COMPUTER SCIENCE
Paper – 2
(PRACTICAL)
(Reading Time: 15 minutes)
(Planning Session AND Examination Session: Three Hours)

The total time to be spent on the Planning and the Examination Session is Three hours.
After completing the Planning Session, the candidate may begin with the Examination Session.
A maximum of 90 minutes is permitted to begin the Examination Session.
(Maximum Marks: 80)

As it is a practical examination the candidate is expected to do the following:

1. Write an algorithm for the selected problem. Algorithm should be expressed clearly using ay standard scheme such as pseudo code or in steps which are simple enough to be obviously computable. **[10]**
2. Write a program in JAVA language. The program should follow the algorithm and should be logically and syntactically correct. **[20]**
3. Document the program using mnemonic names / comments, identifying and clearly describing the choice of data types and meaning of variables. **[20]**
4. Code / Type the program on the computer and get a printout (hard copy). Typically this should be a program that compiles and runs correctly. **[10]**
5. Test run the program on the computer using the given sample data and get a printout of the output in the format specified in the problem. **[20]**
6. Viva-Voce on the **Selected Problem.** **[20]**

Solve any **one** of the following Problems:

Question 1.

An ISBN (International Standard Book Number) is a ten digit code which uniquely identifies a book.

The first nine digits represent the group, publisher and title of the book and the last digit is used to check whether ISBN is correct or not.

Each of the first nine digits of the code can take a value between 0 to 9. Sometimes it is necessary to make the last digit equal to ten. This is done by writing the last digit of the code as X.

To verify an ISBN, calculate 10 times the first digit, plus 9 times the second digit, plus 8 times the third digit and so on until we add 1 time the last digit. If the final number leaves no remainder while divided by 11, the code is a valid ISBN

For example:
0201103311=10*0+9*2+8*0+7*1+6*1+5*0+4*3+3*3+2*1+1*1=55
This is a valid ISBN
007462542X=10*0+9*0+8*7+7*4+6*6+5*2+4*5+3*4+2*2+1*10=176
This is a valid ISBN
Similarly 0112112425 is not a valid ISBN.

Test Data:
Input code: 0201530821
Output: Sum=99
Leaves no remainder – valid ISBN

Input code: 356680324
Output: Sum=invalid input
Input code: 0231428031
Output: Sum=122
Leaves remainder – invalid ISBN

Question 2:

Write a program to declare a **square matrix A[][]** of order (MXN) where 'M' is the number of rows and the number of columns. 'M' should be greater than 2 and less than 20. Allow user to enter integers into this matrix. Display appropriate error message for an invalid input. Perform the following tasks.

1. Display the input matrix
2. Create a **mirror image of the inputted matrix**.
3. Display the mirror image matrix

Test Data:
Input: M=3
4 16 12

8 2 14

6 1 3

Output:
Original matrix
4 16 12

8 2 14

6 1 3

Mirror image matrix:
12 16 4

14 2 8

3 1 6

Input: M=22
Output: Size out of range
Question 3:
A palindrome is a word that may be read the same in either direction.

Accept a sentence in upper case which is terminated by either '.' , ',' , '? ' , '!'. Each word in the sentence is separated by a blank space.

Perform the following tasks:

Test Data:
Input: MOM AND DAD ARE COMING AT NOON.
Output: MOM DAD NOON
Number of palindromic words: 3

Input: HOW ARE YOU?
Output: No palindromic words

For solutions you can see the comments below as students have posted the codes there.

ISC Practical Paper – 2012 (Solved)

January 22, 2013 18 comments

COMPUTER SCIENCE
Paper – 2
(PRACTICAL)

(Reading Time: 15 minutes)
(Planning Session AND Examination Session: Three Hours)

The total time to be spent on the Planning and the Examination Session is Three hours.
After completing the Planning Session, the candidate may begin with the Examination Session.
A maximum of 90 minutes is permitted to begin the Examination Session.
(Maximum Marks: 80)

As it is a practical examination the candidate is expected to do the following:

1. Write an algorithm for the selected problem. **[10]**
2. Algorithm should be expressed clearly using ay standard scheme such as pseudo code or in steps which are simple enough to be obviously computable.
3. Write a program in JAVA language. The program should follow the algorithm and should be logically and syntactically correct. **[20]**
4. Document the program using mnemonic names / comments, identifying and clearly describing the choice of data types and meaning of variables. **[10]**
5. Code / Type the program on the computer and get a printout (hard copy). Typically this should be a program that compiles and runs correctly. **[10]**
6. Test run the program on the computer using the given sample data and get a printout of the output in the format specified in the problem. **[20]**
7. Viva-Voce on the **Selected Problem.** **[20]**

Solve any **one** of the following Problems:

Question 1.

A prime palindrome integer is a positive integer (without leading zeros) which is prime as well as a palindrome. Given two positive integers m and n, where m<= n, **write a program to determine how many prime-palindrome** integers are there in the range between m and n (both inclusive) and output them.

The input contains two positive integers m and n where m>=100 and n<= 3000. Display number of prime palindrome integers in the specified range along with their values in the format specified below:

Test your program with the sample data and some random data:

Example 1:

INPUT:

M=100

N=1000

OUTPUT: The prime palindrome integers are:
101,131,151,181,191,313,351,373,383,727,757,787,797,919,929

Frequency of prime palindrome integers: 15

Example 2:

INPUT:

M=100

N=5000

OUTPUT: Out of Range

```
import java.io.*;
class PrimePal
{
   public void showPrimePal() throws IOException
```

```
{
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    int m,n, c=0;
    System.out.println("Enter the Lower Limit:");
    m=Integer.parseInt(br.readLine());
    System.out.println("Enter the Upper Limit:");
    n=Integer.parseInt(br.readLine());
    if(m>n || n>3000)
        System.out.println("Out of Range.");
    else
    {
        System.out.println("The Prime Palindrome integers are:");
        while(m<=n)
        {
            if(checkPrimePal(m))
            {
                if(c==0)
                    System.out.print(m);
                else
                    System.out.print(", "+m);
                c++;
            }
            m++;
        }
        System.out.println("\nFrequency of Prime Palindrome integers: "+c);
    }
}
boolean checkPrimePal(int x)
{
    boolean flag=false;
    int c=0, temp=x;
    int rev=0,rem=0;
    for(int i=1;i<=x;i++)
    {
        if(x%i==0)
            c++;
    }
    if(c<=2)
    {
        while(x!=0)
        {
            rem=x%10;
            rev=rev*10+rem;
            x/=10;
        }
        if(rev==temp)
            flag=true;
    }
    return flag;
}
public static void main(String args[]) throws IOException
{
    PrimePal ob=new PrimePal ();
    ob.showPrimePal();
```

```
    }
}
```

Question 2.

