01 Object Oriented Programming in Java

01 Coding - Player Rating

```
//Create all the required classes here
import java.util.*;
abstract class Player{
   private String firstName, lastName;
   Player(String firstName, String lastName){
      this.firstName = firstName;
      this.lastName = lastName;
   }
   public String getName(){
      return firstName+" "+lastName;
   }
   abstract int getRating();
}
class CricketPlayer extends Player{
   private double averageRuns;
   CricketPlayer(String firstName, String lastName, double averageRuns){
      super(firstName, lastName);
      this.averageRuns = averageRuns;
   }
   public int getRating()
   {
      if(averageRuns>55)
      return 7;
      else if(averageRuns>50)
      return 6;
      else if(averageRuns>40)
      return 5;
      else if(averageRuns>30)
      return 3;
```

```
        else if(averageRuns>20)
        return 2;
        else
        return 1;
    }
}

class FootballPlayer extends Player{
    private int goals;
    FootballPlayer(String firstName, String lastName, int goals)
    {
        super(firstName, lastName);
        this.goals = goals;

    }
    public int getRating()
    {
        if(goals>20)
        return 5;
        else if(goals>15)
        return 4;
        else if(goals>10)
        return 3;
        else if(goals>5)
        return 2;
        else
        return 1;
    }
}

public class Source{
    public static void main(String[] args){
        //CODE HERE
```

```
        CricketPlayer cp = new CricketPlayer("Hitesh", "Chauhan", 76);
        FootballPlayer fp = new FootballPlayer("Hitesh", "Chauhan", 11);
        System.out.println(cp.getName()+"   "+cp.getRating());
        System.out.println(fp.getName()+"   "+fp.getRating());
    }
}
```

01 Coding - Account Details

```
//Editor is blank for both Java 7 and Java 8

import java.util.*;
class Account{
    int accountNo;
    double balance;
    String accountType;
    public static int counter;

    Account(double balance, String accountType)
    {
        counter++;
        this.balance=balance;
        this.accountType=accountType;
        this.accountNo=counter;
    }
    public String getInfo()
    {
        return accountNo+" "+balance+" "+accountType;
    }
    public void depositAmount(double amount){
        this.balance+=amount;
    }
```

```
    public void printAccountDetails(){
        System.out.println("[Acct No : "+accountNo+", Type : "+accountType+", Balance : "+balance+"]");
    }
    public void NewBalance(){
        System.out.println("New Balance : "+balance);
    }
}

class Source{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int n=2;
        while(n>0)
        {
            double balance=sc.nextDouble();
            String accountType = sc.next();
            double amount = sc.nextDouble();
            Account a = new Account(balance, accountType);
            a.printAccountDetails();
            a.depositAmount(amount);
            a.NewBalance();
            n--;
        }
    }
}
```

