**Q1)**

An emirp number is a number which is prime backwards and forwards. Example : 13 and 31 are both prime numbers. Thus 13 is a emirp number. Write a java program to check whether a number is emirp number or not.

Example:

13, 17, 31, 37, 71, 73, 79, 97, 107, 113, 149, 157, 167, 179, 199

**Algorithm**

1. START
2. Take input form user
3. Execute a loop from i=2 to i<(n/2)
4. Check if n is divisible by I from 2 to n/2
5. If I is not divisible by any value of I then n is prime
6. If I is divisible by any value of I then print “invalid input” and exit program
7. To find the reverse of n calculate the n%10 and add to the reverse number
8. Update n by n/10
9. Now check if the reverse number is prime or not
10. Repeat steps 3 to 5 to check for prime number
11. If reverse number is prime number print n is an Emirp number otherwise print n is not an Emirp number
12. END

**Source code**

import java.util.Scanner;

public class Emirp{

public static boolean isPrime(int num){

if(num<=1)

return false;

for(int i=2;i<(num/2);i++){

if(num%i==0)

return false;

}

return true;

}

public static void main(String args[]){

Scanner nrt=new Scanner(System.in);

System.out.println(“enter a number”);

int user=nrt.nextInt();

if(isPrime(user)){

//reversing the number

int copy=user,reverse=0;

while(copy!=0){

reverse = (reverse\*10) +(copy%10);

copy/=10;

}

if(isPrime(reverse))

System.out.println(user+” is an Emirp Number”);

else

System.out.println(user+” is NOT an Emirp Number”);

}

else

System.out.println(“Invalid Input”);

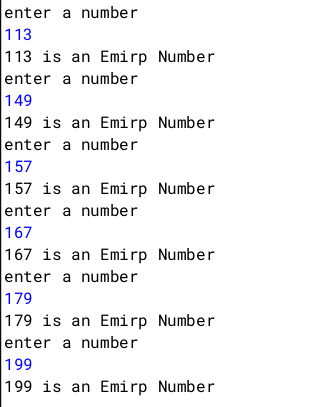
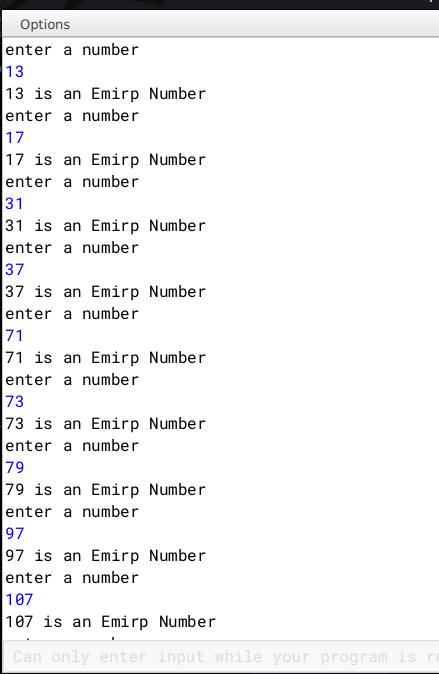
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable name | Data type | Use |
| num | int | Argument for isPrime method |
| nrt | Wrapper object | For taking input |
| user | int | To store the input |
| copy | int | To store a copy of user variable’s value |
| reverse | int | To store the reverse of user |

**Output**



**Q2)**

A Circular Prime is a prime number that remains prime under cyclic shifts of its digits. When the leftmost digit is removed and replaced at the end of the remaining string of digits, the generated number is still prime. The process is repeated until the original number is reached again.

A number is said to be prime if it has only two factors 1 and itself.

Example:

131

311

113

Hence, 131 is a circular prime.

Write a program to accept a positive number N and check whether it is a circular prime or not. The new numbers formed after the shifting of the digits should also be displayed.

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

Example 1

INPUT:

N = 197

OUTPUT:

197

971

719

197 IS A CIRCULAR PRIME.

Example 2

INPUT:

N = 1193

OUTPUT:

1193

1931

9311

3119

1193 IS A CIRCULAR PRIME.

Example 3

INPUT:

N = 29

OUTPUT:

29

92

29 IS NOT A CIRCULAR PRIME.

**Algorithm**

1. Start.

2. Input a number n.

3. Find if the number is prime.

4. Declare a function isPrime() to check prime.

5. Execute loop from i=1 to i<=n.

6. If n is divisible by any i then count the number of times it gets divisible by doing c++.

7. If c==2, then the number n is prime .

8. If not then stop otherwise go to next step.

9. Calculate the length (l) of the number by converting n into string.

10. Then find the divisor=(int)(Math.pow(10,l-1)).

11. Store a copy of n in m.

12. Execute a loop from i=0 to i<l to generate new circulated numbers.

13. Calculate the quotient=n1/divisor and the remainder=n2%divisor.

14. To generate the new circulated number(m) follow the next step.

15. m=r\*10+n1;

16. Call the function isPrime() to check whether new number(m) is prime.

17. If m is not prime then break;

18. If all the numbers are prime then print it is a circular prime otherwise not a circular

prime.