Write a program to accept a sentence as input. The words in the string are to be separated by a blank. Each word must be in upper case. The sentence is terminated by either ".","!" or "?". Perform the following tasks:

(i) Obtain the **length of the sentence.** (measured in words)

(ii) Arrange the sentence in alphabetical order of the words.

Test your program with the sample data and some random data:

Example 1:

INPUT: NECESSITY IS THE MOTHER OF INVENTION.

OUTPUT:

Length: 6

Rearranged Sentence:

INVENTION IS MOTHER NECESSITY OF THE

Example 2:

INPUT: BE GOOD TO OTHERS.

OUTPUT:

Length: 4

Rearranged Sentence: BE GOOD OTHERS TO

```
import java.io.*;
class WordsInSentence
{
    public void take() throws IOException
    {
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        String str, words[], stk="";
        int i,j,c=0,flag, len;
        char ch;
        while(true)
        {
            flag=0;
            System.out.println("Enter the sentence:");
            str=br.readLine();
            len= str.length();
            words=new String[len];
            for(i=0;i
            {
                if(Character.isLowerCase(str.charAt(i)))
                {
                    flag=1;
                    break;
                }
            }
            if (flag==0)
                break;
            else
                System.out.println("Enter the sentence again with all uppercase letters");
        }
        i=0;
        while(i<len)
```

```
        {
            ch=str.charAt(i);
            if(ch==' '|| ch=='.' || ch=='?' || ch=='!')
            {
                words[c]=stk;
                c++;
                i++;
                stk="";
            }
            else
            {
                stk+=ch;
                i++;
            }
        }
        for(i=0;i<c;i++)
        {
            for(j=0;j<c-1;j++)
            {
                if((words[j].compareTo(words[j+1]))>0)
                {
                    stk=words[j];
                    words[j]=words[j+1];
                    words[j+1]=stk;
                }
            }
        }
        System.out.println("Length= "+c);
        System.out.println("\nRearranged Sentence:\n");
        for(i=0;i<c;i++)
            System.out.print(words[i]+" ");
    }
    public static void main(String args[]) throws IOException
    {
        WordsInSentence ob=new WordsInSentence();
        ob.take();
    }
}
```

Question 3.

Write a program to declare a matrix A [][] of order (MXN) where 'M' is the number of rows and 'N' is the number of columns such that both M and N must be greater than 2 and less than 20. Allow the user to input integers into this matrix. Perform the following tasks on the matrix:

1.
2. Display the input matrix
3. Find the maximum and minimum value in the matrix and display them along with their position.
4. Sort the elements of the matrix in ascending order using any standard sorting technique and rearrange them in the matrix.
5. Output the rearranged matrix.

Test your program with the sample data and some random data:

Example 1.

INPUT:

M=3

N=4

Entered values: 8,7,9,3,-2,0,4,5,1,3,6,-4

OUTPUT:

Original matrix:

8 7 9 3

-2 0 4 5

1 3 6 -4

Largest Number: 9

Row: 0

Column: 2

Smallest Number: -4

Row=2

Column=3

Rearranged matrix:

-4 -2 0 1

3 3 4 5

6 7 8 9

```
import java.io.*;
class Matrix
{
public void take()throws Exception
{
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
int arr[][],m,n;
int g,h,i,j,max,min,maxr,maxc,minr,minc;
while(true)
{
System.out.println("\nEnter the number of rows :");
m=Integer.parseInt(br.readLine());
System.out.println("\nEnter the number of columns:");
n=Integer.parseInt(br.readLine());
if(m<2 || n<2 || m>20 || n>20)
System.out.println("\nEnter the number of rows and columns");
else
break;
}
arr=new int[m][n];
for(i=0;i<m; i++)
{
for(j=0;j<n;j++)
{
System.out.println("\nEnter Value:");
arr[i][j]=Integer.parseInt(br.readLine());
}
```

```
}
max=arr[0][0];
min=arr[0][0];
maxr=0;
minr=0;
maxc=0;
minc=0;
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
if(arr[i][j]>max)
{
max=arr[i][j];
maxr=i;
maxc=j;
}
else if(arr[i][j]< min)
{
minr=i;
minc=j;
min=arr[i][j];
}
}
}
System.out.println("\nOriginal Matrix\n");
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
System.out.print(arr[i][j]+" ");
}
System.out.println();
}
System.out.println("\nMaximum Value="+max);
System.out.println("\nRow="+maxr);
System.out.println("\nColumn="+maxc);
System.out.println("\nMinimum Value="+min);
System.out.println("\nRow="+minr);
System.out.println("\nColumn="+minc);
for(g=0;g<m;g++)
{
for(h=0;h<n;h++)
{
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
if(arr[g][h]< arr[i][j])
{
min=arr[g][h];
arr[g][h]=arr[i][j];
arr[i][j]=min;
}
```

```
}
}
}
}
System.out.println("\nSorted Array\n");
for(i=0;i<m;i++)
{
for(j=0;j<n;j++)
{
System.out.print(arr[i][j]+" ");
}
System.out.println();
}
}
public static void main(String args[]) throws Exception
{
Matrix ob=new Matrix();
ob.take();
}
}
```

ISC Computer Practical – 2010 (Solved)

December 11, 2011 3 comments

COMPUTER SCIENCE
Paper – 2
(PRACTICAL)
(Reading Time: 15 minutes)
(Planning Session AND Examination Session: Three Hours)

The total time to be spent on the Planning and the Examination Session is Three hours.
After completing the Planning Session, the candidate may begin with the Examination Session.
A maximum of 90 minutes is permitted to begin the Examination Session.
(Maximum Marks: 80)

As it is a practical examination the candidate is expected to do the following:

1. Write an algorithm for the selected problem. **[10]**
2. Algorithm should be expressed clearly using ay standard scheme such as pseudo code or in steps which are simple enough to be obviously computable.
3. Write a program in JAVA language. The program should follow the algorithm and should be logically and syntactically correct. **[20]**
4. Document the program using mnemonic names / comments, identifying and clearly describing the choice of data types and meaning of variables. **[10]**
5. Code / Type the program on the computer and get a printout (hard copy). Typically this should be a program that compiles and runs correctly. **[10]**
6. Test run the program on the computer using the given sample data and get a printout of the output in the format specified in the problem. **[20]**
7. Viva-Voce on the **Selected Problem.** **[20]**

Solve any **one** of the following Problems:

Question 1

A bank intends to design a program to display the denomination of an input amount, up to 5 digits. The available denomination with the bank are of rupees 1000 , 500 , 100 , 50 , 20 , 10 , 5 , 2 , and 1. Design a program to accept the amount from the user and display the break-up in descending order of denomination. (i.e. preference should be given to the highest denomination available) along with the total number of notes. [Note: Only the denomination used should be displayed]. Also print the amount in words according to the digits.