01 Coding - Customer Information

```
// Empty Editor

import java.util.*;
```

```
class SimpleDate{
private int day, month, year;
SimpleDate()
{
}
SimpleDate(int day, int month, int year)
{
this.day= day;
this.month = month;
this.year = year;
}
public int getDay()
{
return day;
}
public int getMonth()
{
return month;
}
public int getYear()
{
return year;
}
public void setDate(int day, int month, int year){
this.day = day;
this.month = month;
this.year = year;
}
@Override
public String toString(){
return day+"/"+month+"/"+year;
}
public static boolean validateDate(SimpleDate d){
```

```
if(d.getYear()>2000 && d.getMonth()>=1 && d.getMonth()<=12)
{
if(d.getMonth()==2 )
{
if(d.getDay()>=1 && d.getDay()<=28)
{
return true;
}
return false;
}
else if(d.getMonth()==1 || d.getMonth()==3 || d.getMonth()==5 || d.getMonth()==7 || d.getMonth()==8 ||
d.getMonth()==10 || d.getMonth()==12)
{
if(d.getDay()>=1 && d.getDay()<=31)
{
return true;
}
return false;
}
else{
if(d.getDay()>=1 && d.getDay()<=30)
{
return true;
}
return false;
}
}
return false;
}
}
class Address{
private String area, city;
Address()
```

```
{
}
Address(String area, String city){
this.area=area;
this.city=city;
}
public String getArea()
{
return area;
}
public String getCity()
{
return city;
}
public void setArea(String area)
{
this.area=area;
}
public void setCity(String city)
{
this.city=city;
}
public void setAddress(String area,String city){
this.area=area;
this.city=city;
}
@Override
public String toString(){
return area+", "+city;
}
}
class Customer{
private int custId;
```

```
private String name;
public Address address;
public SimpleDate registrationDate;
Customer(){
}
Customer(int custId, String name, Address address, SimpleDate registrationDate){
this.custId = custId;
this.name = name;
this.address = address;
// if(registrationDate.validateDate(registrationDate))
// this.registrationDate = registrationDate;
// else
// this.registrationDate = null;
this.registrationDate = registrationDate;
}
public Address getAddress(){
return address;
}
public SimpleDate getRegistrationDate(){
return registrationDate;
}
@Override
public String toString(){
if(registrationDate==null)
{
if(address.getArea()==null)
{
if(address.getCity()==null)
{
return "Id : "+custId+", Name : "+name+"\nAddress : "+"UNKNOWN"+", " +"UNKNOWN"+"\nRegistered on : "+"UNKNOWN";
}
return "Id : "+custId+", Name : "+name+"\nAddress : "+"UNKNOWN"+", " +address.getCity()+"\nRegistered on : "+"UNKNOWN";
```

```
}
return "Id : "+custId+", Name : "+name+"\nAddress : " +getAddress()+"\nRegistered on : UNKNOWN";
}
return "Id : "+custId+", Name : "+name+"\nAddress : " +getAddress()+"\nRegistered on :
"+getRegistrationDate();
}
}
class Source{
public static void main(String[] args){
Scanner sc = new Scanner(System.in);
int custId = sc.nextInt();
String name = sc.next();
sc.nextLine();
String area = sc.next();
String city = sc.next();
sc.nextLine();
int day = sc.nextInt();
int month = sc.nextInt();
int year = sc.nextInt();
SimpleDate sd = new SimpleDate(day,month,year);
Address a = new Address();
if(!area.isEmpty())
{
a.setArea(area);
}
else{
a.setArea(null);
}
if(!city.isEmpty())
{
a.setCity(city);
}
else{
```

```
a.setCity(null);
}
sd.setDate(day,month,year);
Customer c;
if(SimpleDate.validateDate(sd))
{
c = new Customer(custId, name, a, sd);
}else
c = new Customer(custId, name, a, null);
System.out.println(c);
}
}
```

01 Coding - Equality Check

```
class Customer {
 // STUDENT CODE BEGINS HERE
 private int customerId;
 private String name, city, phone, email;

 // Default constructor
 Customer(){
    this.customerId=0;
    this.name=null;
    this.city=null;
    this.phone=null;
    this.email=null;
 }

 //Parametrized constructor
 Customer(int customerId, String name, String city, String phone, String email)
 {
```

```
    this.customerId = customerId;
    this.name = name;
    this.city = city;
    this.phone = phone;
    this.email=email;
}

public void setCustomerId(int customerId){
    this.customerId = customerId;
}
public void setName(String name)
{
    this.name=name;
}
public void setCity(String city)
{
    this.city=city;
}
public void setPhone(String phone)
{
    this.phone = phone;
}
public void setEmail(String email)
{
    this.email = email;
}
public int getCustomerId(){
    return customerId;
}
public String getName(){
    return name;
}
public String getCity(){
```

```
    return city;
  }
  public String getPhone(){
    return phone;
  }
  public String getEmail(){
    return email;
  }

  @Override //Equals method
  public boolean equals(Object obj) {
    if (obj == this) {
        return true;
     }
     if (obj == null || obj.getClass() != this.getClass()) {
        return false;
     }
    Customer c = (Customer) obj;
    if(c.getCustomerId()==this.getCustomerId() && c.getName()==(this.getName()) &&
c.getCity()==(this.getCity()) && c.getPhone()==(this.getPhone()) && c.getEmail()==(this.getEmail()) && c!=null
&& this!=null){
       return true;
    }
    return false;
  }
  // STUDENT CODE ENDS HERE
}

class Source {
  public static void main(String []args){
  //STUDENT CODE HERE
  Customer c1 = new Customer(1, "Vinod", "Bangalore", null, null);
   Customer c2 = new Customer(1, "Vinod", "Bangalore", null, null);
   System.out.println(c1.equals(c2));
```

```
    System.out.println(c2.equals(c1));
    c1.setEmail("vinod@mailinator.com");
    System.out.println(c1.equals(c2));
    System.out.println(c2.equals(c1));
    c2.setEmail("vinod@mailinator.com");
    System.out.println(c1.equals(c2));
    System.out.println(c2.equals(c1));
    Customer c3 = new Customer();
    Customer c4 = new Customer();
    System.out.println(c3.equals(c4));
  }
}
```

