**Index**

|  |  |  |
| --- | --- | --- |
| **Sno.** | **Question** | **Page no.** |
| 1. | Write a program to input a natural number less than 1000 and display it in words. | 4 |
| 2. | Write a program to change the sentence of the odd rows with an encryption of two characters ahead of the original character and even rows by storing the sentence in reverse order. | 8 |
| 3. | Write a program to print the denominations in an amount entered by the user. | 12 |
| 4. | Write a program to print Kaprekar numbers between a given range. | 15 |
| 5. | Write a program to check if a number is Smith number or not. | 18 |
| 6. | Write a program to print the unique-digit numbers between a given range. | 21 |
| 7. | Write a program to print the frequency of each element in an array. | 25 |
| 8. | Write a program to sort an array using insertion sort. | 28 |
| 9. | Write a program to check is a number is magic number or not. | 31 |
| 10. | Write a program to insert a given element at the given position in an array. | 34 |
| 11. | Write a program to delete a given element from the array | 37 |
| 12. | Write a program to find the no of times a substring is present in the main string. | 40 |
| 13. | Write a program to print the number of vowels in each word of a sentence. | 43 |
| 14. | Write a program to remove consecutively repeating characters from a string. | 46 |
| 15. | Write a program to print the frequency of each word in a string. | 49 |
| 16. | Write a program to arrange a sentence in ascending order of its word lengths. | 53 |
| 17. | Write a program to sort a sentence in alphabetical order. | 57 |
| 18. | Write a program to print the sum of border elements of a n x n matrix. | 60 |
| 19. | Write a program to sort the border elements of a n x n matrix in ascending order. | 63 |
| 20. | Write a program to print the prime elements of a matrix along with their position. | 67 |

Question 1

Write a program to input a natural number less than 1000 and display it in words.

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

INPUT: 29  
OUTPUT: TWENTY NINE  
  
INPUT: 17001  
OUTPUT: OUT OF RANGE  
  
INPUT: 119

OUTPUT: ONE HUNDRED AND NINETEEN  
  
INPUT: 500  
OUTPUT: FIVE HUNDRED

Solution

import java.util.\*;

class Question1{

public static void main(String args[])

throws InputMismatchException{

Scanner scan=new Scanner(System.in);

int n;

System.out.println("Enter a number less than 1000");

n=scan.nextInt();

if(n>=1000)

{

System.out.println("OUT OF RANGE");

}else{

String result,h="",t="",o="";

int a,b,c;

String ones[]={"one", "two","three","four","five","six","seven","eight","nine"};

String teens[]={"eleven","twelve","thirteen","fourteen","fifteen","sixteen","seventeen","eighteen","nineteen"};

String tens[]={"ten","twenty","thirty","forty","fifty","sixty","seventy","eighty","ninety"};

if(n>=100 && n<1000){

a=n/100;

result=ones[a-1]+" hundred";

n=n%100;

}

if(n%10==0 && n!=0){

a=n/10;

if(result!=null){

result+=" and ";

result+=tens[a-1];

}

else{

result=tens[a-1];

}

n=n%10;

}

if(n>20 && n<100){

a=n/10;

b=n%10;

if(result!=null){

result+=" and ";

result+=tens[a-1]+" "+ones[b-1];

}else{

result=tens[a-1]+" "+ones[b-1];

}

}

if(n>10 && n<20){

a=n%10;

if(result!=null){

result+=" and ";

result+=teens[a-1];

}else{

result=teens[a-1];

}

}

if(n<10 && n!=0){

if(result!=null)

{

result+=" and ";

result+=ones[n-1];

}else{

result=ones[n-1];

}

System.out.println("\n"+result.toUpperCase());

}

} //end of main

} //end of class

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.  
  
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

import java.io.\*;

class Question2{

public static void main(String args[])

throws IOException{

BufferedReader br=new BufferedReader(

new InputStreamReader(System.in));

int nos;

System.out.print("Enter number of sentences : ");

nos=Integer.parseInt(br.readLine());

if(nos<1 || nos>=10)

System.out.println("\nInvalid Entry");

else{

int i,j,p,l;

String s[]=new String[nos]; //meaning???