Example 1
INPUT: 14856
OUTPUT:
ONE FOUR EIGHT FIVE SIX

DENOMINATION :

1000 x 14 = 14000

500 x 1 = 500

100 x 3 = 300

50 x 1 = 50

5 x 1 = 5

1 x 1 = 1

TOTAL = 14856

TOTAL NUMBER OF NOTES = 21

Example 2
INPUT : 6043
OUTPUT :
SIX ZERO FOUR THREE

DENOMINATION:

1000 x 6 = 6000

20 x 2 = 40

2 x 1 = 2

1 X 1 = 1

TOTAL = 6043

TOTAL NUMBER OF NOTES = 10

Example 3
INPUT : 235001
OUTPUT:
INVALID AMOUNT

```
import java.io.*;
class QBank
{
   public void intake() throws IOException
   {
      int rev=0,amount,dummy,rem;
      BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
      System.out.println("Enter the Amount:");
```

```
amount=Integer.parseInt(br.readLine());
if(amount >99999)
{
   System.out.println("Invalid Amount…");
   return;
}
dummy=amount;
while(dummy !=0)
{
   rev=rev*10+dummy%10;
   dummy=dummy/10;
}
System.out.print("Amount in words :");
while(rev!=0)
{
   rem=rev%10;
   switch(rem)
   {
      case 0:
      System.out.print(" ZERO");
      break;
      case 1:
      System.out.print(" ONE");
      break;
      case 2:
      System.out.print(" TWO");
      break;
      case 3:
      System.out.print(" THREE");
      break;
      case 4:
      System.out.print(" FOUR");
      break;
      case 5:
      System.out.print(" FIVE");
      break;
      case 6:
      System.out.print(" SIX");
      break;
      case 7:
      System.out.print(" SEVEN");
      break;
      case 8:
      System.out.print(" EIGHT");
      break;
      case 9:
      System.out.print(" NINE");
   }
   rev=rev/10;
}
int den[]={1000,500,100,50,20,10,5,2,1};
int i=0, tot=0;
System.out.println("\nDENOMINATORS:\n");
while (amount!=0)
```

```
        {
            rev=amount/den[i];
            if(rev!=0)
            {
                System.out.println(den[i]+" X " + rev + " = " + rev*den[i]);
                tot+=rev;
            }
            amount=amount%den[i];
            i++;
        }
        System.out.println("TOTAL NUMBER OF NOTES:  "+ tot);
    }
}
```

Question 2

A positive whole number 'n' that has 'd' number of digits is squared and split into two pieces, a right-hand piece that has 'd' digits and a left-hand piece that has remaining 'd' or 'd-1' digits. If the sum of the two pieces is equal to the number, then 'n' is a Kaprekar number. The first few Kaprekar numbers are: 9, 45, 297 ……..

Example 1:
9

9^2 = 81, right-hand piece of 81 = 1 and left hand piece of 81 = 8
Sum = 1 + 8 = 9, i.e. equal to the number.

Example 2:
45

45^2 = 2025, right-hand piece of 2025 = 25 and left hand piece of 2025 = 20
Sum = 25 + 20 = 45, i.e. equal to the number.

Example 3:
297

297^2 = 88209, right-hand piece of 88209 = 209 and left hand piece of 88209 = 88
Sum = 209 + 88 = 297, i.e. equal to the number.

Given the two positive integers *p* and *q*, where *p* < *q*, write a program to determine how many Kaprekar numbers are there in the range between *p* and *q* (both inclusive) and output them.
The input contains two positive integers *p* and *q*. Assume *p* < 5000 and *q* < 5000. You are to output the number of Kaprekar numbers in the specified range along with their values in the format specified below:

SAMPLE DATA:
INPUT:
p = 1
q = 1000
OUTPUT:
THE KAPREKAR NUMBERS ARE:-

1, 9, 45, 55, 99, 297, 703, 999

FREQUENCY OF KAPREKAR NUMBERS IS: 8

```
import java.io.*;
class FindKarpekar
{
    int p,q;
    int count=0;
```

```
    void input()throws IOException
    {
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        p=Integer.parseInt(br.readLine());
        q=Integer.parseInt(br.readLine());
        if(p>=5000 || q>=5000 || p>=q)
            System.out.println("Out of range");
        else
        {
            System.out.println("THE KAPREKAR NUMBERS ARE:- ");
            for(int i=p;i<=q;i++)
            {
                if(check(i)==1)
                {
                   if(count==0)
                       System.out.print(i);
                  else
                        System.out.print("   ,"+i);
                     count++;
                }
            }
            System.out.println();
            System.out.println("FREQUENCY OF KAPREKAR NUMBERS IS:  "+count);
        }
    }
    long check(int n)
    {
        int len=(""+n).length();
        long sq=n*n;
        long k=sq%(long)(Math.pow(10,len));
        long s=sq/(long)(Math.pow(10,len));
        long l=(k+s);
        if(l==n)
            return 1;
        else
            return 0;
    }
    public static void main(String args[])throws IOException
    {
        FindKarpekar ob=new FindKarpekar();
        ob.input();
    }
}
```

Question 3

Input a paragraph containing 'n' number of sentences where (1 = < n < 4). The words are to be separated with a single blank space and are in UPPERCASE. A sentence may be terminated either with a full stop '.' Or a question mark '?' only. Any other character may be ignored. Perform the following operations:

(i) Accept the number of sentences. If the number of sentences exceeds the limit, an appropriate error message must be displayed.

(ii) Find the number of words in the whole paragraph

(iii) Display the words in ascending order of their frequency. Words with same frequency may appear in any order.

Example 1
INPUT: Enter number of sentences.
1

Enter sentences.

TO BE OR NOT TO BE.

OUTPUT:
Total number of words: 6

WORD	FREQUENCY
OR	1
NOT	1
TO	2
BE	2

Example 2
INPUT: Enter number of sentences
3

Enter sentences.

