Course: JAVA **Session:** Methods **Timestamp:** 2021-8-31 10:28:22 **Register Number:** RA2031241010094

Q. Passkey

Pass key: To do the following operations on given input by ignoring the alphabets and special chars in between :

Input=34#2a3
Expected Output is =9

Note that input may have alphabets and special chars in between:

How to proceed further-1:

Find largest digit in given input: largest digit=4 then find the sum of the multiples of largest digit with all individual input digits (Note:Answer should be in single digit):

A=(4*3)+(4*4)+(4*2)+(4*3)=48

We need to convert this sum into single digit sum.
48==> (4+8)=sum = 12 ==> (1+2)=3
call the above sum as sum1;

How to proceed further-2:
Find smallest digit in given input:smallest digit=2 then multiply the numbers which are formed by adding smallest digit with all individual input digits
(Note:Answer should be in single digit):

that is =(2+3)*(2+4)*(2+2)*(2+3)=600=(6+0+0)=6
call the above sum as sum2;

How to proceed further-3:
Find the smallest among sum1 and sum2. min(3,6)=3
digit3=min(sum1,sum2)=3

How to proceed further-4:
Calculate the Output as Square_of digit3
that is , Square(3)= 9

Mandatory:

1. Create a class named "Passkey" and inside "Passkey" class do the following:

a. Method name= passKey()
b. Type of method = void
c. Access specifiers = public
d. Argument = No argument

2. Create an object for Passkey class inside the main method as follows:

a. Objectname = s1
b. Call the method "passKey()" as s1.passKey()

Source Code

```
import java.util.*;
class Passkey
{
  public void passKey()
  {
    Scanner sc=new Scanner(System.in);
    int i;
    String str=stripNonDigits(sc.nextLine());
    char[] num=str.toCharArray();
    int min=0,max=0;
    for(i=0;i<num.length;i++)
    {
      if(max<Character.getNumericValue(num[i]))
      {
        max=Character.getNumericValue(num[i]);
      }
    }
    min=max;
    for(i=0;i<num.length;i++)
    {
      if(min>Character.getNumericValue(num[i]))
      {
        min=Character.getNumericValue(num[i]);
      }
    }
    int a=0,n,sum=0;
    for(i=0;i<num.length;i++)
    {
      a=a+(max*Character.getNumericValue(num[i]));
    }
    while(a>0)
    {
      n=a%10;
      sum=sum+n;
      a=a/10;
    }
    while(sum>0)
    {
      n=sum%10;
      a=a+n;
      sum=sum/10;
    }
    int b=1;
    n=0;sum=0;
    for(i=0;i<num.length;i++)
    {
      b=b+(min*Character.getNumericValue(num[i]));
    }
    while(b>0)
    {
      n=b%10;
      sum=sum+n;
      b=b/10;
    }
    b=sum;
    if(a>b)
    {
      System.out.println(b*b);
    }
    else
    {
      System.out.println(a*a);
    }
   }
    public static String stripNonDigits(final CharSequence input)
    {
      final StringBuilder sb=new StringBuilder(input.length());
      int i;
      for(i=0;i<input.length();i++)
      {
        final char c=input.charAt(i);
        if(c>47&&c<58){
          sb.append(c);
      }
    }
    return sb.toString();
}
}
public class TestClass {
  public static void main(String[] args) {
  Passkey s1=new Passkey();
      s1.passKey();
 }
}
```

Sample Input

34#2a3

Sample Output

9

Result

Thus, Program " **Passkey** " has been successfully executed

Course: JAVA **Session:** Methods **Timestamp:** 2021-8-31 10:28:44 **Register Number:** RA2031241010094

Q. Check Valid Password

Write a Java method to check whether a string is a valid password.

