```
package july;
import java.util.*;

interface IntOperations{
	void even_odd();
	void pos_neg();
	void prime();
	void factorial();
	void sum();
}

class MyNumber implements IntOperations{
	private int n;
	MyNumber(int n){
		this.n=n;
	}
	MyNumber(){
		n=0;
	}
	public void even_odd() {
		if(n%2==0)
			System.out.println("Even");
		else
			System.out.println("Odd");
	}
	public void pos_neg(){
		if(n>0)
			System.out.println("Positive");
		else if(n<0)
			System.out.println("Negative");
		else
```

```
                System.out.println("Zero");
        }
        public void prime() {
                int flag=0;
                for(int i=2;i<(n/2);i++) {
                        if((n%i)==0) {
                                flag=1;
                                break;
                        }
                }
                if(flag==0)
                        System.out.println("Prime");
                else
                        System.out.println("Not Prime");
        }
        public void sum() {
                int n1,sum=0;
                while(n>0) {
                        n1=n%10;
                        sum+=n1;
                        n=n/10;
                }
                System.out.println("Sum of digits "+sum);
        }
        public void factorial() {
                int fact=1;
                for(int i=n;i>=1;i--)
                        fact*=i;
                System.out.println("Factorial "+fact);
        }
}
```

```
interface shape{
	void area();
}

class circle implements shape{
	double r,a;
	circle(double r){
		this.r=r;
	}
	public void area() {
		a=3.14*r*r;
		System.out.println("Area of circle "+a);
	}
}

class sphere implements shape{
	double r,a;
	sphere(double r){
		this.r=r;
	}
	public void area() {
		a=3.14*r*r*r;
		System.out.println("Area of sphere "+a);
	}
}

interface operation{
	double pi=3.14;
	void area();
	void volume();
```

```
}

class cylinder implements operation{
	Scanner sc=new Scanner(System.in);
	double h,r,a,v;
	cylinder(){
		System.out.println("Enter height, radius");
		h=sc.nextDouble();
		r=sc.nextDouble();
	}
	public void volume() {
		v=pi*r*r*h;
		System.out.println("Volume "+v);
	}
	public void area() {
		a=2*pi*r*h;
		System.out.println("Area "+a);
	}
}

interface marker{
	void show();
}

class product implements marker{
	Scanner sc=new Scanner(System.in);
	int id,qua;
	String name;
	double cost;
	product(){
```

```
		System.out.println("Enter product id, product name, product cost, product quantity");
		id=sc.nextInt();
		name=sc.next();
		cost=sc.nextDouble();
		qua=sc.nextInt();
	}
	public void show() {
		System.out.println("Product ID "+id+"\nProduct name "+name+"\nProduct cost "+cost+"\nProduct quantity "+qua);
	}
}

interface StackOperations{
	void push();
	void pop();
	void peek();
}

class stack implements StackOperations{
	Scanner sc=new Scanner(System.in);
	int top=2;
	int flag=1;
	int []a=new int[top];
	stack(){
		System.out.println("Enter "+top+" Values");
		for(int i=0;i<top;i++)
			a[i]=sc.nextInt();

	}
	public void push() {
		if(top<0 || flag==0) {
```

```
                top++;
                a=new int[top];
                int n=a.length;
                n--;
                System.out.println("Enter Element");
                a[n]=sc.nextInt();
                flag=1;
        }
        else {
        int i;
        int []b=new int[top];
        for(i=0;i<top;i++)
                b[i]=a[i];
        int n=a.length;
        top++;
        a=new int[top];
        for(i=0;i<top-1;i++)
                a[i]=b[i];
        System.out.println("Enter Element");
        a[n]=sc.nextInt();
        }
}
public void pop() {
        if(top<0 || flag==0)
                System.out.println("Stack is Empty");
        else {
        int i;
        int n=a.length;
        n--;
        System.out.println("Poped Element "+a[n]);
        int b[]=new int[top];
```

```
                for(i=0;i<top;i++)
                        b[i]=a[i];
                top--;
                if(top>0) {
                a=new int[top];
                for(i=0;i<top;i++)
                        a[i]=b[i];
                }
                else if(top==0) {
                        flag=0;
                }
                }
        }
        public void peek() {
                if(top<0 || flag==0)
                        System.out.println("Stack is Empty");