THIS IS A STRING PROGRAM.IS THIS EASY?YES,IT IS.
OUTPUT:
Total number of words: 11

WORD	FREQUENCY
A	1
STRING	1
PROGRAM	1
EASY	1
YES	1
IT	1
THIS	2
IS	3

Example 3
INPUT : Enter number of sentences
5

OUTPUT:
INVALID ENTRY

```
import java.io.*;
import java.util.*;
class SentencesOrder
{
    String s,str,sarr[],strarr[];
    StringTokenizer st;
    int i,j,n,c,index=0,fre[],index1=0;
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
```

```
public void take() throws IOException
{
    System.out.println("Enter the Number of sentences:");
    n=Integer.parseInt(br.readLine());
    if(n< 1 || n >4)
    { System.out.println("Wrong Input…");
        return;
    }
    System.out.println("Enter the Paragraph:");
    str=br.readLine();
    st=new StringTokenizer(str,",.? ");
    n=st.countTokens();
    System.out.println("Number of Words in the paragraph="+n);
    sarr=new String[n];
    strarr=new String[n];
    fre=new int[n];
    while(st.hasMoreTokens())
    {
        sarr[index++]=st.nextToken();
    }
    for(i=0;i< index-1;i++)
    {
        for(j=i+1;j< index;j++)
        {
            if(sarr[i].compareTo(sarr[j]) > 0)
            {
                s=sarr[i];
                sarr[i]=sarr[j];
                sarr[j]=s;
            }
        }
    }
    c=1;
    s=sarr[0];
    for(i=1;i< index;i++)
    {
        if(!s.equals(sarr[i]))
        {
            strarr[index1]=s;
            fre[index1++]=c;
            c=1;
            s=sarr[i];
        }
        else
            c++;
    }
    strarr[index1]=s;
    fre[index1++]=c;
    for(i=0;i< index1-1;i++)
    {
        for(j=i+1;j< index1;j++)
        {
            if(fre[i] > fre[j])
            {
```

```
                n=fre[i];
                fre[i]=fre[j];
                fre[j]=n;
                s= strarr[i];
                strarr[i]=strarr[j];
                strarr[j]=s;
            }
        }
    }
    System.out.println("WORD          FREQUENCY");
    for(i=0;i< index1;i++)
        System.out.println(strarr[i]+"    "+fre[i]);
  }
}
```

Share this:

- Facebook9
- Twitter
- LinkedIn
- More
-

ISC Computer Practical-2006

December 11, 2011Leave a comment

COMPUTER SCIENCE
Paper – 2
(PRACTICAL)
(Reading Time: 15 minutes)
(Planning Session: 60 – 90 minutes)
(Examination Session: 90 – 120 minutes)
(Maximum Marks: 80)
——————————————————
INSTRUCTIONS

As it is a practical examination the candidate is expected to do the following:

(a) Write an algorithm for the selected problem.

(b) Write a program in C++ or JAVA. Document the program by using mnemonic names and comments.

(c) Test run the program on the computer using the given test data and get a printout (hardcopy) in the format specified in the problem along with the program listing.

——————————————————————————————————————

Solve any **one** of the following Problems:

Question 1

A positive natural number, (for e.g. 27), can be represented as follows –

2+3+4+5+6+7

8+9+10

13+14

where every row represents a combination of consecutive natural numbers, which add up to 27.

Write a program which inputs a positive natural number N and prints the possible consecutive number combinations, which when added give N.

Test your program for the following data and some random data.

SAMPLE DATA
INPUT:
N = 9

OUTPUT:
4 + 5

2 + 3+ 4

INPUT:
N = 15
OUTPUT:
7 +8

1 +2+ 3+ 4+ 5

4 +5+ 6

INPUT:
N = 21

OUTPUT:
10+ 11

1+ 2+ 3+ 4+ 5+ 6

6+ 7+ 8

Question 2.
Write a program that inputs the names of people into two different arrays A and B. Array A has N number of names while Array B has M number of names, with no duplicates in either of them. Merge arrays A and B into a single array C, such that the resulting array is sorted alphabetically.
Display all the three arrays, A, B and C, sorted alphabetically.

Test your program for the given data and some random data.

SAMPLE DATA:
INPUT:
Enter number of names in Array A, N = 2
Enter number of names in Array B, M = 3

First array: (A)

Suman

Anil

Second array: (B)

Usha

Sachin

John

OUTPUT:

Sorted Merged array: (C)

Anil

John

Sachin

Suman

Usha

Sorted first array: (A)

Anil

Suman

Sorted Second array: (B)

John

Sachin

Usha

Question 3.
A new advanced Operating System, incorporating the latest hi-tech features has been designed by Opera Computer Systems.

The task of generating copy protection codes to prevent software piracy has been entrusted to the Security Department.

The Security Department has decided to have codes containing a jumbled combination of alternate uppercase letters of the alphabet starting from A upto K (namely among A,C,E,G,I,K). The code may or may not be in the consecutive series of alphabets.

Each code should not exceed 6 characters and there should be no repetition of characters. If it exceeds 6 characters, display an appropriate error message.

Write a program to input a code and its length. At the first instance of an error display invalid!" stating the appropriate reason. In case of no error, display the message "Valid!".

Test your program for the following data and some random data.

SAMPLE DATA
INPUT:
N = 4

ABCE

OUTPUT:
Invalid! Only alternate letters permitted!

INPUT:
N = 4

AcIK

OUTPUT:
Invalid! Only upper case letters permitted!

INPUT:
N = 4

AAKE

OUTPUT:
Invalid! Repetition of characters not permitted!

INPUT:
N = 7

OUTPUT:
Error! Length of string should not exceed 6 characters!

INPUT:
N = 4

AEGIK

OUTPUT:
Invalid! String length not the same as specified!

INPUT:
N = 3

ACE

OUTPUT:
Valid!

INPUT:
N = 5

GEAIK

OUTPUT:
Valid!

Share this:

- Facebook
- Twitter
- LinkedIn
- More
-

ISC Computer Practical-2005

December 11, 2011Leave a comment

COMPUTER SCIENCE
Paper – 2
(PRACTICAL)
(Reading Time: 15 minutes)
(Planning Session: 60 – 90 minutes)
(Examination Session: 90 – 120 minutes)
(Maximum Marks: 80)
————————————————————
INSTRUCTIONS

As it is a practical examination the candidate is expected to do the following:

(a) Write an algorithm for the selected problem.

(b) Write a program in C++ / JAVA. Document the program by using mnemonic names and comments.