01 Coding -Fahrenheit to Celsius

```
import java.util.*;
class Utility{
    public static int fahrenheitToCelcius(double fahrenheit){
        double celcius = ((fahrenheit-32)*5)/9;
        celcius = Math.round(celcius);
        int n = (int) celcius;
        return n;
    }

    public static String getLevel(int[] arr){
        int sum=0;
        for(int i=0;i<arr.length;i++){
            sum+=arr[i];
        }
        if(sum>=100)
        {
        return "HIGH";
```

```
        }
        else if(sum>=70){
        return "MEDIUM";
        }
        return "LOW";
    }
}

class Source{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        if(n==1)
        {
            int temp = sc.nextInt();
            System.out.println(Utility.fahrenheitToCelcius(temp));
        }
        else if(n==2)
        {
            int m = sc.nextInt();
            int arr[] = new int[m];
            for(int i=0;i<m;i++)
            {
                arr[i] = sc.nextInt();
            }
            System.out.println(Utility.getLevel(arr));
        }
    }
}
```

01 Coding -Shape Hierarchy

```
import java.util.*;
interface Shape{
   double getArea();
   double getPerimeter();
}
abstract class AbstractShape implements Shape{
   private String colour;

   AbstractShape(String colour){
      this.colour=colour;
   }
   // public String getColour(){
   //    return colour;
   // }

   public String toString()
   {
      return "colour="+colour;
   }
}

class Rectangle extends AbstractShape{
   private int length, breadth;

   Rectangle(String colour, int length, int breadth){
      super(colour);
      this.length = length;
      this.breadth = breadth;
   }

   public double getArea(){
```

```
        return length*breadth;
    }
    public double getPerimeter(){
        return 2*(length+breadth);
    }

    public String toString(){
        return "Rectangle ["+super.toString()+", length="+length+", breadth="+breadth+"]";
    }
}

class Circle extends AbstractShape{
    private int radius;

    Circle(String colour, int radius){
        super(colour);
        this.radius = radius;
    }
    public double getArea(){
        return Math.PI*radius*radius;
    }
    public double getPerimeter(){
        return 2*Math.PI*radius;
    }
    public String toString(){
        return "Circle ["+super.toString()+", radius="+radius+"]";
    }
}
class Source{
    public static void printShape(AbstractShape obj){
        if(obj instanceof Rectangle)
        {
            Rectangle r = (Rectangle) obj;
```

```
        System.out.println("Area:"+Math.round(r.getArea())+", Perimeter:"+Math.round(r.getPerimeter()));

    }
    if(obj instanceof Circle){
        Circle c = (Circle) obj;
        System.out.println("Area:"+Math.round(c.getArea())+", Perimeter:"+Math.round(c.getPerimeter()));
    }
}
public static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    String shape = sc.next();
    if(shape.equals("CIRCLE")){
        String colour = sc.next();
        int radius = sc.nextInt();
        AbstractShape as = new Circle(colour, radius);
        Circle c = new Circle(colour, radius);
        System.out.println(c);
        printShape(as);


    }
    if(shape.equals("RECTANGLE")){
        String colour = sc.next();
        int length = sc.nextInt();
        int breadth = sc.nextInt();
        AbstractShape as = new Rectangle(colour, length, breadth);
        Rectangle r = new Rectangle(colour, length, breadth);
        System.out.println(r);
        printShape(as);

    }

}
```