System.out.print("\nEnter "+nos+" sentences : ");

for(i=0;i< nos;i++)

s[i]=(br.readLine()).toUpperCase();

for(i=0;i< nos;i++)

{

String t;

s[i]=" "+s[i];// add a blank space before each sentence

l=s[i].length();

if(i%2==0){

t="";

for(j=0;j< l;j++){

// store the ASCII code of the character

int ch=s[i].charAt(j);

if(ch!=32 && ch!='.'){

ch=ch+2;//shift the letter two spaces

if(ch>90)//to maintain cyclic order

ch=ch-26;// subtract 26

}

//convert to character and add to a temporary string

t=t+(char)ch;

}

s[i]=t.trim();// remove leading or trailing spaces //MEANING

}else{

t="";

p=l-1;

for(j=l-1;j>=0;j--){

//reverse loop to start extraction of words

//from last to first

char ch=s[i].charAt(j);

if(ch==' '){

t=t+s[i].substring(j+1,p)+" ";

// add the extracted word and a space

p=j;

}

}

t=t+".";

s[i]=t;

}

}

System.out.println("\nOUTPUT:");

for(i=0;i< nos;i++)

System.out.print(s[i]);

}

} //end of main

} //end of class

Question 3

A bank intends to design a program to display the denomination of an input amount, upto 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 denominations. (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: 14836  
  
OUTPUT: ONE FOUR EIGHT THREE 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  
  
EXAMPLE 2:  
  
INPUT: 235001  
OUTPUT: INVALID AMOUNT

Solution

import java.util.\*;

class Question3{

public static void main(String args[])throws InputMismatchException{

Scanner scan=new Scanner(System.in);

int amt;

System.out.print("Enter a five-digit amount : ");

amt=scan.nextInt();

if(amt>99999)

{

System.out.println("INVALID AMOUNT.");

}else{

int a[]={1000,500,100,50,20,10,5,2,1};

int i,p,r,b,t;

p=amt;

for(i=0;i<a.length;i++){

t=amt/a[i];

if(t!=0){

System.out.println(a[i]+"X"+t+"="+(t\*a[i]));

amt=amt%a[i];

}

}

String ones[]={"one","two","three","four","five","six","seven","eight","nine"};

r=0;

while(p>0){

r=r\*10+p%10;

p/=10;

}

while(r>0){

b=r%10;

System.out.print(ones[b-1].toUpperCase()+" ");

r/=10;

}

}//end of if

} //end of main

} //end of class

Question 4

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. About 'kaprekar' number:  
  
A positive whole number 'n' that has 'd' number of digits is squared and split into 2 pieces, a right hand piece that has 'd' digits and a left hand piece that has remaining 'd' or 'd-1' digits. If sum of the pieces is equal to the number then it's a kaprekar number.  
  
SAMPLE DATA:  
  
INPUT:  
p=1  
Q=1000  
  
OUTPUT:  
  
THE KAPREKAR NUMBERS ARE:  
1,9,45,55,99,297,999  
  
FREQUENCY OF KAPREKAR NUMBERS IS:8

Solution

import java.util.\*;

class Question4{

public static void main(String args[])throws InputMismatchException{

Scanner scan=new Scanner(System.in);

System.out.println("Enter the range : ");

int p=scan.nextInt();

int q=scan.nextInt();

int d,i,n,a,b,s,freq;

freq=0; // to find the frequency of kaprekar numbers

System.out.println("The Kaprekar numbers are : ");

for(i=p;i<=q;i++)

{

n=i;

d=0; //to store the number of digits

//count the number of digits in the number

while(n>0){

d++;

n/=10;

}

s=i\*i; // find the square of the number

//extract 'd' digits from the right of the square of the number

a=s%(int)Math.pow(10,d);

//extract 'd' or 'd-1' digits from the left of the square of the number

b=s/(int)Math.pow(10,d);

//Check if the two parts add up to the original number i.e. Condition for Kaprekar number

if(a+b==i){

System.out.print(i+" ");

freq++;

}

}

System.out.println("\nFREQUENCY OF KAPREKAR NUMBER IS : "+freq);

} //end of main

} //end of class

Question 5

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.   
  
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

Write a program to input a number and check whether it is a smith number or not.

Sample data:  
Input: 94 Output: SMITH Number  
Input: 102 Output: NOT SMITH Number

Solution

import java.util.\*;

class Question5{

public static void main(String sr[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

System.out.println("Enter a number: ");

int n=sc.nextInt();

int p,q,i,sod=0,sopf=0,t;

p=q=n;

//Find the sum of all the digits of the number

while(p>0){

sod+=p%10;

p/=10;

}

for(i=2;i<=q;i++){

if(q%i==0){ //check if ‘i’ is a factor

t=i;

while(t>0){ //find the sum of the digits of the factor

sopf+=t%10;

t/=10;

}

q=q/i;

i--; //decrement the factor so that next time the same factor is checked again and again until it is not a factor. This is the prime factorization method.

