Mettl program for Milestone2

STRING BASED PROGRAMS

1.First String code

input1=input1.toLowerCase();

        String s[]=input1.split(" ");

        int sum=0;

        String res="";

        for(int i=0;i<s.length;i++)

        {

            int len=s[i].length()-1;

            sum=0;

            for(int j=0;j<=s[i].length()/2;j++)

            {

                if(j==len)

                {

                    int m=s[i].charAt(j)-'a'+1;

                    sum+=m;

                }

                if(j<len)

                {

                    int a=Math.abs(s[i].charAt(j)-s[i].charAt(len));

                    sum+=a;

                }

                if(j>len)

                break;

                len--;

            }

            res+=Integer.toString(sum);

        }

     return Integer.parseInt(res);

input1=WORLD WIDE WEB // world wide web

i.e. a=A=1,b=B=2,c=C=3………………z=Z=26

WORLD=[W-D]+[O-L]+[R] = [23-4]+[15-12]+[18] = 19+3+18 = 40

WIDE=[W-E]+[I-D] = [23-5]+[9-4] = 18+5 = 23

WEB=[W-B]+[E] = [23-2]+[5] = 21+5 = 26

[40][23][26]

o/p:[402326]

2.Get code through Strings

String str[]=input1.split(" ");

        int sum=0;

        for(int i=0;i<str.length;i++)

        {

            sum+=str[i].length();

        }

        int s=0;

        while(sum!=0 || s>9)

        {

            s+=sum%10;

            sum/=10;

            if(sum==0 && s>9)

            {

                sum=s;

                s=0;

            }

        }

        return s;

input1="Welcome the World";

o/p: 6

3.Addition using Strings

int maxlen;

        if(input1.length()>=input2.length())

        maxlen=input1.length();

        else

        maxlen=input2.length();

        input1=new StringBuffer(input1).reverse().toString();

        input2=new StringBuffer(input2).reverse().toString();

        int a[]=new int[maxlen];

        int b[]=new int[maxlen];

        for(int i=0;i<input1.length();i++)

        a[i]=Character.getNumericValue(input1.charAt(i));

        for(int i=0;i<input2.length();i++)

        b[i]=Character.getNumericValue(input2.charAt(i));

        int sum[]=new int[maxlen+1];

        int carry=0;

        for(int i=0;i<maxlen;i++)

        {

            sum[i]=(a[i]+b[i]+carry)%10;

            if((a[i]+b[i]+carry)>=10)

            carry=1;

            else

            carry=0;

        }

        sum[maxlen]=carry;

        String res="";

        for(int i=maxlen;i>=0;i--)

        {

            if((i==maxlen && sum[i]==1)||i!=maxlen)

            res+=Integer.toString(sum[i]);

        }

        return res;

Input : input1 = "3333311111111111",

input2 = "44422222221111"

Output : 3377733333332222

Input : input1 = "7777555511111111",

Input2 = "3332222221111"

Output : 7780887733332222

ARRAY BASED PROGRAMS

Simple Encoded Array

Simple Encoded Array: Maya has stored few confidential numbers in an array (array of int). To ensure that others do not find the numbers easily, she has applied a simple encoding.  
Encoding used: Each array element has been substituted with a value that is the sum of its original value and its succeeding element’s value.  
i.e. arr[i] = arr[i] + arr[i+1]  
e.g. value in arr[0] = original value of arr[0] + original value of arr[1]  
Also note that value of last element i.e. arr[last index] remains unchanged.

NOTE: Only the “Encoded array” will be supplied to the function and it is expected to do the processing to find the expected result values.

int orrar[]=new int[input2];

        for(int i=input2-1;i>=0;i--)

        {

            if(i==input2-1)

            orrar[i]=input1[i];

            else

            {

                orrar[i]=input1[i]-orrar[i+1];

            }

        }

        int sum=0;

        for(int i=0;i<input2;i++)

        sum+=orrar[i];

        return new Result(orrar[0],sum);

Example:  
If the original array is –  
{2, 5, 1, 7, 9, 3}  
The encoded array would be –  
{7, 6, 8, 16, 12, 3}

Provided the encoded array, you are expected to find the –  
a) First number (value in index 0) in the original array  
b) Sum of all numbers in the original array

The prototype of the function is:  
public static void findOriginalFirstAndSum(int[] input1);  
where input1 is the encoded array.  
The method is expected to –

* find the value of the first number of the original array and store it in the member output1 and
* find the sum of all numbers in the original array and store it in the member output2