}

01 Coding - Product Analysis

```
import java.util.*;
//DO NOT EDIT THIS CLASS
class Product {
	private int prodCode;
	private String prodName;
	private double price;
	private String category;

	public Product(int prodCode, String prodName, double price, String category) {
		this.prodCode = prodCode;
		this.prodName = prodName;
		this.price = price;
		this.category = category;
	}

	public int getProdCode() {
		return prodCode;
	}

	public void setProdCode(int prodCode) {
		this.prodCode = prodCode;
	}

	public String getProdName() {
		return prodName;
	}
```

```
	public void setProdName(String prodName) {
		this.prodName = prodName;
	}

	public double getPrice() {
		return price;
	}

	public void setPrice(double price) {
		this.price = price;
	}

	public String getCategory() {
		return category;
	}

	public void setCategory(String category) {
		this.category = category;
	}
}


//DO NOT EDIT THIS CLASS
class ProductData {

	private static Product[] products;

	static {
		products = new Product[8];
		products[0] = new Product(101, "keyboard", 300, "computers");
		products[1] = new Product(102, "mouse", 600, "computers");
		products[2] = new Product(103, "monitor", 5000, "computers");
		products[3] = new Product(104, "t-shirt", 500, "clothing");
```

```
                products[4] = new Product(105, "jeans", 2000, "clothing");
                products[5] = new Product(106, "sweater", 1000, "clothing");
                products[6] = new Product(107, "doll", 500, "toys");
                products[7] = new Product(108, "car", 1000, "toys");
        }

        public static Product[] getProducts() {
                return products;
        }
}



class ProductService
{
   //CODE HERE
   public static String findNameByCode(int prodcode){
      Product[] myproducts = ProductData.getProducts();
      for(Product p: myproducts){
         if(p.getProdCode()==prodcode)
         return p.getProdName();
      }
      return null;

   }
   public static Product findMaxPriceProduct(String Category){
      Product[] myproducts = ProductData.getProducts();
      Product result=null;
      double price=0;
      for(Product p:myproducts)
      {
         if(Category.equalsIgnoreCase(p.getCategory()) && price<p.getPrice()){
            result = p;
            price=p.getPrice();
```

```
        }
      }
      return result;
   }
}

public class Source{
   public static void main(String [] args){
      //CODE HERE
      Scanner sc = new Scanner(System.in);
      int n = sc.nextInt();
      // Product[] mypro = ProductData.getProducts();
      // System.out.println(mypro[0]);
      if(n==1)
      {
         int prodcode = sc.nextInt();
         String result = ProductService.findNameByCode(prodcode);
         if(result!=null)
         {
         System.out.println(result);
         }
         else{
            System.out.println("Product Not Found");
         }
      }
      else if(n==2)
      {
         String category = sc.next();
         Product p = ProductService.findMaxPriceProduct(category);
         if(p!=null)
         {
         System.out.println(p.getProdCode()+" | "+p.getProdName()+" | "+p.getPrice());
         }
```

```
        else{
            System.out.println("No products in given category");
        }
      }
      else{
        System.out.println("Invalid choice");
      }
   }
}
```

02 - Useful Utility Classes

02 Coding - Colour Code Validator

```
import java.util.*;
import java.util.regex.*;
class Source{
   public static int validateHexCode(String code){
      String regex = "^#([A-F0-9]{6})$";
      Pattern p = Pattern.compile(regex);
      Matcher m = p.matcher(code);
      boolean flag = m.matches();
      if(flag)
      return 1;
      return -1;

   }
   public static int validateDecimalCode(String s) {
   if (s.startsWith("rgb") == false) return -1;
```

```
    s = s.substring(4, s.length() - 1);
    String str[] = s.split(",");
    if (str.length != 3) return -1;
    try {
        int x = Integer.parseInt(str[0]);
        if (x < 0 || x > 255) return -1;
        x = Integer.parseInt(str[1]);
        if (x < 0 || x > 255) return -1;
        x = Integer.parseInt(str[2]);
        if (x < 0 || x > 255) return -1;
    } catch (Exception e) {
        return -1;
    }
    return 1;
}

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        if(n==1){
            String code = sc.next();
            int m = validateHexCode(code);
            if(m==1){
                System.out.println("Valid Code");
            }
            else{
                System.out.println("Invalid Code");
            }

        }
        else if(n==2){
            String code = sc.next();
            int m = validateDecimalCode(code);
```

```
        if(m==1){
            System.out.println("Valid Code");
        }
        else{
            System.out.println("Invalid Code");
        }

    }
    else{
        System.out.println("Invalid choice");
    }
  }
}
```