(c) Test run the program on the computer using the given test data and get a printout (hardcopy) in the format specified in the problem along with the program listing.

——

Solve any **one** of the following Problems:

Question 1

Write a program which takes a string (maximum 80 characters) terminated by a full stop. The words in this string are assumed to be separated by one or more blanks.
Arrange the words of the input string in descending order of their lengths. Same length words should be sorted alphabetically. Each word must star with an uppercase letter and the sentence should be terminated by a full stop.

Test your program for the following data and some random data.

SAMPLE DATA:
INPUT:
"This is human resource department."

OUTPUT:
Department Resource Human This Is.

INPUT:
"To handle yourself use your head and to handle others use your heart."

OUTPUT:
Yourself Handle handle Others Heart Head You're your And Use Use To To.

Question 2.

A wondrous square is an n by n grid which fulfills the following conditions:

(i) It contains integers from 1 to n^2, where each integer appears only once.
(ii) The sum of integers in any row or column must add up to $0.5 \times n \times (n^2 + 1)$.
For example the following grid is a wondrous square where the sum of each row or column is 65 when n = 5:

17	24	1	8	15
23	5	7	14	16
4	6	13	20	22
10	12	19	21	3
11	18	25	2	9

Write a program to read n ($2 \leq n \leq 10$) and the values stored in these n by n cells and output if the grid represents a wondrous square or not.
Also output all the prime numbers in the grid along with their row index and column index as shown in the output. A natural number is said to be prime if it has exactly two divisors. E.g. 2,3,5,7,11 ……..

The first element of the given grid i.e 17 is stored at row index 0 and column index 0 and the next element in the row i.e. 24 is stored at row index 0 and column index 1.

Test your program for the following data and some random data.

SAMPLE DATA:
INPUT:
N = 4

16	15	1	2
6	4	10	14
9	8	12	5
3	7	11	13

OUTPUT:
YES IT REPRESENTS A WONDROUS SQUARE.

PRIME	ROW INDEX	COLUMN INDEX
2	0	3
3	3	0
5	2	3
7	3	1
11	3	2
13	3	3
15	0	1

INPUT:
N = 3

1	2	4
3	7	5
8	9	6

OUTPUT:
NOT A WONDROUS SQUARE.

PRIME	ROW INDEX	COLUMN INDEX
2	0	1
3	1	0
5	1	2
7	1	1

INPUT:
N = 2

2	3
3	2

OUTPUT:

NOT A WONDROUS SQUARE.

PRIME	ROW INDEX	COLUMN INDEX
2	0	0
2	1	1

Question 3.
We would like to generate all possible anagrams of a word. For example if the given word is 'TOP', there will be 6 possible anagrams:

TOP

TPO

OPT

OTP

PTO

POT

An anagram must be printed only once. You may output the anagrams in any order. Also, output the total number of anagrams. You may assume that the number of letters, N, in the word will be 7 at most, i.e. $N \leq 7$
Test your program for the given data and some random data.

SAMPLE DATA:
INPUT:
TO

OUTPUT:
TO

OT

Total number of anagrams=2

INPUT:
LEAN

OUTPUT:
LEAN

LENA

LAEN

LANE

LNEA

LNAE

EALN

EANL

ELNA

ENLA

ENAL

ALNE

ALEN

ANLE

ANEL

AENL

AELN

NLEA

NLAE

NELA

NEAL

NALE

NAEL

Total number of anagrams = 24

Share this:

- Facebook5
- Twitter
- LinkedIn
- More
-

ISC Computer Practical -2004

December 11, 2011 2 comments

COMPUTER SCIENCE
Paper – 2
(PRACTICAL)
(Planning Session: Two hours)
(Examination Session: One hour)
————————————————————
INSTRUCTIONS

As it is a practical examination the candidate is expected to do the following:

(a) Indicate the required inputs for the problem. State the method/formula for solving the problem. Mention the required output for the selected problem.

(b) Write an algorithm for the selected problem.

(c) Write a program in C++ / JAVA. Document the program by using mnemonic names and comments.

——

Solve any **one** of the following Problems:

Question 1

Numbers have different representations depending on the bases on which they are expressed. For example in base 3, the number 12 is written as 110 ($1 \times 3^2 + 1 \times 3^1 + 0 \times 3^0$), but in base 8 it is written as 14($1 \times 8^1 + 4 \times 8^0$).

Consider, for example, the integers 12 and 5. Certainly these are not equal if base 10 is used for each. But suppose 12 was a base 3 number and 5 was a base 6 number then what happens, 12 base 3 = $1 \times 3^1 + 2 \times 3^0$, or 5 base 6 or 5 base 10 (5 in any base is equal to 5 base 10). So 12 and 5 can be equal if you select the right bases for each of them.

Write a program to input two integers, X and Y, and calculate the smallest base for X and smallest base for Y (likely different from X) so that X and Y represent the same value. The base associated with X and Y will be between 1 and 20 (both inclusive). In representing these numbers the digits 0 to 9 have their usual decimal interpretations. The upper case alphabetic characters A through J represent digits 10 through 19 respectively.

Test your program for the following data and some random data:

SAMPLE DATA:
INPUT:
X = 12, Y = 5

OUTPUT:
12(base 3) = 5 (base 6)

INPUT:
X = 10, Y = A

OUTPUT:
10 (base 10) = A (base 11)

INPUT:
X = 12, Y = 34

OUTPUT:
12 (base 17) = 34 (base 5)

INPUT:
X = 123, Y = 456

OUTPUT:
1 2 3 is not equal to 456 in any base between 2 to 20

INPUT:
X = 42, Y = 36

OUTPUT:
42 (base 7) = 36 (base 8)

Question 2.

The computer department of the Agency of International Espionage is trying to decode Intercepted messages. The agency's spies have determined that the enemy encodes Messages by first converting all characters to their ASCII values and then reversing the string.

For example, consider A_z (the underscore is just to highlight the space). The ASCII values of A,, z are 65, 32, 122 respectively. Concatenate them to get 6532122, then reverse this to get 2212356 as the coded message.

Write a program which reads a coded message and decodes it. The coded message will not exceed 200 characters. It will contain only alphabets (A …… Z, and a …….z) and spaces.

ASCII values of A ……Z are 65 ….. 90 and those of a……z are 97 …..122.

Test your Program for the following data and some random data.