                else {
                int n=a.length;
                n--;
                System.out.println("Top Element "+a[n]);
                }
        }
}

interface card{
        void viewAmt();
        void viewPin();
        void changePin();
}

class customer implements card{
```

```
	Scanner sc=new Scanner(System.in);
	String name;
	int pin=1234,cardNo;
	double amt=0;
	public void viewAmt() {
		System.out.println("Amount "+amt);
	}
	public void viewPin() {
		System.out.println("Current pin "+pin);
	}
	public void changePin() {
		System.out.println("Enter new pin ");
		int npin=sc.nextInt();
		if(npin==pin)
			System.out.println("New pin cannot be same as old pin.");
		else
			pin=npin;
	}
}

class regular extends customer{
	double maxAmt=250000;
	regular(){
		amt=maxAmt;
	}
	public void use() {
		System.out.println("Enter amount ");
		double useAmt=sc.nextDouble();
		if(useAmt<=amt) {
			amt-=useAmt;
			System.out.println("Transaction successfull\nBalance "+amt);
```

```
                }
                else
                        System.out.println("Insufficient balance");
        }
        public void pay() {
                double pay=maxAmt-amt;
                if(pay>=0) {
                        System.out.println("Pay\n\t1.Total Amount due "+pay+"\t2.Current Amount
due "+(pay*0.50));
                        int ch=sc.nextInt();
                        switch(ch) {
                        case 1:
                                amt+=pay;
                                System.out.println("Transaction successfull\nBalance "+amt);
                                break;
                        case 2:
                                amt+=(pay*0.50);
                                System.out.println("Transaction successfull\nBalance "+amt);
                                break;
                        default:
                                System.out.println("You enter invalid option. Transaction
cancelled.");
                                break;
                        }
                }
                else
                        System.out.println("Your Credit Card has no pending payments.");
        }
}

class gold extends customer{
        String special="your max Limit is 500000";
```

```
	double maxAmt=500000;
	gold(){
		amt=maxAmt;
	}
	public void use() {
		System.out.println("Enter amount ");
		double useAmt=sc.nextDouble();
		if(useAmt<=amt) {
			amt-=useAmt;
			System.out.println("Transaction successfull\nBalance "+amt);
		}
		else
			System.out.println("Insufficient balance");
	}
	public void pay() {
		double pay=maxAmt-amt;
		if(pay>=0) {
			System.out.println("Pay\n\t1.Total Amount due "+pay+"\t2.Current Amount
due "+(pay*0.50));
			int ch=sc.nextInt();
			switch(ch) {
			case 1:
				amt+=pay;
				System.out.println("Transaction successfull\nBalance "+amt);
				break;
			case 2:
				amt+=(pay*0.50);
				System.out.println("Transaction successfull\nBalance "+amt);
				break;
			default:
				System.out.println("You enter invalid option. Transaction
cancelled.");
```

```
                                        break;
                                }
                        }
                        else
                                System.out.println("Your Credit Card has no pending payments.");
                }
}

interface que{
        void push();
        void remove();
        void peek();
}

class MyQueue implements que{
        Scanner sc=new Scanner(System.in);
        int front,rear,i;
        int a[];
        MyQueue(){
                front=0;
                rear=1;
                this.a=new int[rear+1];
                System.out.println("Enter "+(rear+1)+" values");
                for(i=front;i<=rear;i++)
                        a[i]=sc.nextInt();
        }

        public void push() {
                int b[]=new int[rear+1];
                for(i=front;i<=rear;i++)
                        b[i]=a[i];
```

```
                rear++;
                this.a=new int[rear+1];
                System.out.println("Enter Element ");
                a[rear]=sc.nextInt();
                for(i=front;i<rear;i++)
                        a[i]=b[i];
                for(i=front;i<=rear;i++)
                        System.out.println(a[i]);
        }
        public void remove() {
                if(rear<0)
                        System.out.println("Queue is empty");