02 Coding - Lucky Registration Number

```
import java.util.*;
import java.util.regex.*;
class Source{
   public static int checkRegistrationNumber(String num){
      if(num!=null){
      String regex = "(KA|DL)(01|02|03|04|05|06|07|08|09|10)[A-Z]{1,2}[1-9]{1}[0-9]{3}$";
      Pattern p = Pattern.compile(regex);
      Matcher ma = p.matcher(num);
      boolean flag = ma.matches();
      if(flag)
      {
         String s= num.substring(num.length()-4, num.length());
         int m = Integer.parseInt(s);
         // System.out.println(m);
         int sum=0;
         while(m>0 || sum>9){
```

```
            if(sum>9 && m==0)
            {
               m=sum;
               sum=0;
            }
            int rem = m%10;
            sum+=rem;
            m/=10;
         }
         if(sum==6)
         {
            return 1;
         }
         else{
            return 0;
         }
      }
      return -1;
      }
      else
         return -1;
   }
   public static void main(String[] args)
   {

      Scanner sc = new Scanner(System.in);
      String num = sc.next();
      int n = checkRegistrationNumber(num);
      if(n==1)
      {
            System.out.println("Lucky registration number");
         }
         else if(n==0)
```

```
        {
            System.out.println("Valid registration number");
        }
      else{
          System.out.println("Invalid registration number");
      }
   }
}
```

02 Coding - Calculate Age

```
import java.time.*;
import java.util.*;
class AgeCalculator{
   public int[] calculateAge(String str){
      int arr[] = new int[2];
      String []date = str.split("/");
      int day = Integer.parseInt(date[0]);
      int month = Integer.parseInt(date[1]);
      int year = Integer.parseInt(date[2]);
      if(year<=2019){
            if(year==2019 && month>=4){
            }
            else{
         if(month>4){
            arr[0] = 2019-year-1;
            arr[1] = 16-month;

         }
         else{
            arr[0] = 2019-year;
            arr[1] = 4-month;
```

```
            }
            return arr;
        }
        }
        return null;
    }
}
class Source{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        String date = sc.nextLine();
        AgeCalculator ac = new AgeCalculator();
        int arr[] = ac.calculateAge(date);
        boolean flag = true;
        if(arr!=null){
        if(arr[0]!=0)
        {
            if(arr[1]==0){
                System.out.print("Years : "+arr[0]);
                flag = false;
            }
            else{
            System.out.print("Years : "+arr[0]+", ");
            flag = true;
            }
        }
        if(flag){
            System.out.print("Months : "+arr[1]);
        }
        }
        else{
            System.out.println("Invalid date of birth");
        }
```

```
    }
}
```

02 Coding - Count Unique Characters

```
import java.util.Scanner;

public class Source {
    public static int getUniqueCharacterCount(String str){
        str = str.replaceAll("\\s", "");
        str = str.toLowerCase();
        // System.out.println(str);
        int count=0;
        // System.out.println(str.length());
        for(int i=0;i<str.length();i++){
            boolean flag = false;
            for(int j=0;j<str.length();j++){
                if(str.charAt(i)==str.charAt(j) && i!=j)
                {
                    flag = true;
                    break;
                }
            }
            if(!flag)
            {
                // System.out.println(str.charAt(i));
                count++;
            }
        }
        if(count>0)
        {
```

```
        return count;
    }
    return -1;
  }
        public static void main(String[] args) {
                //write code here
                Scanner sc = new Scanner(System.in);
                String word = sc.nextLine();
                int n = getUniqueCharacterCount(word);
                if(n==-1)
                {
                   System.out.println("No unique character/s");
                }
                else{
                   System.out.println(n+" unique character/s");
                }
         }

        //write method here

}
```

02 Coding -Day of Date

```
//import statements here
import java.util.*;
import java.text.*;
public class Source {
        public static void main(String[] args) throws ParseException {
           //CODE HERE
           Scanner sc = new Scanner(System.in);
           String str = sc.next();
```

```
            String day = getDayOfDate(str);
            System.out.println(day);
        }

public static String getDayOfDate(String date) throws ParseException {
                //CODE HERE
                Date date1=new SimpleDateFormat("dd/MM/yyyy").parse(date);
                SimpleDateFormat sdf = new SimpleDateFormat("EEEE");
                String stringDate = sdf.format(date1);
                return stringDate;
        }
}
```

02 Coding - Max Digit in a String

```
import java.util.Scanner;
import java.util.regex.*;
public class Source{
   // Code here
   public static int getMaxDigit(String str){
      if(str!=null )
      {
      Pattern p = Pattern.compile("\\d+");
      Matcher m = p.matcher(str);
      int max = 0;
      String digits = "";
      while(m.find()){
         digits+=m.group();
      }
      // System.out.println(digits);
      if(digits.length()>1){
```

```
    String arr[] = digits.split("");
    for(int i=0;i<arr.length;i++)
    {
        if (max<Integer.parseInt(arr[i]))
        {
            max = Integer.parseInt(arr[i]);
        }
    }
    if(max>0)
    {
        return max;
    }
    if(digits.isEmpty()){
        return 0;
    }
    return 0;
    }
    return -1;

    }
    return -1;
}
public static void main(String[] args){
    //Code here
    Scanner sc = new Scanner(System.in);
    String str = sc.nextLine();
    int n = getMaxDigit(str);
    if(n==0 || n==-1)
    {
        System.out.println("No digits in string");
    }
    else{
        System.out.println(n);
```

```
        }
    }
}
```