SAMPLE DATA:
INPUT:
Encoded Message:
2 3 1 2 1 7 9 8 6 2 3 1 0 1 9 9 5 0 1 8 7 2 3 7 9 2 3 1 0 1 8 1 1 7 9 2 7

OUTPUT:
THE DECODED MESSAGE: Have a Nice Day

* * *

INPUT:
Encoded Message:
23 5 1 1 0 1 1 5 0 1 7 8 2 3 5 1 1 1 2 1 7 9 9 1 1 8 0 1 5 6 2 3 4 0 1 6 1 1 7 1 1 4 1 1 4 8

OUTPUT:
THE DECODED MESSAGE: Truth Always Wins

* * *

(* should be capital letters)

Question 3.
The manager of a company wants to analyse the machine usage from the records to find the utilization of the machine. He wants to know how long each user used the machine. When the user wants to use the machine he must login to the machine and after finishing the work he must log off the machine.

Each log record consists of:
User identification number.

Login time and date.

Logout time and date.

Time consists of:

Hours

Minutes

Date consists of:

Day

Month

You may assume all logins and logouts are in the same year and there are 100 users at the most. The time format is 24 hour machine hours and minutes.
Design a program:
(a) To find the duration for which each user has logged. Output all records along with the duration in hours (format hours: minutes).

(b) Output the record of the user who logged for the longest duration. You may assume that no user will login for more than 48 hours.

Test your program for the following data values and some more random data.

SAMPLE DATA:
INPUT:
Number of users: 3

USER IDENTIFICATION	LOGIN TIME & DATE		LOGOUT TIME & DATE	
149	20:10	20-12	2:50	21-12
173	12:30	20-12	12:30	21-12
142	16:20	20-12	16:30	20-12

OUTPUT:

USER IDENTIFICATION	LOGIN TIME & DATE		LOGOUT TIME & DATE		DURATION HOURS : MINS
149	20:10	20-12	2:50	21-12	6 : 40
173	12:30	20-12	12:30	21-12	24 : 00
142	16:20	20-12	16:30	20-12	00 : 10

THE USER WHO LOGGED IN FOR THE LONGEST DURATION

173	12:30	20-12	12:30	21-12	24 : 00

Share this:

- Facebook
- Twitter
- LinkedIn
- More

ISC Computer Science (Practical) -2008

December 4, 2011 4 comments

COMPUTER SCIENCE
Paper – 2
(PRACTICAL)
(Reading Time: 15 minutes)
(Planning Session AND Examination Session: Three Hours)

The total time to be spent on the Planning and the Examination Session is Three hours. After completing the Planning Session, the candidate may begin with the Examination Session. A maximum of 90 minutes is permitted to begin the Examination Session.

(Maximum Marks: 80)

INSTRUCTIONS

As it is a practical examination the candidate is expected to do the following:

(a) Write an algorithm for the selected problem.

(b) Write a program in C++/ JAVA. Document the program by using mnemonic names and comments.

(c) Test run the program on the computer using the given test data and get a printout (hardcopy) in the format specified in the problem along with the program listing.

Solve any **one** of the following Problems:

Question 1

A Smith number is a composite number, the sum of whose digits is the sum of the digits of its prime factors obtained as a result of prime factorization (excluding 1). The first few such numbers are 4, 22, 27, 58, 85, 94, 121 ……..

Example 1

1. 666

Prime factors are 2, 3, 3, and 37

Sum of the digits are (6+6+6) = 18

Sum of the digits of the factors (2+3+3+(3+7)) = 18

2. 4937775

Prime factors are 3, 5, 5, 65837

Sum of the digits are (4+9+3+7+7+7+5) = 42

Sum of the digits of the factors (3+5+5+(6+5+8+3+7)) = 42

Write a program to input a number and display whether the number is a Smith number or not.

Sample data:

Input 94 Output SMITH Number

Input 102 Output NOT SMITH Number

Input 666 Output SMITH Number

Input 999 Output NOT SMITH Number

Question 2

A sentence is terminated by either ".", or "?" followed by a space. Input a piece of text consisting of sentences. Assume that there will be a maximum of 10 sentences in block letters.

Write a program to:

(i) Obtain the length of the sentence (measured in words) and the frequency of vowels in each sentence

(ii) Generate the output as shown below using the given data

Sample Data:

INPUT

HELLO! HOW ARE YOU? HOPE EVERYTHING IS FINE. BEST OF LUCK.

OUTPUT

Sentence	No. of Vowels	No. of Words
1	2	1
2	5	3
3	8	4
4	3	3

Sentence	No. of words / vowels
1	VVVVVV
	WWW
2	VVVVVVVVVVVVVVV
	WWWWWWWWW
3	VVVVVVVVVVVVVVVVVVVVVVVVVVVVV
	WWWWWWWWWWWWWWWW
4	VVVVVVVVVVV
	WWWWWWWWWWW

Scale used: 1:3

Question 3

Given a square matrix list [ ] [ ] of order 'n'. The maximum value possible for 'n' is 20. Input the value for 'n' and the positive integers in the matrix and perform the following tasks:

Display the original matrix

1. Print the row and column position of the largest element of the matrix
2. Print the row and column position of the second largest element of the matrix
3. Sort the elements of the rows in the ascending order and display the new matrix

Sample Data:
INPUT:
N=3
LIST [ ] [ ]

5	1	3
7	4	6
9	8	2

OUTPUT

5	1	3
7	4	6
9	8	2

The largest element 9 is in row 3 and column 1

The second largest element 8 is in row 3 and column 2

Sorted list

1	3	5
4	6	7
2	8	9

Share this:

- Facebook
- Twitter
- LinkedIn
- More
-

ISC Computer Science (Practical) – 2007

December 3, 2011Leave a comment

COMPUTER SCIENCE
Paper – 2
(PRACTICAL)
(Reading Time: 15 minutes)
(Planning Session: 60 – 90 minutes)
(Examination Session: 90 – 120 minutes)
(Maximum Marks: 80)
————————————————————

INSTRUCTIONS

As it is a practical examination the candidate is expected to do the following:

(a) Write an algorithm for the selected problem.

(b) Write a program in C++ or JAVA. Document the program by using mnemonic names and comments.

(c) Test run the program on the computer using the given test data and get a printout (hardcopy) in the format specified in the problem along with the program listing.

——

Solve any **one** of the following Problems:

Question 1

Write a program to accept a date in the string format dd/mm/yyyy and accept the name of the day on 1st of January of the corresponding year. Find the day for the given date.

Example:

INPUT

Date : 5/7/2001

Day on 1st January : MONDAY

OUTPUT

Day on 5/7/2001 : THURSDAY

Test run the program on the following inputs.