Password rules:
A password must have at least ten characters.
A password consists of only letters and digits.
A password must contain at least two digits

Mandatory:

1. Create method under main class (TestClass)
2. The other methods in the main class should be a static methods
3. Use boolean Value in the method and if the boolean method returns true then print "Password is valid" else print "Not a valid password"

Source Code

```
import java.util.*;
public class TestClass {
  static boolean pass(String str)
  {
    int i,j=str.length(),c=0;
    for(i=0;i<j;i++)
    {
      if((str.charAt(i)=='0'||str.charAt(i)=='1'||str.charAt(i)=='2'||str.charAt(i)=='3'||str.charAt(i)=='4'||str.charAt(i)=='5'||str.charAt(i)=='6'||str.charAt(i)=='7'||str.charAt(i)=='8'||str.charAt(i)=='9'))
          c++;
    }
          if(c>=2&&j>=10)
          return true;
          else
          return false;
          }
  public static void main(String[] args) {
       Scanner sc=new Scanner(System.in);
       String str=sc.nextLine();
       if(TestClass .pass(str))
         System.out.println("Password is valid");
       else
         System.out.println("Not a valid password");

  }
}
```

Sample Input

12qwetrtyujhjg5

Sample Output

Password is valid

Result

Thus, Program " **Check Valid Password** " has been successfully executed

Course: JAVA **Session:** Methods **Timestamp:** 2021-8-31 10:29:02 **Register Number:** RA2031241010094

Q. Palindrome using static method and variable

Given the string, check if it is a palindrome or not using class and objects concept

Mandatory:

1. Create a class "Sample" with two static methods

2. Method 1

a. Method Name = getInput()
b.Access Specifier = static
c. Argument = No Arguments
d. Method Return type = void
e. Use = To get the String input and and pass the string input to checkPalindrome() method

3. Method 2

a. Method Name = checkPalindrome(str)
b.Access Specifier = static
c. Argument = Single Argument with argument type String
d. Method Return type = void

4. Access the method getInput() under class "Sample" from the main method using classname.methodname

5. Access the method checkPalindrome(str) from the getInput() method itself

Note: Do not access checkPalindrome(str) from the main method, it needs to be accessed only from getInput() method and the argument variable name is "str".

Source Code

```
import java.util.*;
class Sample
{
  static String str;
  static void getInput()
  {
    String sr=str;
  }
  static void checkPalindrome(String str)
  {
    int i,j=str.length()-1;
    String rev="";
    for(i=j;i>=0;i--)
    {
      rev=rev+str.charAt(i);
    }
    if(str.equals(rev))
      System.out.println(str+" is a palindrome");
    else
      System.out.println(str+" is not a palindrome");
  }
}
public class TestClass {
  public static void main(String[] args) {
       Scanner sc=new Scanner(System.in);
       String str=sc.nextLine();
       Sample methodname=new Sample();
       Sample.getInput();
       Sample.checkPalindrome(str);

  }
}
```

Sample Input

madam

Sample Output

madam is a palindrome

Result

Thus, Program " **Palindrome using static method and variable** " has been successfully executed

Course: JAVA **Session:** Methods **Timestamp:** 2021-8-31 10:29:31 **Register Number:** RA2031241010094

Q. Determinant

Write a java method to find the determinant of given matrix.

Mandatory:

1.Under the Main Class(TestClass) create the following methods to get the input values and to calculate the determinant respectively.

2. Method Details:

a. Method Name = input ()
b. Access Specifier = public
c. Argument = No argument
d. Return type = void

public void input()

a. Method Name =determinant ()
b. Access Specifier = public
c. Argument = Two Arguments of type integer
d. Return type = void

public int determinant(int A[][],int N)

3.Create an object "d" for the main class in the main method and access input() and determinant() method from the main method to print the result of the determinant calculation using System.out.println class.