}

}

if(sod==sopf) // if sum of digits and sum of prime factors are equal, it is smith number

System.out.println("Smith number");

else

System.out.println("Not Smith number");

} //end of main

} //end of class

Question 6

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, where m< 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 INTEGERS IS: 9.

Solution

import java.util.\*;

class Question6{

public static void main(String ar[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the range: ");

int m=sc.nextInt();

int n=sc.nextInt();

int a[]=new int[10];

int i,j,k,p,x,freq=0;

boolean flag;

System.out.println("Unique-Digit numbers are:");

for(i=m;i<=n;i++){

p=i;

x=0;

while(p>0){

a[x++]=p%10;

p/=10;

}

flag=true;

for(j=0;j<x;j++){

for(k=j+1;k<x;k++){

if(a[j]==a[k]){

flag=false;

j=x;

break;

}

}

}

if(flag){

System.out.print(i+" ");

freq++;

}

}

System.out.println("\nFrequenct of Unique-digit numbers: "+freq);

} //end of main

} //end of class

Question 7

Write a program to create an array of n integers and display the frequency of each element of the array.

Example:

Input:

Enter the number of terms: 10

Enter 10 integers: 1 2 2 2 3 4 3 4 5 6

Output:

Frequency of 1: 1

Frequency of 2: 3

Frequency of 3: 2

Frequency of 4: 2

Frequency of 5: 1

Frequency of 6: 1

Solution

import java.util.\*;

class Question7{

public static void main(String args[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the number of terms: ");

int n=sc.nextInt();

int a[]=new int[n];

System.out.println("Enter "+n+ " integers");

int i;

for(i=0;i<n;i++)

a[i]=sc.nextInt();

int j,f;

for(i=0;i<n;i++)

{

f=1;

for(j=i+1;j<n;j++)

{

if(a[i]==a[j] && a[j]!=0)

{

f++;

a[j]=0;

}

}

if(a[i]!=0)

System.out.println("Frequency of "+a[i]+" : "+f);

}

} //end of main

} //end of class

Question 8

Write a program to create an array of n integers and sort the array in ascending order using Insertion sort technique.

Example –

INPUT:

Enter the size of the array: 5

Enter 5 elements: 5 9 7 3 -4

OUTPUT:

-4 3 5 7 9

Solution

import java.util.\*;

class Question8 {

public static void main(String args[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the size of the array: ");

int n=sc.nextInt();

int a[]=new int[n];

int i,j,k;

System.out.println("Enter "+n+" integers:");

for(i=0;i<n;i++){

a[i]=sc.nextInt();

}

for(i=1;i<n;i++){

k=a[i];

for(j=i-1;j>=0 && k<a[j];j--){ //Shift the elements to the right until the condition is false

a[j+1]=a[j];

}

a[j+1]=k; //Store the element in the proper position of the array

}

System.out.println("Array in ascending order: ");

for(i=0;i<n;i++){

System.out.print(a[i]+" ");

}

}

}

Question 9

Write a program to enter a number and check if it is a magic number or not.

A number whose sum of digits ultimately yields 1 is known as a Magic number.

Example:

n = 109

109 = 9+0+1

=10 (Still a number, find the sum again)

10 =0+1

=1

The result is 1. Hence, 109 is a magic number.

n=18

18 =8+1

=9

The result is 9, which is not equal to 1, hence 18 is not a magic number.

Input: n=109

Output: 109 is a magic number

Input: n=18

Output: 18 is not a magic number.

Solution

import java.util.\*;

public class Question9 {

public static void main(String args[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

System.out.println("Enter a number: ");

int n=sc.nextInt();

int s=0,p,a;

p=n;

do{

while(p>0){

a=p%10;

s=s+a;

p=p/10;

}

if(s>9){

p=s;

s=0;

}

}while(p>0);

if(s==1)

System.out.println(n+" is a magic number");

else

System.out.println(n+" is not a magic number");

} //end of main

} //end of class

Question 10

Write a program to create an array of n elements and insert a given element at the given position in the array. Your program should display appropriate error message if the position is invalid.