INPUT DATE	DAY ON 1ST JANUARY	OUTPUT DAY FOR INPUT DATE
4/9/1998	THURSDAY	FRIDAY
31/8/1999	FRIDAY	TUESDAY
6/12/2000	SATURDAY	WEDNESDAY

The program should include the part for validating the inputs namely the date and the day on 1st January of that year.

Question 2.

The input in this problem will consist of a number of lines of English text consisting of the letters of the English alphabet, the punctuation marks (‘) apostrophe, (.) full stop, (,) comma, (;) semicolon, (:) colon and white space characters (blank, newline). Your task is to print the words of the text in reverse order without any punctuation marks other than blanks.

For example consider the following input text:

This is a sample piece of text to illustrate this problem.
If you are smart you will solve this right.

The corresponding output would read as:

right this solve will you smart are you If problem this illustrate to text of piece sample a is This

Note: That is, the lines are printed in reverse order. Individual words are not reversed

Input format:

The first line of input contains a single integer *N (< = 20),* indicating the number of lines in the input. This is followed by *N* lines of input text. Each line should accept a maximum of 80 characters.

Output format
Output the text containing the input lines in reverse order without punctuation except blanks as illustrated above.

Test your program for the following data and some random data.
SAMPLE DATA
INPUT:
2
Emotions, controlled and directed to work, is character.

By Swami Vivekananda.

OUTPUT:
Vivekananda Swami By character is work to directed and controlled Emotions

INPUT:
1

Do not judge a book by its cover.

OUTPUT:
Cover its by book a judge not Do

Question 3.
A unique-digit integer is a positive integer (without leading zeros) with no duplicate digits. For example 7, 135, 214 are all unique-digit integers whereas 33, 3121, 300 are not.

Given two positive integers *m* and *n*, write a program to determine how many unique-digit integers are there in the range between *m* and *n* (both inclusive) and output them.
The input contains two positive integers *m* and *n*. Assume *m* < 30000 and *n* < 30000. You are to output the number of unique-digit integers in the specified range along with their values in the format specified below:

SAMPLE DATA:
INPUT:
m = 100
n = 120

OUTPUT:
THE UNIQUE-DIGIT INTEGERS ARE:-
102, 103, 104, 105, 106, 107, 108, 109, 120.

FREQUENCY OF UNIQUE-DIGIT IN INTEGERS IS : 9.

INPUT:
m = 2500

n = 2550

OUTPUT:
THE UNIQUE-DIGIT INTEGERS ARE:-

2501, 2503, 2504, 2506, 2507, 2508, 2509, 2510, 2513, 2514, 2516, 2517, 2518, 2519, 2530, 2531, 2534, 2536, 2537, 2538, 2539, 2540, 2541, 2543, 2546, 2547, 2548, 2549.

FREQUENCY OF UNIQUE-DIGIT INTEGERS IS : 28

Share this:

- Facebook
- Twitter
- LinkedIn
- More
-

ISC Computer Practical Paper – 2011

June 11, 2011 5 comments

Question 1.
Write a program to input a natural number less than 1000 and display it in words.
Test your program for the given sample data and some random data.
Sample Data:
Input: 29
Output: TWENTY NINE
Input: 17001
Output: OUT OF RANGE
Input: 119
Output: ONE HUNDRED AND NINETEEN
Input:500
Output: FIVE HUNDRED
SOLUTION TO QUESTION 1:

```
import java.io.*;
public class ISC2011Q1
{
    //Declaring Instance variables
    int n;
    //Method to take input
    void inputNumber() throws IOException
    {
        BufferedReader inp=new BufferedReader(new InputStreamReader(System.in));
        System.out.println("\nINPUT:\t");
        n=Integer.parseInt(inp.readLine());
    }
    //Method to Extract digits , call other methods and display output
    public void extract()
    {
        String str="",str1="",str2="";
        int hund=0,tens=0,units=0,i=0;
        if(n<1 || n>999)
            System.out.println("\nOUT OF RANGE ");
        else
        {
            while(n!=0)
            {
                if(i==0)
                    units=n%10;
                else if(i==1)
                    tens=n%10;
                else if(i==2)
                    hund=n%10;
```

```
            i++;
            n=n/10;
        }
       if(hund>0)
            str=word1(hund)+ ” HUNDRED “;
        if(tens>1)
            str1= word2(tens);
        if(tens==1)
            str2= word3(units);
        else
            str2=word1(units);
        if((!str.equals(“”))&&(!str1.equals(“”) || !str2.equals(“”)))
            str=str+ “AND “;
        if(!str1.equals(“”))
            str=str+ str1+ ” “;
        if(!str2.equals(“”))
            str=str+ str2;
        System.out.println(“OUTPUT:\t”+str);
    }//EndElse
}
```

//Method to convert digits 0-9

```
String word1(int x)
{
    String s=””;
    switch(x)
    {
        case 1:
        s=”ONE”;
        break;
        case 2:
        s=”TWO”;
        break;
        case 3:
        s=”THREE”;
        break;
        case 4:
        s=”FOUR”;
        break;
        case 5:
        s=”FIVE”;
        break;
        case 6:
        s=”SIX”;
        break;
        case 7:
        s=”SEVEN”;
        break;
        case 8:
        s=”EIGHT”;
        break;
        case 9:
        s=”NINE”;
        break;
    }
```

```
    return s;
}
//Method to convert numbers 20, 30,40,.....90
String word2(int x)
{
    String s="";
    switch(x)
    {
        case 2:
        s="TWENTY";
        break;
        case 3:
        s="THIRTY";
        break;
        case 4:
        s="FOURTY";
        break;
        case 5:
        s="FIFTY";
        break;
        case 6:
        s="SIXTY";
        break;
        case 7:
        s="SEVENTY";
        break;
        case 8:
        s="EIGHTY";
        break;
        case 9:
        s="NINETY";
        break;
    }
    return s;
}
//Method to convert numbers 10 – 19
String word3(int x)
{
    String s="";
    switch(x)
    {
        case 0:
        s="TEN";
        break;
        case 1:
        s="ELEVEN";
        break;
        case 2:
        s="TWELVE";
        break;
        case 3:
        s="THIRTEEN";
        break;
        case 4:
```

```
            s="FOURTEEN";
            break;
            case 5:
            s="FIFTEEN";
            break;
            case 6:
            s="SIXTEEN";
            break;
            case 7:
            s="SEVENTEEN";
            break;
            case 8:
            s="EIGHTEN";
            break;
            case 9:
            s="NINETEEN";
            break;
        }
        return s;
    }
//    Main method to call other methods
public static void main(String args[]) throws IOException
    {
        ISC2011Q1 obj=new ISC2011Q1();
        obj.inputNumber();
        obj.extract();
    }
}
```

Question 2.