Assumption(s):

* The array elements can be positive and/or negative numbers

Example 1:  
Original array = {2, 5, 1, 7, 9, 3}  
Encoded array = {7, 6, 8, 16, 12, 3}  
First number in original array = 2  
Sum of all numbers in original array = 27

2.Decreasing Sequence

int seq=0,max=0,count=1;

        for(int i=0;i<input2-1;i++)

        {

            if(input1[i]>input1[i+1])

                count++;

            else

            {

                if(count>1)

                    seq++;

                if(count>max && seq>0)

                    max=count;

                count=1;

            }

        }

        if(count>max && input2>1 &&count!=1 )

        max=count;

        if(count>1)

        seq++;

            return new Result(seq,max);

input1[]={11,3,1,4,7,8,12,2,3,7};

input2=10

Output1=2

Output2=3

Explanation array input1 ,the decreasing sequences are “11,3,1” and “12,2” i.e there are two decreasing sequence in the array and so output1 is assigned 2.the first sequence i.e “11,3,1” is the longer one containing three items.when compared to the second sequence “12,2” which contains 2 items.So, the lengtnof the longest decreasingsequence output=3

Ex:2

input1[]={9};

input2=1

Output1=0

Output2=0

Ex:3

input1[]={12,51,100,212,15,12,7,3,57,300,312,78,19,100,102,101,99,74,0,-5};

input2=20

Output1=3

Output2=6

3.Most frequently Occuring the digits

   int a[]=new int[10000];

        int m=0;

        for(int i=0;i<input2;i++)

        {

            while(input1[i]!=0)

            {

                a[m]=input1[i]%10;

                input1[i]/=10;

                m++;

            }

        }

        int coun[]=new int[10];

        for(int i=0;i<10;i++)

        {

            int count=0;

            for(int j=0;j<m;j++)

            {

                if(i==a[j])

                count++;

            }

            coun[i]=count;

        }

        int max=0,ele=0;

        for(int i=0;i<10;i++)

        {

            if(max<coun[i])

            {

                max=coun[i];

                ele=i;

            }

            if(max==coun[i])

            {

                if(ele<i)

                ele=i;

            }

        }

        return ele;

Example 1:

input1[]={1237,262,666,140}

input2:number of array elements (i.e) 4

0 occurs 1 times

1 occurs 2 times

2 occurs 3 times

3 occurs 1 times

4 occurs 1 times

5 occurs 0 times

6 occurs 4 times

7 occurs 1 times

8 occurs 0 times

9 occurs 0 times

4 is the highest frequency in this series and 6 is the digit that occurs 4 times

o/p: 6

Exampl 2:

input1[]={1237,202,666,140}

input2:number of array elements (i.e) 4

0 occurs 2 times

1 occurs 2 times

2 occurs 3 times

3 occurs 1 times

4 occurs 1 times

5 occurs 0 times

6 occurs 3 times

7 occurs 1 times

8 occurs 0 times

9 occurs 0 times

3 is the highest frequency in this series and

2 and 6 is the digit that occurs 3 times

So largest number of two digits is 6

o/p: 6

NUMBER BASED PROBLEMS

1.Sum of power of Digits

int a[]=new int[1000];

        int i=0;

        while(input1!=0)

        {

            a[i]=input1%10;

            input1/=10;

            i++;

        }

        int b[]=new int[1000];

        int m=0;

        for(int j=i-1;j>=0;j--)

        {

            b[m]=a[j];

            m++;

        }

        int sum=0;

        for(int j=0;j<m-1;j++)

        {

            int n=(int)(Math.pow(b[j],b[j+1]));

            sum+=n;

        }

        sum+=1;

        return sum;

input1=582109

(5 to the power of 8)+ (8 to the power of 2)+ (2 to the power of 1)+ (1 to the power of 0)+ (0 to the power of 9)+ (9 to the power of 0)=390625+64+2+1+0+1=390693

o/p: 390693

2.Sum of sum of the Digits in Cyclic Order

int a[]=new int[10000];

        int k=0;

        while(input1!=0)

        {

            a[k]=input1%10;

            input1/=10;

            k++;

        }

        int sum=0;

        for(int i=k-1;i>=0;i--)

        {

            for(int j=0;j<=i;j++)

            sum+=a[j];

        }

        return sum;

input1=582109

(5+8+2+1+0+9)+(8+2+1+0+9)+(2+1+0+9)+(1+0+9)+(0+9)+(9)=85

o/p: 85

STRING BASED PROGRAMS (PART2)