Example –

INPUT:

Size of the array: 5

Input 5 integers: 1 2 4 5 6

Enter the element to be inserted: 3

Enter the position at which the element should be inserted: 3

OUTPUT:

1 2 3 4 5 6

INPUT:

Size of the array: 7

Input 5 integers: 4 3 5 8 6 2 7

Enter the element to be inserted: 13

Enter the position at which the element should be inserted: 10

OUTPUT:

Invalid position entered.

Solution

import java.util.\*;

public class Question10{

public static void main(String args[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

int i,j,k;

System.out.println("Enter the size of the array: ");

int n=sc.nextInt();

int a[]=new int[n+1];

System.out.println("Enter "+n+" integers:");

for(i=0;i<n;i++){

a[i]=sc.nextInt();

}

System.out.println("Enter the element to be inserted: ");

int e=sc.nextInt();

System.out.println("Enter the position: ");

int p=sc.nextInt();

boolean flag=false;

for(i=0;i<n;i++){

if(i+1==p){

for(j=n;j>i;j--){

a[j]=a[j-1];

}

a[i]=e;

flag=true;

n++;

break;

}

}

if(flag){

for(i=0;i<n;i++)

System.out.print(a[i]+" ");

}else{

System.out.print("Position Invalid");

}

} //end of main

} //end of class

Question 11

Write a program to create an array of n elements and delete a given element from the array. Your program should display appropriate error message if the element to be deleted is not found in the array.

Example –

Input:

Size of the array: 5

Input 5 integers: 1 2 4 5 6

Enter the element to be deleted: 4

Output:

1 2 5 6

Input:

Size of the array: 7

Input 5 integers: 4 3 5 8 6 2 7

Enter the element to be deleted: 1

Output:

Element not found in the array.

Solution

import java.util.\*;

public class Question11{

public static void main(String args[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the size of the array: ");

int n=sc.nextInt();

int a[]=new int[n];

int i,j,k;

System.out.println("Enter "+n+" integers:");

for(i=0;i<n;i++){

a[i]=sc.nextInt();

}

System.out.println("Enter the element to be deleted: ");

int e=sc.nextInt();

boolean flag=false;

for(i=0;i<n;i++){

if(a[i]==e){

for(j=i;j<n-1;j++){

a[j]=a[j+1];

}

flag=true;

n--;

break;

}

}

if(flag){

for(i=0;i<n;i++)

System.out.print(a[i]+" ");

}else{

System.out.print("Element not found");

}

} //end of main

} //end of class

Question 12

Write a program to enter a main string and a substring and find the number of times the substring is present in the main string ignoring case considerations.

Assume that substring is present only as a single word. It is neither found inside the string nor as a group of words.

Example –

Input:

Enter a main string: The man goes to the theatre.

Enter a substring: the

Output:

No of times substring is present in the main string: 2

Solution

import java.io.\*;

public class Question12{

public static void main(String args[])throws IOException{

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter a main string: ");

String ms=br.readLine();

System.out.println("Enter a substring: ");

String ss=br.readLine();

int l=ms.length();

int i,p=0,freq=0;

for(i=0;i<l;i++){

char ch=ms.charAt(i);

if(ch==' '){

String word=ms.substring(p,i);

if(word.equalsIgnoreCase(ss)){

freq++;

}

p=i+1;

}

}

System.out.println("No of times substring is present in the main string: "+freq);

} //end of main

} //end of class

Question 13

Write a program to enter a string and print each word along with the number of vowels in it.

Example –

INPUT:

Enter a string: These are wonderful times

OUTPUT:

Word Vowel

These 2

are 2

wonderful 3

times 2

Solution

import java.io.\*;

class Question13

{

public static void main(String fh[])throws IOException

{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter a sentence : ");

String str=br.readLine();

str+=" ";

int l=str.length();

String t=" ";

int i,j,c=0,p=0;

for(i=0;i<l;i++)

{

char ch=str.charAt(i);

if(ch==' ')

{

t=str.substring(p,i+1);

c=0;

for(j=0;j<t.length();j++)

{

char k=t.charAt(j);

if(k=='a'||k=='e'|| k=='i' || k=='o' ||k=='u' || k=='A' || k=='E' || k=='I' || k=='O' || k=='U')

c++;

}

System.out.println(t+"\t"+c);

p=i+1;

}

}

} //end of main

} //end of class

Question 14

Write a program to enter a string and remove consecutively repeating characters from a string.