                else {
                        System.out.println("Removed Element "+a[front]);
                        int b[]=new int[a.length];
                        for(i=0;i<a.length;i++)
                                b[i]=a[i];
                        rear--;
                        this.a=new int[rear+1];
                        for(i=front;i<b.length-1;i++)
                                a[i]=b[i+1];
                }
        }
        public void peek() {
                if(rear<0)
                        System.out.println("Queue is empty");
                else
                        System.out.println("Front Element "+a[front]);
        }
}
```

```
public class July_27 {

	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		/*System.out.println("Enter number");
		int num=sc.nextInt();
		MyNumber n=new MyNumber(num);
		n.even_odd();
		n.factorial();
		n.pos_neg();
		n.prime();
		n.sum();

		System.out.println("Enter radius");
		double r=sc.nextDouble();
		circle c=new circle(r);
		sphere s=new sphere(r);
		c.area();
		s.area();

		System.out.println("Enter n");
		int n=sc.nextInt();
		cylinder c[]=new cylinder[n];
		for(int i=0;i<n;i++) {
			c[i]=new cylinder();
			c[i].area();
			c[i].volume();
		}

		System.out.println("Enter n");
		int n=sc.nextInt();
```

```
product p[]=new product[n];
for(int i=0;i<n;i++) {
        p[i]=new product();
        p[i].show();
}

stack s=new stack();
int ch;
do {
System.out.println("1.Push\n2.Peek\n3.Pop\n4.Exit");
ch=sc.nextInt();
switch(ch) {
case 1:
        s.push();
        break;
case 2:
        s.peek();
        break;
case 3:
        s.pop();
        break;
case 4:
        System.exit(1);
        break;
default:
        System.out.println("Enter valid option");
        break;
}
}while(ch!=4);

int i,ch;
```

```
System.out.println("Enter n");
int n=sc.nextInt();
regular r[]=new regular[n];
gold g[]=new gold[n];
for(i=0;i<n;i++) {
        r[i]=new regular();
        g[i]=new gold();
}
do {
System.out.println("1.Regular user\n2.Gold user");
ch=sc.nextInt();
if(ch==1) {
        System.out.println("Out of "+n+" user which one are you ");
        int user=sc.nextInt();
        user--;
        System.out.println("Enter pin ");
        int pin=sc.nextInt();
        if(r[user].pin==pin) {
                System.out.println("1.Use card\n2.Pay Balance.\n3Change Pin");
                int choice=sc.nextInt();
                switch(choice) {
                case 1:
                        r[user].use();
                        break;
                case 2:
                        r[user].pay();
                        break;
                case 3:
                        r[user].changePin();
                        break;
                default:
```

```
                        System.out.println("Incorrect option");
                }
        }
        else
                System.out.println("Wrong pin");
}
else if(ch==2) {
        System.out.println("Out of "+n+" user which one are you ");
        int user=sc.nextInt();
        user--;
        System.out.println("Enter pin ");
        int pin=sc.nextInt();
        if(g[user].pin==pin) {
                System.out.println("Your speical privilege is, "+g[user].special);
                System.out.println("1.Use card\n2.Pay Balance.\n3Change Pin");
                int choice=sc.nextInt();
                switch(choice) {
                case 1:
                        g[user].use();
                        break;
                case 2:
                        g[user].pay();
                        break;
                case 3:
                        g[user].changePin();
                        break;
                default:
                        System.out.println("Incorrect option");
                }
        }
        else
```

```
                    System.out.println("Wrong pin");
}
else if(ch==3) {

}
else
        System.out.println("Invalid option");
}while(ch!=3);*/

MyQueue q=new MyQueue();
int ch;
do {
System.out.println("1.Push\n2.Peek\n3.Pop\n4.Exit");
ch=sc.nextInt();
switch(ch) {
case 1:
        q.push();
        break;
case 2:
        q.peek();
        break;
case 3:
        q.remove();
        break;
case 4:
        System.exit(1);
        break;
default:
        System.out.println("Enter valid option");
        break;
}
```

```
                }while(ch!=4);

        }

}
```