Note:
If all the above conditions are not satisfied then it wont show 100%

Source Code

```
import java.util.*;
public class TestClass {
  public int a[][],n;
  public static void input()
  {
  }
  public int determinant(int a[][],int n)
  {
    int i=0;
    if(n==1)
    {
      i=a[0][0];
    }
    else if(n==2)
    {
      int j=a[0][0]*a[1][1]-a[0][1]*a[1][0];
      i=j;
    }
    else if(n==3)
    {
      int k=a[0][0]*(a[1][1]*a[2][2]-a[1][2]*a[2][1]);
      int l=a[0][1]*(a[1][0]*a[2][2]-a[1][2]*a[2][0]);
      int m=a[0][2]*(a[1][0]*a[2][1]-a[1][1]*a[2][0]);
      i=k-l+m;
    }
    return i;
  }

  public static void main(String[] args)
    {
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int[][] a=new int[n][n];
      for(int c=0;c<n;c++)
      {
        for(int d=0;d<n;d++)
        {
          a[c][d]=sc.nextInt();
        }
      }
     TestClass d=new TestClass();
      d.input();
      System.out.println(d.determinant(a,n));

  }
}
```

Sample Input

```
3
6 1 1
4 -2 5
2 8 7
```

Sample Output

```
-306
```

Result

Thus, Program " **Determinant** " has been successfully executed

Course: JAVA **Session:** Methods **Timestamp:** 2021-8-31 10:29:49 **Register Number:** RA2031241010094

Q. FrontBack

Given 2 integer a and b, return true if one of them is 10 or if their sum is 10.

Mandatory:

1. Create a class "Sample" and a method named "public void getNumbers()" with no arguments . This is used only to get two integer numbers
2. Create an another method inside "Sample" class titled as makes10() with two arguments and the type of the method is boolean
3 Print the final answer in the main method under main class (TestClass) as follows:

```
boolean output=objectname.makes10()
System.out.println(output)
```

Source Code

```
import java.util.*;
class Sample
{
  public int a,b;
  public void getNumbers()
  {
  }
  public boolean makes10(int a,int b)
  {
    if((a==10||b==10)||(a+b==10))
    return true;
    else
      return false;
  }
}
public class TestClass {
  public static void main(String[] args)
     {
       Scanner sc=new Scanner(System.in);
       int a=sc.nextInt();
       int b=sc.nextInt();
       Sample objectname=new Sample();
       objectname.getNumbers();
       boolean output=objectname.makes10(a,b);
       System.out.println(output);

 }
}
```

Sample Input

9 10

Sample Output

true

Result

Thus, Program " **FrontBack** " has been successfully executed

Course: JAVA **Session:** Methods **Timestamp:** 2021-8-31 10:30:38 **Register Number:** RA2031241010094

Q. Fibonacci sequence

The Fibonacci sequence is defined by the following rule. The first 2 values in the sequence are 1, 1. Every subsequent value is the sum of the 2 values preceding it.

Write a Java program that uses both recursive and non recursive functions to print the n th value of the Fibonacci sequence.

Mandatory :

1.Create a new class and two methods as follows

a. Method name = getInput()
b. Type = void
c. Access specifiers = public
d. Arguments = No argument

a. a. Method name = displayFib()
b. Type = void
c. Access specifiers = public
d. Arguments = No argument

2. Create object for above mentioned class in the main method and access the methods getInput() and displayFib()

Source Code

```
import java.util.*;
public class TestClass {
  public int n;
  public void getInput()
  {
  }
  public void displayFib()
  {
    Scanner sc=new Scanner(System.in);
    int n=sc.nextInt();
    int n1=1,n2=1,n3,i;
    System.out.print(n1+ " "+n2);
    for(i=2;i<n;i++)
    {
      n3=n1+n2;
      System.out.print(" "+n3);
      n1=n2;
      n2=n3;
    }
  }
  public static void main(String[] args) {
       TestClass obj=new TestClass();
       obj.getInput();
       obj.displayFib();

  }
}
```

Sample Input

5

Sample Output

1 1 2 3 5

Result

Thus, Program " **Fibonacci sequence** " has been successfully executed

Course: JAVA **Session:** Methods **Timestamp:** 2021-8-31 10:31:02 **Register Number:** RA2031241010094

Q. Difference of adjacent elements

Given an array of integers, find the pair of adjacent elements that has the largest difference and return that difference.