Example –

Input:

Enter a string: The freedom of expression is immensely overused.

Output:

The fredom of expresion is imensely overused.

Solution

import java.io.\*;

public class Question14{

public static void main(String fh[])throws IOException

{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter a string : ");

String str = br.readLine();

char a[]=str.toCharArray();

int i,j,l=a.length;

for(i=0;i<l-1;i++){

if(a[i]==a[i+1]){

for(j=i;j<l-1;j++){

a[j]=a[j+1];

}

l--;

i--;

}

}

str="";

for(i=0;i<l;i++){

str+=a[i];

}

System.out.println("The modified string is: "+str);

} //end of main

} //end of class

Question 15

Write a program to enter a string and print the frequency of each word in a string.

Example –

Input:

Enter a string: The need for enlightenment is the need of the hour.

Output:

**Word Frequency**

The 3

need 2

for 1

enlightenment 1

is 1

of 1

hour 1

Solution

import java.io.\*;

class Question15

{

public static void main(String args[])throws IOException

{

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

System.out.print("Enter a String");

String s=br.readLine();

s=s+" ";

int i,t=0,p=0,f=0,j,k;

int l=s.length();

String a[]=new String[l];

String b;

for(i=0;i<l;i++)

{

if(s.charAt(i)==' ')

{

b= s.substring(p,i);

a[t]=b;

t++;

p=i+1;

}

}

for(i=0;i<t;i++)

{

f=1;

for(j=i+1;j<t;j++)

{

if(a[i].equals(a[j]))

{

f++;

for(k=j;k<t-1;k++)

{

a[k]=a[k+1];

}

t--;

j--;

}

}

System.out.println("Frequency of " + a[i] + " : " + f);

}

} //end of main

} //end of class

Question 16

Write a program to enter a sentence and print it in ascending order of its word lengths.

A sentence may either terminate with a period (.), exclamation mark (!) or a question mark (?).

Example –

INPUT:

Enter a sentence: Days are longer in summers.

OUTPUT:

in are days longer summers.

Solution

import java.io.\*;

class Question16

{

public static void main (String args[])throws IOException

{

BufferedReader br=new BufferedReader(new InputStreamReader (System.in));

System.out.print("enter the string");

String s = br.readLine();

s=s.toLowerCase();

int l= s.length();

String a[]= new String[l];

int t=0,i,p,j;

String g;

char k;

p=0;

for(i=0;i<l;i++)

{

k=s.charAt(i);

if(k==' ' || k== '.' || k==’!’ || k==’?’)// Check for delimiters to extract words

{

String b = s.substring(p,i);

a[t]= b; //Store the words in an array

t++;

p=i+1;

}

}

//Using bubble sort to arrange the string array in ascending order of word length

for(i=0;i<t;i++)

{

for(j=0;j<t-i-1;j++)

{

if(a[j].length()>a[j+1].length()) //if the word on the left is smaller in length, swap it.

{

g=a[j];

a[j]=a[j+1];

a[j+1]=g;

}

}

}

for(i=0;i<t;i++)

{

System.out.print(a[i]+” “); //print the words with a blank space

}

System.out.print("."); //display period to make it a sentence

} //end of main

} //end of class

Question 17

Write a program to enter a sentence and sort it in alphabetical order. You can convert the sentence in either uppercase or lowercase before sorting.

Example –

INPUT:

Enter a sentence: Every cloud has a silver lining.

OUTPUT:

a cloud every has lining silver

Solution

import java.io.\*;

public class Question17 {

public static void main(String args[])throws IOException{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter a string : ");

String str = br.readLine();

str=str.toLowerCase();

int l=str.length();

String a[]=new String[l];

int i,j,p=0,x=0;

for(i=0;i<l;i++){

char ch=str.charAt(i);

if(ch==' ' || ch=='.'){ //Extract the word when a space or period is found

String temp=str.substring(p,i);

a[x++]=temp; //store the word in a string array

p=i+1;

}

}

//Sort the array alphabetically

for(i=0;i<x;i++){ // limit is x since array a[] has x elements

for(j=i+1;j<x;j++){

if(a[i].compareTo(a[j])>0){ //Comparing strings

String temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

}

System.out.println("Sentence in alphabetical order: ");

for(i=0;i<x;i++)

System.out.print(a[i]+" ");

} //end of main

} //end of class

Question 18

Write a program to print the sum of border elements of an n x n matrix.

Example –

INPUT:

Enter the number of rows for a square matrix: 3

Enter 9 elements:

1 2 3

4 5 6

7 8 9

OUTPUT:

Sum of the border elements: 40

Solution

import java.util.\*;

public class Question18 {

public static void main(String args[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the number of rows for a sqaure matrix: ");

int n=sc.nextInt();

int a[][]=new int[n][n];

int i,j,s=0;

System.out.println("Enter "+(n\*n)+" elements:");

for(i=0;i<n;i++){

for(j=0;j<n;j++){

a[i][j]=sc.nextInt();

}

}

/\* Border of a matrix consists of the first row, first column,

\* last row and last column.

\*/

for(i=0;i<n;i++){

for(j=0;j<n;j++){

if(i==0 || j==0 || i==n-1 || j==n-1){

s=s+a[i][j];

}

}

}

System.out.println("Sum of border elements: "+s);

} //end of main

} //end of class

Question 19

Write a program to sort the border elements of a n x n matrix in ascending order.

Example –

INPUT:

Enter the number of rows for a square matrix: 3

Enter 9 elements:

4 9 3

1 7 6

5 8 2

OUTPUT:

Matrix with border in ascending order:

1 2 3

9 5 4

8 7 6

Solution

import java.util.\*;

public class Question19 {

public static void main(String args[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the number of rows for a sqaure matrix: ");

int n=sc.nextInt();

int a[][]=new int[n][n];

int i,j,s=0;

System.out.println("Enter "+(n\*n)+" elements:");

for(i=0;i<n;i++){

for(j=0;j<n;j++){

a[i][j]=sc.nextInt();

}

}

int b[]=new int[n\*n];

int x=0;

//Run loops to store the border elements in a single dimension array

for(i=0;i<n;i++){

for(j=0;j<n;j++){

if(i==0 || j==0 || i==n-1 || j==n-1){

b[x++]=a[i][j];

}

}

}

// Sort the array containing border elements

for(i=0;i<x;i++){

for(j=i+1;j<x;j++){

if(b[i]>b[j]){

int t=b[i];

b[i]=b[j];

b[j]=t;

}

}

}

//Now store the sorted border elements back to the matrix

int c=0;

for(i=0;i<n;i++) //Store the sorted elements in the first row

a[0][i]=b[c++];

for(i=1;i<n;i++) //Store the sorted elements in the last column

a[i][n-1]=b[c++];

for(i=n-2;i>=0;i--) //Store the sorted elements in the last row

a[n-1][i]=b[c++];

for(i=n-2;i>0;i--) //Store the sorted elements in the first column

a[i][0]=b[c++];

for(i=0;i<n;i++){

for(j=0;j<n;j++){

System.out.print(a[i][j]+" ") ;

}

System.out.println();

}

} //end of main

} //end of class

Question 20

Write a program to create a m x n matrix and print the prime elements in it along with the row and column in which they are present.

Example –

INPUT:

Enter the row and column for the matrix: 3

4

Enter 12 elements:

4 5 1 6

8 25 30 2

16 9 45 3

OUTPUT:

Prime Element Row Column

5 0 1

2 1 3

3 2 3

Algorithm

Step-1: Input the number of rows and columns as m and n

Step-2: Create a m x n matrix

Step-3: Enter m x n elements in the matrix

Step-4: Run a loop for the rows m

Step-5: Run a loop for the columns n

Step-6: Now run a loop to find the number of factors of the matrix element

Step-7: If the number of factors is 2 print the element, its row index and column index

Step-8: Continue process for all the elements

Step-9: End

Solution

import java.util.\*;

public class Question20{

public static void main(String args[])throws InputMismatchException{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the number of rows and columns of a matrix: ");

int m=sc.nextInt();

int n=sc.nextInt();

int a[][]=new int[m][n];

int i,j,k,s=0;

System.out.println("Enter "+(m\*n)+" elements:");

for(i=0;i<m;i++){

for(j=0;j<n;j++){

a[i][j]=sc.nextInt();

}

}

System.out.println("Prime Element\t Row\t Column");

for(i=0;i<m;i++){

for(j=0;j<n;j++){

int c=0;

for(k=1;k<=a[i][j];k++){

if(a[i][j]%k==0){

c++;

}

}

if(c==2){

System.out.println(a[i][j]+"\t\t"+i+"\t\t"+j);

}

}

}

} //end of main

} //end of class