1.Identify possible words

input1=input1.toLowerCase();

        input2=input2.toLowerCase();

        String str[]=input2.split(":");

        int count=0;

        String res="";

        for(int i=0;i<str.length;i++)

        {

            count=0;

            if(str[i].length()==input1.length())

            {

                for(int j=0;j<input1.length();j++)

                {

                    if(input1.charAt(j)!='\_')

                    {

                        if(input1.charAt(j)!=str[i].charAt(j))

                        {

                            count++;

                            break;

                        }

                    }

                }

                if(count==0)

                {

                    String s=str[i].toUpperCase();

                    res+=s+":";

                }

            }

        }

        if(res.length()==0)

        return "ERROR-009";

        else

        return res.substring(0,res.length()-1);

input1="Fi\_er";

input2="Fever:filter:filer:Fixer:fiber:fibre:tailor:offer";

O/p:

FILER:FIXER:FIBER

2.Encoding three String

   String first[]=new String[3];

        String second[]=new String[3];

        String third[]=new String[3];

        first=method(input1);

        second=method(input2);

        third=method(input3);

        String res1=first[0]+second[0]+third[0];

        String res2=first[1]+second[1]+third[1];

        String res3=first[2]+second[2]+third[2];

        String res33="";

        for(int i=0;i<res3.length();i++)

        {

            if(Character.isUpperCase(res3.charAt(i)))

            res33+=Character.toLowerCase(res3.charAt(i));

            if(Character.isLowerCase(res3.charAt(i)))

            res33+=Character.toUpperCase(res3.charAt(i));

        }

        return new Result(res1,res2,res33);

}

    public static String[] method(String str)

    {

        int split=str.length()/3;

        String part[]=new String[3];

        if(str.length()%3==0)

        {

            int k=0;

            for(int i=0;i<str.length();i=i+split)

            {

                part[k]=str.substring(i,i+split);

                k++;

            }

        }

        else if(str.length()%3==1)

        {

            int count=0,k=0;

            for(int i=0;i<str.length();i=i+split)

            {

                count++;

                if(count==2)

                {

                    part[k]=str.substring(i,(i+split)+1);

                    k++;

                    i++;

                }

                else

                {

                    part[k]=str.substring(i,i+split);

                    k++;

                }

            }

        }

        else if(str.length()%3==2)

        {

            int count=0,k=0;

            for(int i=0;i<str.length();i=i+split)

            {

                count++;

                if(count==2)

                {

                    part[k]=str.substring(i,i+split);

                    k++;

                }

                else

                {

                    part[k]=str.substring(i,(i+split)+1);

                    k++;

                    i++;

                }

            }

        }

        return part;

    }

Input1=”john”

Input2=”johny”

Input3=”janardhan”

**Step 1:**

“John” should be split into “J” ,”oh”,”n” as FRONT,MIDDLE and END parts respectively.

“Johny” should be split into “Jo” ,”h”,”ny” as FRONT,MIDDLE and END parts respectively.

“Janardhan” should be split into “Jan” ,”ard”,”han” as FRONT,MIDDLE and END parts respectively.

i.e. if the no of character in the string are in multuiples os 3 ,then each split-part will contain equal no of character ,as seen in the example of “janardhan”

if the no of character in the string are NOT in multuiples of 3,and if there is one character more than multiple of 3, then the MIDDLE part will get the extra one character,as seen in the example of “john”

if the no of character in the string are NOT in multuiples of 3,and if there is two character more than multiple of 3, then the FRONT and END part will get the extra one character,as seen in the example of “johny”

**Step 2:**

Concatenate(join) the FRONT MIDDLE and END parts of the stringsper the below specified concatenation-rule to form three output Strings.

Output1=FRONT part of input1+ FRONT part of input2+ FRONT part of input3

Output2=MIDDLE part of input1+ MIDDLE part of input2+ MIDDLE part of input3

Output3=END part of input1+ END part of input2+ END part of input3

For example:

Output1=”J”+”Jo”+”Jan” = “JJoJan”

Output2=”oh”+”h”+”ard” = “ohhard”

Output3=”n”+”ny”+”han” = “nnyhan”

**Step 3:**

Toggle of each character in the string(output3)

For example:After applying the toggle rule,output3 should be come “NNYHAN”

**Final Output:**

Output1=”JJoJan”

Output2=”ohhard”

Output3=”NNYHAN”