02 Coding - Message Encryption

```
import java.util.*;
class Source{
    public static String encrypt(String message){
        message = message.toLowerCase();
        char arr[] = message.toCharArray();
        for(int i=0;i<arr.length;i++){
            if(arr[i]=='a'){
                arr[i] = 'e';
            }
            else if(arr[i]=='e'){
                arr[i] = 'a';
            }
            else if(arr[i]=='o'){
                arr[i] = 'u';
            }
            else if(arr[i]=='u'){
                arr[i] = 'o';
            }
            else if(arr[i]=='i'){
                arr[i] = '!';
            }
            else if(arr[i]==' '){
                arr[i] = '+';
            }
            else if(arr[i]=='z'){
```

```
            arr[i] = 'b';
        }
        else if(arr[i]=='d'){
            arr[i] = 'f';
        }
        else if(arr[i]=='h'){
            arr[i] = 'j';
        }
        else if(arr[i]=='n'){
            arr[i] = 'p';
        }
        else if(arr[i]=='t'){
            arr[i] = 'v';
        }
        else if(arr[i]=='9'){
            arr[i] = '0';
        }
        else if(arr[i]>=97 && arr[i]<=122)
        {
            arr[i]+=1;
        }
        else if(arr[i]>=48 && arr[i]<=57)
        {
            arr[i]+=1;
        }
    }
    String result = "";
    for(int i=0;i<arr.length;i++)
    {
        result+=arr[i];
    }
    return result;
}
```

```
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        String message = sc.nextLine();
        String result = encrypt(message);
        System.out.println(result);
    }
}
```

03 Exception Handling and Utilities

03 Coding - Multiple Catch

```
import java.util.*;
class Sequence{
    public static int sequences(String str){
        String s[] = str.split(",");
        int arr[] = new int[s.length];
        int sum=0;
        try{
        for(int i=0;i<s.length;i++){
            arr[i] = Integer.parseInt(s[i]);
        }
        int n[] = new int[arr.length];
        int j=1;
        int m = arr.length;
        while(j<m){
            for(int i=0;i<n.length-1;i++){
                arr[i] = arr[i+1]-arr[i];
                n[i] = arr[i];
                // sum+=n[i];
```

```
            // System.out.print(n[i]+" ");
          }
          // System.out.println();
          n = new int[n.length-1];
          j++;
        }
        }
        catch(NumberFormatException e)
        {
          throw e;
        }
        return arr[0];
    }
}
class Source{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        String str;
        try{
        str = sc.next();
        int result = Sequence.sequences(str);
        System.out.println(result);
        }
        catch(ArrayIndexOutOfBoundsException e){
        }
        catch(NumberFormatException e){

        }

    }
}
```

03 Coding -Date Validation

```
import java.util.Scanner;
import java.text.ParseException;
import java.text.SimpleDateFormat;

public class Source {
   public static void main(String[] args) {
            // STUDENT CODE BEGINS HERE
      Scanner sc = new Scanner(System.in);
      String date = sc.next();
      int n = Utility.checkDate(date);
      if(n==1){
         System.out.println("Valid");
      }
      else{
         System.out.println("Invalid");
      }
     // STUDENT CODE ENDs HERE
   }
}


class Utility {
          // STUDENT CODE BEGINS HERE
          public static int checkDate(String dateValue){
             int returnVal = -1;
      String[] permissFormats = new String[]{"dd/MM/yyyy","dd-MM-yyyy","dd.MM.yyyy"};
      for (int i = 0; i < permissFormats.length; i++)
      {
         try
         {
```

```
            SimpleDateFormat sdfObj = new SimpleDateFormat(permissFormats[i]);
            sdfObj.setLenient(false);
            sdfObj.parse(dateValue);
            returnVal = 1;
            break;
          }
          catch(ParseException e)
          {
          }
        }
        return returnVal;
            }
    // STUDENT CODE ENDs HERE
}
```