Example

For inputArray = [3, 6, -2, -5, 7, 3], the output should be

adjacentElementsDifference(inputArray, lenght) = 8. 6 and -2 produce the largest difference.

Input Format:
An array of integers containing at least 2 elements

Output : An integer
The largest difference of adjacent elements

Mandatory:

1. Create a class "Sample" with two following method:

2. Method Details:

a. Method Name = adjacentElementsDifference ()
b. Access Specifier = public
c. Argument = Two arguments [Array elements and array length] of type integer ,

In first test- case the array elements are 73 32 11 92 19 71 22 41 and array length is 8
d. Return type = integer , It returns the largest sum of adjacent elements to the main method

3. Access the adajcentElementsDifference(int[] arr,int n) in "Sample" class from main method class (TestClass)

Note:
The method definition should have variables as "arr[]" and "n" public int adajcentElementsDifferenceint arr[],int n)
Return the largest sum after calculating adjacentElementsDifference in the method to the main method

Source Code

```
import java.util.*;
import java.lang.*;
class Sample
{
  public int adjacentElementsDifference(int arr[],int n)
  {
    if(arr.length<2)
    {
      return 0;
    }
    int min=arr[1]-arr[0];
    for(int i=0;i<arr.length-1;i++)
    {
      if(min>arr[i+1]-arr[i])
      {
        min=arr[i+1]-arr[i];
      }
    }
    return Math.abs(min);
  }
}
public class TestClass {
  public static void main(String[] args) {
  Scanner sc=new Scanner(System.in);
       int n=sc.nextInt();
       int[] arr=new int[n];
       for(int i=0;i<arr.length;i++)
       {
         arr[i]=sc.nextInt();
       }
       Sample s=new Sample();
       int minResult;
       minResult=s.adjacentElementsDifference(arr,n);
       System.out.println(minResult);
 }
}
```

Sample Input

```
8
73 32 11 92 19 71 22 41
```

Sample Output

```
73
```

Result

Thus, Program " **Difference of adjacent elements** " has been successfully executed

Course: JAVA **Session:** Methods **Timestamp:** 2021-8-31 10:31:25 **Register Number:** RA2031241010094

Q. Alternating subarray prefix

There's an array A consisting of N non-zero integers A1..N. A subarray of A is called alternating if any two adjacent elements in it have different signs (i.e. one of them should be negative and the other should be positive).

For each x from 1 to N, compute the length of the longest alternating subarray that starts at x - that is, a subarray Ax..y for the maximum possible y x. The length of such a subarray is y-x+1.

Input
The first line of the input contains an integer T - the number of test cases.
The first line of each test case contains N.
The following line contains N space-separated integers A1..N

Output
For each test case, output one line with N space-separated integers - the lengths of the longest alternating subarray starting at x, for each x from 1 to N.

Mandatory:

1. Create a new class and the class should have method as follows:

a. Method name = subarray()
b. Method Type = void
c. Method arguments = no arguments
d. Access Specifiers = public

2. The method "subarray()" needs to be accessed from the main method class(TestClass) to perform all operations.

3. The objectname to access subarray method is "objname"

Source Code

```
import java.util.*;
public class TestClass {
  public void subarray()
  {
    Scanner sc=new Scanner(System.in);
    int t=sc.nextInt();
    while(t-- >0)
    {
      int n=sc.nextInt();
      long[] arr=new long[n];
      for(int i=0;i<n;i++)
        arr[i]=sc.nextInt();
      long[] dp=new long[n];
      dp[n-1]=1;
      for(int i=n-2;i>=0;i--)
      {
        if(arr[i]*arr[i+1]<=0)
        {
          dp[i]=dp[i+1]+1;
        }
        else
        {
          dp[i]=1;
        }
      }
      for(int i=0;i<n;i++)
      System.out.print(dp[i]+" ");
      System.out.println();
    }
  }
  public static void main(String[] args) {
       TestClass objname=new TestClass();
       objname.subarray();

  }
}
```