Encryption is a technique of coding messages to maintain their secrecy. A String array of size 'n' where 'n' is greater than 1 and less than 10, stores single sentences (each sentence ends with a full stop) in each row of the array.

Write a program to accept the size of the array. Display an appropriate message if the size is not satisfying the given condition. Define a string array of the inputted size and fill it with sentences row-wise. Change the sentence of the odd rows with an encryption of two characters ahead of the original character. Also change the sentence of the even rows by storing the sentence in reverse order. Display the encrypted sentences as per the sample data given below.

Test your program on the sample data and some random data.

Sample Data:

Input: n=4

IT IS CLOUDY.

IT MAY RAIN.

THE WEATHER IS FINE.

IT IS COOL.

Output:

KV KU ENQWFA.

RAIN MAY IT.

VJG YGCVJGT KU HKPG.

COOL IS IT.

Input: n=13

Output: INVALID ENTRY

SOLUTION TO QUESTION 2.

```
import java.io.*;
```

```
public class ISC2011Q2
{
//Declare Instance variables
    String str[]; //to store original strings
    String encryp[];//to store encrypted strings
    int n;//to store number of strings
    int count;//to be used in the methods encrypt and reverse
    //Method to input data
    void takeInput() throws IOException
    {
        BufferedReader inp=new BufferedReader(new InputStreamReader(System.in));
        String s;
        count=0;
        System.out.print("n =\t");
        n=Integer.parseInt(inp.readLine());
        if(n< 1 || n >9)
            System.out.println("\nINVALID ENTRY:");
        else
        {
            str=new String[n];
            encryp=new String[n];
            System.out.println("\nEnter sentences terminated with full stop");
            for(int x=0;x< n;x++)
            {           s=inp.readLine().trim();
                if(s.charAt(s.length()-1)!='.')
                {
                    System.out.println("\nSentence must terminate with a Full stop. Enter again.");
                    x–;
                    continue;
                }
                else
                    str[x]=s;
            }//EndFor
            callEncryption();
        }//EndElse
    }//End Method
    //Method to extract strings and encrypt
    void callEncryption()
    {
        for(int x=0;x< n;x++)
        {
            if(x%2==0)
                encrypt(str[x]);
            else
                reverse(str[x]);
        }//EndFor
    }//EndMethod
    //Method to encrypt string
    void encrypt(String s)
    {
        char ch;
        String temp="";
        int len=s.length();
        for(int x=0;x< len;x++)
```

```
        {
            ch=s.charAt(x);
            if((ch>=65 && ch<=90)||(ch>=97 && ch<=122))
            {
                ch=(char)(ch+2);
                if(ch>90 && ch<97)
                {
                    ch=(char)((64+ch-90));
                }
                else if(ch>122)
                {
                    ch=(char)((96+ch-122));
                }
            }//Endif
            temp=temp+ch;
        }//EndFor
        encryp[count]=temp;
        count++;
    }//EndMethod
    //Method to Reverse String
    void reverse(String s)
    {
        int x;
        String s1, word="";
        s=s.substring(0,s.length()-1);
        while(true)
        {
            x=s.lastIndexOf(" ");
            if(x<0)
                break;
            s1=s.substring(x).trim();
            word+=s1+"  ";
            s=s.substring(0,x).trim();
        }//EndWhile
        word+=s;
        encryp[count]=word;
        count++;
    }//EndMethod
//Method to print encrypted strings
void showOutput()
    {
System.out.println("\nOutput:");
        for(int x=0;x<count;x++)
            System.out.println(encryp[x]);
    }//EndMethod
    //Main method to call other methods
    public static void main(String args[]) throws IOException
    {
        ISC2011Q2 obj=new ISC2011Q2();
        obj.takeInput();
        obj.showOutput();
    }//EndMain
}//EndClass
```

Question 3.

Design a program which accepts your date of birth in dd mm yyyy format. Check whether the date entered is valid or not. If it is valid, display “VALID DATE”, also compute and display the day number of the year for the date of birth. If it is invalid, display “INVALID DATE” and then terminate the program.

Test your program for the given sample data and some random data.

Input:

Enter your date of birth in dd mm yyyy format
05
01
2010

Output:

VALID DATE
5

Input:

Enter your date of birth in dd mm yyyy format
03
04
2010

Output:

VALID DATE
93

Input:

Enter your date of birth in dd mm yyyy format
34
06
2010

Output:

INVALID DATE

SOLUTION FOR QUESTION 3:

```
import java.io.*;
public class ISC2011Q3
{
//Declare Instance variable
int d,m,y;
   //Method to input date
   void takeInput() throws IOException
   {
      BufferedReader inp=new BufferedReader(new InputStreamReader(System.in));
      System.out.println("Enter your date of birth in dd mm yyyy format:");
      d=Integer.parseInt(inp.readLine());
      m=Integer.parseInt(inp.readLine());
      y=Integer.parseInt(inp.readLine());
   }//EndMethod
//Method to check validity of date
   void checkValidity()
   {
      int days[]={31,28,31,30,31,30,31,31,30,31,30,31};
      int dateToDays=0;
      //Check year is leap or not
      int l=checkLeap(y);
      days[1]+=l;//to add 1 to feb if year is leap
      if(m<=0 || m>12  || d<=0 || d>days[m-1] || y<=0 )
         System.out.println("INVALID DATE");
      else
```

```
        {
            for(int x=0;x< m-1;x++)
            {
                dateToDays+=days[x];
            }//EndFor
           dateToDays+=d;//to add days of the current month
            System.out.println("\nVALID DATE\n"+ dateToDays);
        }//ENDELSE
    }//ENDMETHOD
    //Method to find and return year is leap or not
    int checkLeap(int year)
    {
        if (year% 4 == 0)
        {
            if (year % 100 != 0)
                return 1;
            else if (year % 400 == 0)
                return 1;
            else
                return 0;
        }
        else
            return 0;
    }//EndMethod
    //Main method to call other methods
    public static void main(String args[]) throws IOException
    {
        ISC2011Q3 obj=new ISC2011Q3();
        obj.takeInput();
        obj.checkValidity();
    }//EndMain
}//EndMethod
```

Share this: