. (Hint – When the existing array is full, you can use a new array with double the capacity to complete this task). Implement the method named “capacity” of class Stack, this should return the size of the current array. Develop the method named “toArray” in the class Stack, which returns the existing data of the Stack in an array.

```
class Stack{
    private int nextIndex;
    private int[] dataArray;
    Stack(int size){
        dataArray=new int[size];
        nextIndex=0;
    }
    private boolean isEmpty(){
        return nextIndex<=0;
    }
    public void push(int data){
        dataArray[nextIndex++]=data;
    }
    public void printStack(){
        System.out.print("[");
        for(int i=nextIndex-1;i>=0;i--){
            System.out.print(dataArray[i]+", ");
        }
        System.out.println(isEmpty() ? "empty]": "\b\b]");
    }
    public void pop(){
        if(isEmpty()){
            System.out.println("Stack is empty...");
        }else{
            nextIndex--;
        }
    }
    public int size(){
        return nextIndex;
    }
    public void clear(){
        nextIndex=0;
    }
}
```

```
class Demo{
        public static void main(String args[]){
            Stack s1=new Stack(10); //Initial capacity of the stack is 10
            s1.printStack(); //[empty]
            System.out.println("Size of the stack is : "+s1.size()); //0
            System.out.println("Capacity of the stack is : "+s1.capacity()); //10
            s1.push(10);
            s1.push(20);
            s1.push(30);
            s1.push(40);
            s1.push(50);
            s1.printStack(); //[50, 40, 30, 20, 10]
            System.out.println("Size of the stack is : "+s1.size()); //5
            System.out.println("Capacity of the stack is : "+s1.capacity()); //10

            s1.push(60);
            s1.push(70);
            s1.push(80);
            s1.push(90);
            s1.push(100);
            s1.printStack(); //[100,90,80,70,60,50,40, 30, 20, 10]
            System.out.println("Size of the stack is : "+s1.size()); //10
            System.out.println("Capacity of the stack is : "+s1.capacity()); //10

            s1.push(111);
            s1.printStack(); //[111,100,90,80,70,60,50,40, 30, 20, 10]
            System.out.println("Size of the stack is : "+s1.size()); //11
            System.out.println("Capacity of the stack is : "+s1.capacity()); //20

            s1.push(222);
            s1.push(333);
            s1.push(444);
            s1.printStack(); //[444,333,222,111,100,90,80,70,60,50,40, 30,20,10]
            System.out.println("Size of the stack is : "+s1.size()); //14
            System.out.println("Capacity of the stack is : "+s1.capacity()); //20

            int[] ar=s1.toArray();
            for(int a : ar){
                System.out.print(a+" "); //444 333 222 111 100 90 80 70 60 50 40 30 20 10
            }
        }
}
```

02. Create a class called "PriorityQueue" using a fixed-length integer array to store integer data with the following functionalities. Always highest priority item is the first element of the Queue and no specific order for the other items.

```
class PriorityQueue{

}
class Demo{
    public static void main(String args[]){
        PriorityQueue pq=new PriorityQueue(10); //PriorityQueue(int initialSize)
        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]
    }
}
```

03. You are requested to create a class named PatientQueue, which holds entities of Patients as a structure of a queue. Additionally, you will have to create an entity class named Patient with attributes id - Integer and name – String.

```
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
    }
}
```

04. Class "Registry" is a collection of integers as a list. (Use an integer array to store data)

```
class Demo{
    public static void main(String args[]){
        Registry reg=new Registry(100); //
        reg.add(10);
        reg.add(20);
        reg.add(30);
        reg.add(40);
        reg.printRegistry();  //[10,20,30,40]

        reg.remove();          //remove the first element
        reg.printRegistry();  //[20,30,40]

        reg.add(1,25);//add(int index, int data)
        reg.printRegistry();  //[20,25,30,40]

        reg.add(new int[]{100,200,300,400});        //add(int[] data)
        reg.printRegistry();  //[20,25,30,40,100,200,300,400]

        reg.remove(1);   //remove(int index)
        reg.printRegistry();  //[20,30,40,100,200,300,400]

        reg.add(3,new int[]{1,2,3}); //add(int index, int[] data)
        reg.printRegistry();  //[20,30,40,1,2,3,100,200,300,400]

        reg.remove(3,6);     //remove(int startIndex, int endIndex-1)
        reg.printRegistry();  //[20,30,40,100,200,300,400]
    }
}
```