Sample Input

```
3
4
1 2 3 4
4
1 -5 1 -5
6
-5 -1 -1 2 -2 -3
```

Sample Output

```
1 1 1 1
4 3 2 1
1 1 3 2 1 1
```

Result

Thus, Program " **Alternating subarray prefix** " has been successfully executed

Course: JAVA **Session:** Methods **Timestamp:** 2021-8-31 10:31:52 **Register Number:** RA2031241010094

Q. Summing pairs

Write a java program to find all pairs of elements in the given array whose sum is equal to a given number.

For example, if {4, 5, 7, 11, 9, 13, 8, 12} is an array and 20 is the given number, then you have to find all pairs of elements in this array whose sum must be 20.

In this example (7, 13), (11,9) and (8, 12) are such pairs whose sum is 20

Testcases:

1. The First Line denotes the number of inputs
2. The Second Lines denotes the input elements
3. The third line is the number and find all pair elements whose sum is equal to the given number

Mandatory:

1. Create a new class called "Sample" and a method as follows:

a. Method name = findThe Pairs()
b. Type = void
c. Access Specifiers = static

2. Access the findThePairs() from the main method as classname.methodname() (The class name is Sample)

Source Code

```
import java.util.*;
class Sample
{
  static void findThePairs()
  {
    Scanner sc=new Scanner(System.in);
    int n=sc.nextInt();
    int[] arr=new int[n];
    for(int i=0;i<n;i++)
    {
      arr[i]=sc.nextInt();
    }
    int k=sc.nextInt();
    for(int p=0;p<n;p++)
    {
      for(int q=p+1;q<n;q++)
      {
        if((arr[p]+arr[q])==k)
        {
          System.out.println(arr[p]+"+"+arr[q]+"="+k);
        }
      }
    }
  }
}
public class TestClass {
  public static void main(String[] args) {
      Sample.findThePairs();

  }
}
```

Sample Input

```
6
1 8 25 24 17 32
49
```

Sample Output

```
25+24=49
17+32=49
```

Result

Thus, Program " **Summing pairs** " has been successfully executed

Q. Product with same static method name and different arguments

Write a methods that returns the product of two numbers and product of three numbers.

The method name should be same

Example

For param1 = 1, param2 = 2 and param3 = 3, the output should be
product(param1, param2) = 2.
product(param1, param2, param3) = 6

Mandatory:

1. Create two methods with same name and same type in the main class (TestClass) as follows:

Method name = productTwo
Arguments = Two arguments of type integer
Return type - integer
Access specifier - static

Method name = productTwo
Arguments = Three arguments of type integer
Return type - integer
Access specifier - static

2. Get three inputs from the user in the main method with variable names as "num1" , "num2", "num3" of type integer

3. Call or access the methods "productTwo(num1,num2)" and "productThree(num1,num2,num3)" from the main method using classname.methodname

4. The final returned value from the methods should be printed in the main method itself.

Hint for 4th Mandatory Test Case:

a. System.out.println(TestClass.productTwo(num1,num2))
b. System.out.println(TestClass.productTwo(num1,num2,num3))

Source Code

```
import java.util.*;
public class TestClass {
  static int productTwo(int num1,int num2)
  {
    return num1*num2;
  }
  static int productTwo(int num1,int num2,int num3)
  {
    return num1*num2*num3;
  }
  public static void main(String[] args)
     {
       Scanner sc=new Scanner(System.in);
       int num1=sc.nextInt();
       int num2=sc.nextInt();
       int num3=sc.nextInt();
       System.out.println(TestClass.productTwo(num1,num2));
       System.out.println(TestClass.productTwo(num1,num2,num3));

  }
}
```

Sample Input

4 5 12

Sample Output

20
240

Result

Thus, Program " **Product with same static method name and different arguments** " has been successfully executed