19. Stop

**Source Code**

import java.util.Scanner;

class CircularPrime{

static boolean isPrime(int num) {

int c = 0;

for (int i = 2; i <= num/2; i++)

{ if (num % i == 0){

c++;

}

}

if(c==2)

System.out.println(num+"is prime");

else

System.out.println(num+"is not prime");

return c == 2;

}

public static void main(String args[]){

Scanner in = new Scanner(System.in);

System.out.print("Enter the number: ");

int n = in.nextInt();

int f=1;

if (isPrime(n)){

//System.out.println(n);

String s=Integer.toString(n);

int l=s.length();

int divisor = (int)(Math.pow(10, l- 1));

int m = n;

for (int i = 1; i < l; i++){

int n1 = m / divisor;

int n2 = m % divisor;

m = n2 \* 10 + n1;

//System.out.println(m);

if (!isPrime(m)) {

f=0;

break;

}

}

}

else {

f=0;

}

if (f==1){

System.out.println(n + " is a circular prime.");

}

else {

System.out.println(n + " is not a circular prime.");

}

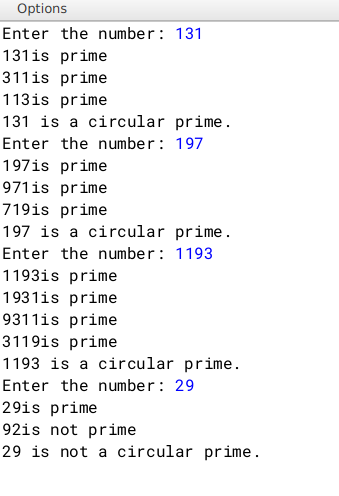
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable name | Data type | Use |
| num | int | Argument for isPrime method |
| in | Wrapper object | For taking input |
| n | int | To store the input |
| m | int | Stores a copy of n |
| n1 | int | Stores quotient |
| n2 | int | divisor |
| copy | int | To store a copy of user variable’s value |
| c | int | To check if a number is prime or not |
| l | int | Stores length of n |
| s | String | Stores String form of n |
| divisor | int | Stores the divisor |

**Output**



**Q3**)

Write a program to declare a square matrix M [ ] [ ] of order ‘N’ where ‘N’ must be greater than 3 and less than 10. Allow the user to accept three different characters from the keyboard and fill the array according to the instruction given below:

(i) Fill the four corners of the square matrix by character 1.

(ii) Fill the boundary elements of the matrix (except the four corners) by character 2.

(iii) Fill the non-boundary elements of the matrix by character 3.

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

INPUT: N = 5

FIRST CHARACTER: A

SECOND CHARACTER: C

THIRD CHARACTER: X

OUTPUT:

A C C C A

C X X X C

C X X X C

C X X X C

A C C C A

INPUT: N = 4

FIRST CHARACTER: @

SECOND CHARACTER: ?

THIRD CHARACTER: #

OUTPUT:

@ ? ? @

? # # ?

? # # ?

@ ? ? @

**Algorithm**

1 START

2 Ask user to input three characters.

3 Ask user to input the order of matrix 'M'.

4 Declare a two dimensional array of type char with name 'M'.

5 Start a for loop for(int i=0;i<n;i++).

6 Start a for loop inside as for(int j=0;j<n;j++)

7 If (i and j are both zero) or (either i is zero and j is n-1) or (i is

n-1 and j is zero) or (both i and j are n-1), then

fill two dimensional array M at location M[i][j] with the

first character the user has given as input.

8 If i and j lies between 1 and n-2 then fill two dimensional array M at

location M[i][j] with the second character user has given.

9 Else fill M[i][j] with the third character given as input by the user.

10 Run two nested loops as for(int i=0;i<n;i++) and inside

for(int j=0;j<n;j++) and print all elements.

11 END

**Source Code**

import java.util.Scanner;

public class Matrix{

public static void main(){

Scanner nrt=new Scanner(System.in);

System.out.println("Enter three characters");

char ch1,ch2,ch3;

ch1=(nrt.nextLine()).charAt(0);

ch2=(nrt.nextLine()).charAt(0);

ch3=(nrt.nextLine()).charAt(0);

System.out.println("Enter order of Matrix");

int n=nrt.nextInt();

char M[][]= new char[n][n];

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

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

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

M[i][j]=ch1;

else if ( (i>=1 && i<=(n-2)) && (j>=1 && j<=(n-2)))

M[i][j]=ch2;

else

M[i][j]=ch3;

}

}

//printing the matrix

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

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

System.out.print("\t"+M[i][j]);

}

System.out.println();

}

nrt.close();

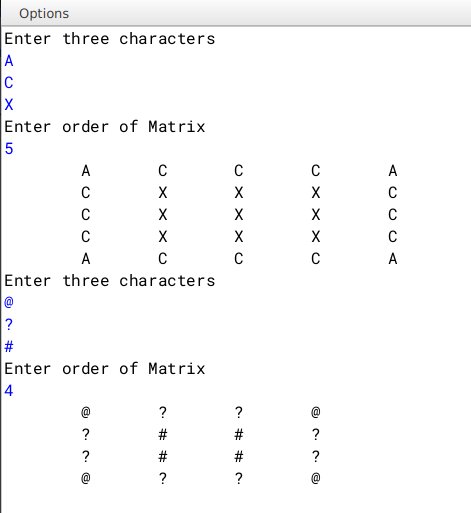
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Type | Name | Description |
| char | M | Two dimensional array to store the characters |
| int | i ,j | Loop variables |
| Int | n | Stores the order of the matrix |
| char | ch1 ,ch2,ch3 | Stores the first, second and third characters given as input by the user respectively |

**Output**



**Q4**)

Write a program to declare a square matrix a[][] of order M× M, where M is a positive integer and represents rows and columns for the matrix. M should be greater than 2 and less than 10. Accept the value of M from the user. Display an appropriate message for an invalid input.

Perform the following tasks:

a) Display the original matrix.

b) Find the sum of the elements in each row of the matrix and display them.

c) Find the sum of the elements in each column of the matrix and display them.

d) Find the sum of the elements of left and right diagonals of the matrix and display them.

Example 1:

INPUT:

M = 3

1 2 3

2 4 5

3 5 6

OUTPUT:

1 2 3

2 4 5

3 5 6

Sum of row1 = 6

Sum of row2 = 11

Sum of row3 = 14

Sum of column1 = 6

Sum of column2 = 11

Sum of column3 = 14

Sum of the left diagonal = 11

Sum of the right diagonal = 10

Example 2:

INPUT:

M = 4

7 8 9 2

4 5 6 3

8 5 3 1

7 6 4 2

OUTPUT:

7 8 9 2

4 5 6 3

8 5 3 1

7 6 4 2

Sum of row1 = 26

Sum of row2 = 18

Sum of row3 = 17

Sum of row4 = 19

Sum of column1 = 26

Sum of column2 = 24

Sum of column3 = 22

Sum of the left diagonal = 17

Sum of the right diagonal = 20

**Algorithm**

1 START

2 Take input for the order of matrix as 'n'

3 Declare and initialise a two dimensional array as int M[][] =

new int[n][n]. This will set order of 'M' as n X n

4 Initialise variables with names r\_sum, c\_sum, l\_d\_sum, r\_d\_sum( standing for

row sum, column sum, left diagonal sum, right diagonal sum) as type int

and declare their values to be zero

5 Run a for loop as for(int i=o;i<n;i++) and inside another nested loop

as for(int j=0;i<n;j++)

6 Inside nested loop, take input for two dimensional array M

7 After taking input, run another nested loop as Line 5 and print the array

8 Run another nested loop as in Line 5

9 Inside the inner loop, store value of r\_sum and c\_sum as r\_sum+= M[i][j]

and c\_sum+= M[j][i] respectively

10 Check if (i==j), if true then store value of l\_d\_sum as

l\_d\_sum+= M[i][j]

11 Check if (i+j)==(n-1), if true then store value of r\_d\_sum as

r\_d\_sum += M[i][j]

12 Outside the inner loop, print values of r\_sum and c\_sum after every iteration

and re-initialise their values with zero

13 Outside the nested loop. print the values of left diagonal sum and

right diagonal sum as l\_d\_sum and r\_d\_sum respectively

14 END

**Source Code**

import java.util.Scanner;

public class Sum{

public static void main(String args[]){

Scanner nrt=new Scanner(System.in);

int n,r\_sum=0,c\_sum=0,l\_d\_sum=0,r\_d\_sum=0;

System.out.println("Enter the order of Matrix");

n=nrt.nextInt();

int M[][]=new int[n][n];

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

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

System.out.println("Enter value at row "+i+" and column "+j);

M[i][j]=nrt.nextInt();

}

}

// printing the array

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

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

System.out.print("\t"+M[i][j]);

}

System.out.println();

}

//condition checking and finding out the sums

int i,j=0;

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

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

r\_sum+= M[i][j];

c\_sum+= M[j][i];

if(i==j)

l\_d\_sum+= M[i][j];

if((i+j)==(n-1))

r\_d\_sum+= M[i][j];

}

System.out.println("Sum of row "+i+" is "+r\_sum+"\nSum of column "+j+" is "+c\_sum);

r\_sum=0;

c\_sum=0;

}

System.out.println("Sum of left diagonal elements of the matrix = "+l\_d\_sum+"\nSum of right diagonal elements of the matrix = "+r\_d\_sum);

nrt.close();

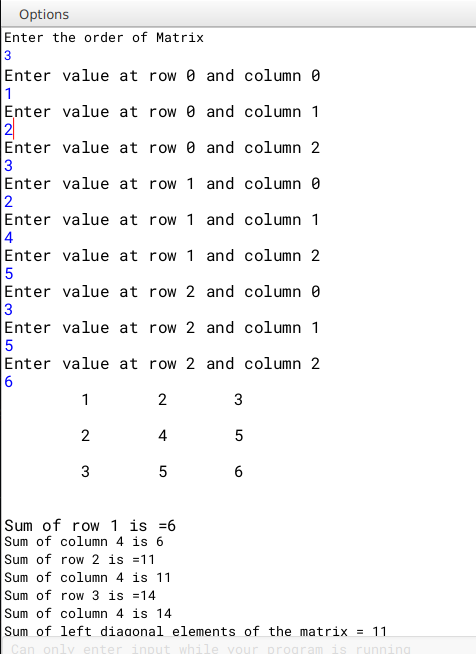
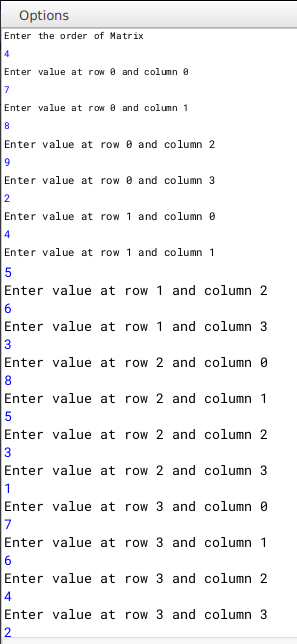
}

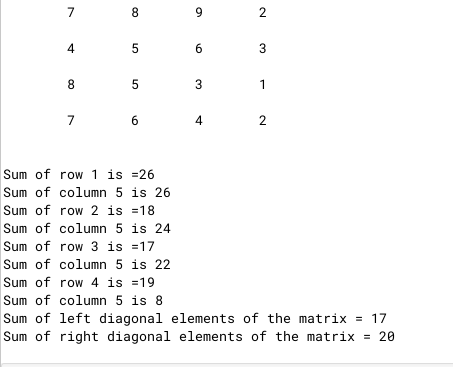
}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable Name | Type | Description |
| n | int | Order of two dimensional array |
| r\_sum | int | stores sum of a particular row |
| c\_sum | int | stores sum of a particular column |
| l\_d\_sum | int | store sum of left diagonal elements |
| r\_d\_sum | int | store sum of right diagonal elements |
| i,j | int | loop variable |
| M | int | Two dimensional array |

**Output**





**Q5)**

|  |  |
| --- | --- |
| **Class name** | **TheString** |
| **Data members/Instance variables** |  |
| Str | to store a string |
| Len | integer to store the length of the string |
| wordcount | integer to store the number of words |
| Cons | integer to store the number of consonants |
| **Member functions/Methods** |  |
| TheString | default constructor to initialize the data members |
| TheString(String ds) | parameterized constructor to assign str=ds |
| void countFreq() | To count the number of words and the number of consonants and store them in wordcount and cons respectively |
| void Display() | to display the original string, alongside the number of words and the number of consonants |

Specify the class TheString giving the details of the constructors, void countFreq() and void Display(). Define the main() method to create an object and call the functions accordingly to enable the task.

**Algorithm**

1. START

2. Declare class with name TheString

3. Declare variables str(String type) and wordcount, len, cons(all three of int

type)

4. Declare a default constructor and inside set str="", wordcount=0, len=0,

cons=0

5. Declare a parameterised constructor and set str=(ds.trim()).toUpperCase(), where ds is the parameter passed while calling the object of TheString class

6. Set len=str.length(),wordcount=0, cons=0

7. Declare a countFreq method and declare an String array d[]=str.split(" ").

This stores the words in that sentence. Set wordcount=d.length()

8. Run a for each loop to iterate through array d

9. Run a inner loop i=0; i<tmp.length() and declare a char tmp\_c=tmp.charAt(i)

10. Check if tmp\_c is a special character or not. If it is then continue

11. Check if tmp\_c is not a vowel. If not a vowel, increase value of cons by 1

12. Declare display method and print the original string, wordcount, cons

13. Declare a main method and take a string input.

14. Create an object of class TheString and pass the user input as a parameter

15. Call TheString class's countFreq and display method

**Source Code**

import java.util.Scanner;

public class TheString{

String str,copy;

int len,wordcount,cons;

TheString(){

str="";

wordcount=0;len=0;cons=0;

}

TheString(String ds){

copy=ds;

str=(ds.trim()).toUpperCase();

len=str.length();

cons=0;wordcount=0;

}

void countFreq(){

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

wordcount=d.length;

for(String tmp: d){

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

char tmp\_c=tmp.charAt(i);

if(tmp\_c=='.' || tmp\_c==',' || tmp\_c==' ' || tmp\_c=='\*' || tmp\_c=='\'')

continue;

if(tmp\_c!='A' && tmp\_c!='E' && tmp\_c!='I' && tmp\_c!='O' && tmp\_c!='U')

cons++;

}

}

}

void display(){

System.out.println("Original String ="+copy+"\nwordcount = "+wordcount+"\nnumber of consonants = "+(cons));

}

public static void main(){

Scanner nrt=new Scanner(System.in);

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

String user=nrt.nextLine();

TheString obj1= new TheString(user);

obj1.countFreq();

obj1.display();

nrt.close();

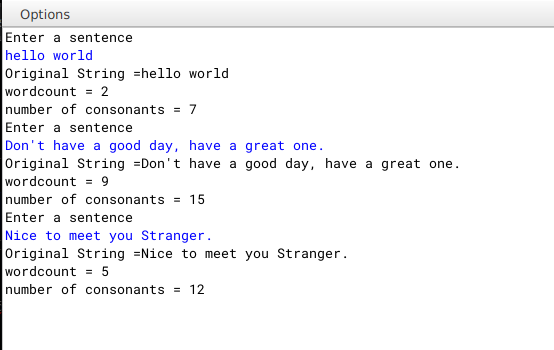
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable Name | Type | Description |
| str | String | stores the user input |
| len | int | stores length of str |
| wordcount | int | stores the number of words in str |
| cons | int | stores the number of consonants in the sentence |
| tmp | String | loop variable |
| tmp\_c | char | temporary character for condition checking |
| d | String | an array to store the words in str |
| user | String | stores user input in main method |
| copy | String | stores a copy of the original user input |
| obj1 | class object | Object of TheString |

**Output**



**Q6)**

A Happy number is a number in which the eventual sum of the square of the digits of the number is equal to 1.

e.g. 28 = (2)2 + (8)² = 4 + 64 = 68

68 = (6)² +(8) 2 = 36+ 64 = 100

100 = (1)2 + (0)2 + (0)2=1+0+0=1 notional

Hence, 28 is a happy number.

e.g. 12 = (1)2 + (2)²=1+4 = 5

Hence, 12 is not a happy number.

Design a class Happy to check if a given number is a happy number. Some of the members of the class are given below:

|  |  |
| --- | --- |
| Class name: | Happy |
| Data members/instance variables |  |
| n | store the numbers |
| Member functions |  |
| Happy () | constructor to assign 0 to n |
| void getnum (int nn) | to assign the parameter value to the number n = nn |
| int sum\_sq\_digits (int x) | returns the sum of the square of the digits of the number x |
| void ishappy() | checks if the given number is a happy number by calling the function sum\_sq\_digits (int) and displays an appropriate message. |

Specify the class Happy giving details of the constructor(), void getnum(int). int sum sq\_digits (int) and void ishappy(). Also define a main() function to create an object and call the methods to check for happy number.

**Algorithm**

1. START

2. Declare class with name Happy

3. Declare a default constructor setting value of n=0

4. Declare a getnum method to set value of n to the value the user enters

5. Declare a sum\_sq\_digits method to take x as an argument and find out the sum of square of x

6. Inside sum\_sq\_digits declare int copy=x and int sum=0

7. Run a While loop as long as copy is greater than 0. assign sum as sum += Math.pow((copy%10), 2) and copy/=10

8. Return sum

9. Declare a isHappy method to check if n is a Happy number or not

10. Inside declare int result=n and run a while loop as long as result!=1 and result!=4

11. Inside assign result=sum\_sq\_digits(result)

12. Outside the while loop, check if result is 1 or not. If result is 1 , print n is a happy number else print n is not a happy number

13. Declare a main method to take input from user.

14. Create a object of Happy class and call getnum method and pass user input as argument and call isHappy method

15. END

**Source code**

import java.util.Scanner;

public class Happy{

int n;

Happy(){

n=0;

}

void getnum(int nn){

n=nn;

}

int sum\_sq\_digits(int x){

int copy=x,sum=0;

while(copy>0){

sum +=Math.pow((copy%10),2);

copy/=10;

}

return sum;

}

void isHappy(){

int result=n;

while(result != 1 && result != 4){

result = sum\_sq\_digits(result);

}

//happy numbers end with 1

if(result==1)

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

else if(result==4)

System.out.println("Not a happy number");

}

public static void main(){

Scanner nrt=new Scanner(System.in);

System.out.println("enter a number");

int user=nrt.nextInt();

Happy H=new Happy();

H.getnum(user);

H.isHappy();

nrt.close();

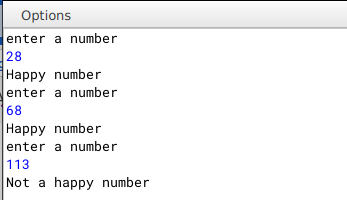
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable Name | Type | Description |
| n | int | to store user input |
| nn | int | argument for getnum method |
| x | int | argument for sum\_sq\_digits method |
| copy | int | stores a copy of x |
| sum | int | stores the sum of squares of x |
| result | int | stores value of n |
| user | int | stores user input and is required for passing as argument to getnum method |

**Output**



**Q7)**

Input a sentence from the user and count the number of times, the words “an” and “and” are present in the sentence. Design a class Frequency using the description given below:

Class name : Frequency

Data Members/ variables :

Text : stores the sentence

Countand : to store the frequency of the word “and”

Countan : to store the frequency of the word “an”

Len : stores the length of the string

Member functions / methods:

Frequency( ) :constructor to initialize the instance variables

void accept(String n):to assign n to text,where the value of the parameter n should be in lower case.

void checkandfreq( ):to count the frequency of “and”

void checkanfreq( ) :to count the frequency of “an”

void display( ) :to display the number of”and” and “an” with appropriate messages.

Specify the class Frequency giving details of the constructor( ), void accept(String),void checkandfreq(),void checkanfreq( ) and void display( ).Also define the main( ) function to create an object and call methods accordingly to enable the task.

**Algorithm**

1. START

2. Declare class with name Frequency

3. Declare variables text(String type) and countand,countan, len(all of int type

4. Declare a default constructor and inside set values of text="" and 0 for len,countand,countan

5. Declare a accept method to accept the string form user and set text=n

6. Declare a checkandfreq method to count the frequency of "and"

7. Inside declare and initialise a String array d[]=text.split(" ")

8. Run a for each loop and check if tmp.equals("and")

9. Declare a checkanfreq method to count the frquency of "an"

10. Repeat step 7

11. Run a for each loop and check if tmp.equals("an")

12. Declare a display method to print the frequencies of "and" and "an"

13. Declare a main method to take input from user.

14. Create a object of Frequency class and call accept method,checkandfreq,

checkanfreq,display

15. END

**Source Code**

import java.util.Scanner;

public class Frequency{

String text;

int countand,countan,len;

Frequency(){

text="";

countand=0;

countan=0;

len=0;}

void accept(String n){

text=(n.trim()).toLowerCase();

len=text.length();

}

void checkandfreq(){

String d[]=text.split(" ");

for(String tmp:d){

if(tmp.equals("and"))

countand++;

} }

void checkanfreq(){

String d[]=text.split(" ");

for(String tmp:d){

if(tmp.equals("an"))

countan++;

} }

void display(){

System.out.println("Frequency of and = "+countand+"\n an = "+countan+"\nlength of string = "+(text.length()));

}

public void main(){

Scanner nrt=new Scanner(System.in);

System.out.println("enter a sentence");

String user=nrt.nextLine();

Frequency F=new Frequency();

F.accept(user);

F.checkandfreq();

F.checkanfreq();

F.display();

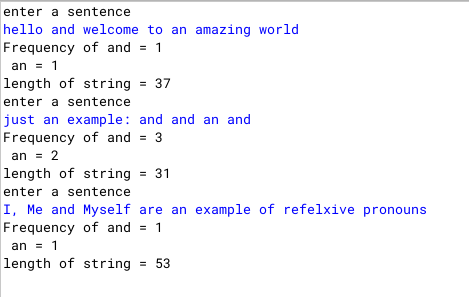
nrt.close();

} }

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable name | Type | Description |
| text | String | to store user input |
| len | int | to store length of text |
| countand | int | to store frequency of "and" |
| countan | int | to store frequency of "an" |
| tmp | String | loop variable |
| F | class object |  |
| d | String array | stores the words in text |

**Output**



**Q8)**

Two matrices are said to be equal if they have the same dimension and their corresponding elements are equal.

For example , the two matrices A and B given below are equal:

Matrix A Matrix B

1 2 3 1 2 3

2 4 5 2 4 5

3 5 6 3 5 6

Design a class EqMat to check if tow matrices are equal or not. Assume that the two matrices have the same dimension.

Class name : EqMat

Data members:

a[][] : to store integer elements

m, n : to store the number of rows and columns

Member functions:

EqMat(int mm, int nn) : initialize the data members m=mm and n=nn

void readarray() : to enter the elements in the array

int check(EqMat P, EqMat Q) : checks if the parameterized objects P and Q are equal and returns 1 if true,otherwise returns 0.

void print() : displays the array elements

Define the class and define main() to create objects and call the functions accordingly to enable the task.

**Algorithm**

1. START

2. Declare class with name EqMat

3. Declare two dimensional array int a[][] and int m,n;

4. Declare a parameterised constructor taking arguments mm and nn and inside set values of m=mm and n=nn;

5. Declare a readArray method to accept elements of the array by running two for loops as int i=0;i<m;i++ and inside int j=0;j<n;j++

6.Declare a check method to accept two objects of EqMat class check if two arrays are equal or not by running loops as int i=0;i<m;i++ and inside int j=0;j<n;j++ and inside checking if P.

7. Declare a print method to print the array by running two for loops as int i=0;i<m;i++ and inside int j=0;j<n;j++

8. Inside main method take user input for rows and columns and initialise two objects of EqMat class and pass rows and columns as parameters while creating them

9. class readArray and print method for both the objects and finally call check method to determine if the two matrix are equal or not

10. End

**Source Code**

import java.util.Scanner;

public class EqMat{

int a[][];

int m,n;

EqMat(int mm,int nn){

m=mm;

n=nn;

a=new int[m][n];

}

void readArray(){

Scanner nrt=new Scanner(System.in);

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

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

System.out.println("enter element at position: "+i+" "+j);

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

}

}

}

int check(EqMat P, EqMat Q){

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

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

if(P.a[i][j] != Q.a[i][j])

return 0;

}

}

return 1;

}

void print(){

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

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

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

}

System.out.println();

}

}

public static void main(){

Scanner nrt=new Scanner(System.in);

System.out.println("enter number of rows and columns");

int row=nrt.nextInt();

int col=nrt.nextInt();

EqMat A=new EqMat(row,col);

EqMat B=new EqMat(row,col);

A.readArray();

A.print();

B.readArray();

B.print();

if(A.check(A,B)==1)

System.out.println("Equal Matrix");

else

System.out.println("Not Equal Matrix");

nrt.close();

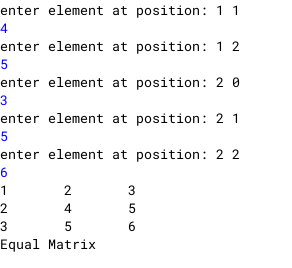
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable name | Type | Description |
| i,j | int | loop variables |
| mm, nn | int | constructor parameter |
| m | int | stores number of rows |
| n | int | stores number of columns |
| P,Q | Objects of class EqMat |  |
| a | two dimensional int array | stores the matrix |

**Output**



**Q9)**

Class name: Adder

Data member/instance variable:

a[] : integer array to hold two elements (hours and minutes)

Member functions/methods

Adder() : constructor to assign 0 to the array elements

void readtime() : to enter the elements of the array

void addtime(Adder X, Adder Y) : adds the time of the two parameterized objects X and Y and stores the sum in the current calling object

void disptime() : displays the array elements with an appropriate message (i.e. hours = and minutes =

Specify the class Adder giving details of the constructor(), void readtime(), void addtime (Adder, Adder) and void disptime(). Define the main() function to create objects and call the functions accordingly to enable the task.

Example: Time A : 6 hours 35 minutes, Time B : 7 hours 45 minutes. Their sum is 14 hours 20 minutes (where 60 minutes = 1 hour)

**Algorithm**

1. START

2. Declare a class Adder

3. Declare a int array a and h\_ans and m\_ans both to 0

4. Declare a default constructor setting the value of array a[0]=0 and a[1]=0

5. Declare a method readtime to take input of hours and minutes

6. Declare a method addtime taking two objects(X ,Y) of class Adder as parameters and inside first check if the total sum of minutes of both the time are greater than or equal to 60 or not

7. If greater than or equal to 60 find the quotient and remainder by dividing the sum by 60 and m\_sum%60 for finding the remainder respectively and setting h\_ans = quotient + X.a[0] + Y.a[0] and m\_ans =remiander

8. Else set h\_ans = X.a[0] + Y.a[0] and m\_ans= X.a[1] + Y.a[1]

9. Declare a method disptime to display the added time

10. Inside main method Declare two objects(A,B) of class Adder and call them as

A.readtime();

B.readtime();

A.addtime(A,B);

A.disptime();

11. END

**Source Code**

import java.util.Scanner;

public class Adder{

int a[]=new int[2];

int h\_ans=0,m\_ans=0;

Adder(){

a[0]=0;

a[1]=0;

}

void readtime(){

Scanner nrt=new Scanner(System.in);

System.out.println("enter hours and minutes");

a[0]=nrt.nextInt();

a[1]=nrt.nextInt();

}

void addtime(Adder X, Adder Y){

int m\_sum=X.a[1] + Y.a[1];

if(m\_sum >= 60){

int tmp= m\_sum/60;

int tmp2=m\_sum%60;

m\_ans=tmp2;

h\_ans=(tmp+ X.a[0] + Y.a[0]);

}

else{

h\_ans= X.a[0] + Y.a[0];

m\_ans= X.a[1] + Y.a[1];

} }

void disptime(){

System.out.println("hours: "+h\_ans+"\t minutes: "+m\_ans);

}

public static void main(){

Scanner nrt=new Scanner(System.in);

Adder A=new Adder();

Adder B=new Adder();

A.readtime();

B.readtime();

A.addtime(A,B);

A.disptime();

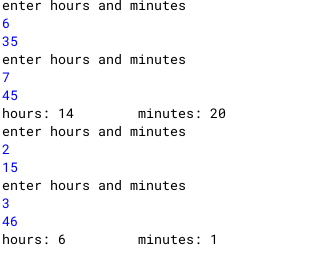
nrt.close();

} }

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable name | Type | Description |
| a | one dimensional int array | stores the hours and minutes |
| h\_ans | int | store the final hours |
| m\_ans | int | store the final minutes |
| m\_sum | int | stores the total minutes of both the time |
| tmp | int | quotient (stores the number of hours the total minutes would amount to) |
| tmp2 | int | remainder (stores the number of minutes remaining |
| A,B | objects of class Adder |  |

**Output**



**Q10**

Write a program to declare a matrix a[][] of order (M × N) 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:

Display the input matrix. Find the maximum and minimum value in each row and each column of the matrix and display them.

**Algorithm**

1. START

2. Declare a class named MaxMin

3. Declare a two dimension array as int arr[][] and initialise int M,N

4. Declare a parameterised constructor to take the rows and columns and inside set M=m; N=n and set arr=new int[m][n]

5. Declare a method readArray to take input for the array.

6. Run a loop as for(int i=0;i<M;i++) and inside for(int j=0;i<N;j++)

7. Inside take input as arr[i][j]=nrt.nextInt()

8. Declare a method display to print the array by running a loop as in LINE 6

9. Declare a method findMaxMin. Inside run a loop as for (int i=0;i<M;i++) and inside initialise int maxR=0,minR=arr[i][0]. Run another loop as for(int j=0;j<N;j++). Inside check if(maxR<arr[i][j]) then set maxR=arr[i][j]. Also check if(minR>arr[i][j]) then set minR=arr[i][j]. Outside the inner loop print the max and min value ans re-set maxR=0;minR=0.

10. Outside the loops, run another loop for columns as for(int j=0;j<N;j++) and initialize int maxC=0,minC=arr[0][j]. Run another loop as for(int i=0;i<M;i++). Inside check if(maxC<arr[i][j]) then set maxC=arr[i][j]. Also check if if(minC>arr[i][j]) then set minC=arr[i][j]. Outside the inner loop print the max and min values and re-set minC=0;maxC=0.

11. Declare a main method to take user input for number of rows and columns. Check if both rows and columns are greater than 2 and lesser than 20 or not. If not then exit showing appropriate message. Else, call the readArray , display and findMaxMin methods.

12. END

**Source Code**

import java.util.Scanner;

public class MaxMin{

int arr[][];

int M,N;

MaxMin(int m,int n){

M=m;N=n;

arr=new int[m][n];

}

void readArray(){

Scanner nrt=new Scanner(System.in);

System.out.println("enter elements:");

for(int i=0;i<M;i++){

for(int j=0;j<N;j++){

arr[i][j]=nrt.nextInt();

}

}

}

void display(){

for(int i=0;i<M;i++){

for(int j=0;j<N;j++){

System.out.print("\t"+arr[i][j]);

}

System.out.println();

}

}

void findMaxMin(){

for(int i=0;i<M;i++){// for rows

int maxR=0,minR=arr[i][0];

for(int j=0;j<N;j++){

if(maxR<arr[i][j])

maxR=arr[i][j];

if(minR>arr[i][j])

minR=arr[i][j];

}

System.out.println("Max of row: "+(i+1)+" is "+maxR+"\nMin is: "+minR);

maxR=0;minR=0;

}

for(int j=0;j<N;j++){// for columns

int maxC=0,minC=arr[0][j];

for(int i=0;i<M;i++){

if(maxC<arr[i][j])

maxC=arr[i][j];

if(minC>arr[i][j])

minC=arr[i][j];

}

System.out.println("Max of column: "+(j+1)+" is "+maxC+"\nMin is: "+minC);

minC=0;maxC=0;

}

}

public static void main(String h[]){

Scanner nrt=new Scanner(System.in);

System.out.println("enter dimensions of the matrix: ");

int row=nrt.nextInt();

int col=nrt.nextInt();

if(((row<2)||(row>20))||((col<2)||(col>20))){

System.out.println("Number of rows and columns must be greater than 2 and less than 20");

System.exit(0);

}

MaxMin M=new MaxMin(row,col);

M.readArray();

M.display();

M.findMaxMin();

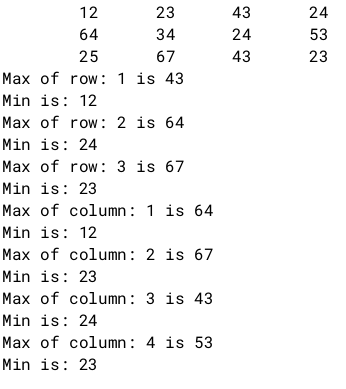
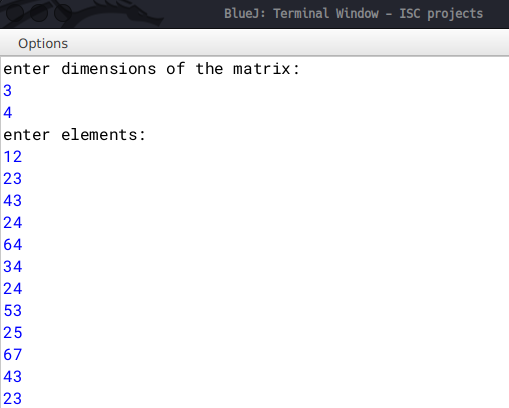
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable | Type | Description |
| arr | 2 dimensional integer array | stores the elements |
| M | int | stores the number of rows |
| N | int | stores the number of columns |
| i,j | int | loop variables |
| maxR | int | temporarily stores the max value for the current row |
| minR | int | temporarily stores the min value for the current row |
| maxC | int | temporarily stores the max value for the current column |
| minC | int | temporarily stores the min value for the current column |
| row | int | stores the no. of rows in main method |
| col | int | stores the no. of columns in main method |

**Output**



**Q11)**

Design a class to overload a function stringload() as follows:

(i) void stringload (String s, char ch1, char ch2) with one string argument and two character arguments that replaces the character argument ch1 with the character argument ch2 in the given string s and prints the new string.

Example:

Input value of s =”TECHNALAGY”

ch1=’A’,

ch2=’O’

Output : “TECHNOLOGY”

(ii) void stringload (String s) with one string argument that prints the position of the first space and the last space of the given string s.

Example:

Input value of =”Cloud computing means Internet based computing”

Output : First index : 5

Last index : 36

(iii) void stringload ( String s1, String s2 ) with two string arguments that combines the two strings with a space between them and prints the resultant string.

Example:

Input value of s1 =”COMMON WEALTH “

Input value of s2=”GAMES “

Output : COMMON WEALTH GAMES

(use library functions)

**Algorithm**

1. START

2. Declare a class Overload

3. Declare a method stringload with three arguments, String s, char ch1, char ch2.

4. Declare a String ans="". Run a loop as for(int i=0;i<s.length();i++) and inside initialise char tmp=s.charAt(i). Check if tmp==ch1. if yes then add ch2 to ans or add tmp to ans.

5. Declare a method stringload(String s) and inside using library class print the first and last positions of space.

6. Declare a method stringload(String s1, String s2). Inside concatenate s1 and s2 and store it inside ans and print ans.

7. Declare a main method and create a object of Overload class and call stringload with three different types of parameters passed.

8. End

**Source Code**

public class Overload {

void stringload(String s, char ch1, char ch2){

String ans="";

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

char tmp=s.charAt(i);

if(tmp==ch1)

ans+=ch2;

else

ans+=tmp;

}

System.out.println(ans);

}

void stringload(String s){

System.out.println("First space found at: "+(s.indexOf(' '))+"\nLast space found at: "+(s.lastIndexOf(' ')));

}

void stringload(String s1, String s2){

String ans=s1+s2;

System.out.println(ans);

}

public static void main(){

Overload O=new Overload();

System.out.println("Input: TECHNALAGY");

O.stringload("TECHNALAGY",'A','O');

System.out.println("Input: Cloud computing means Internet based computing");

O.stringload("Cloud computing means Internet based computing");

System.out.println("Input: COMMON WEALTH \nGAMES");

O.stringload("COMMON WEALTH","GAMES");

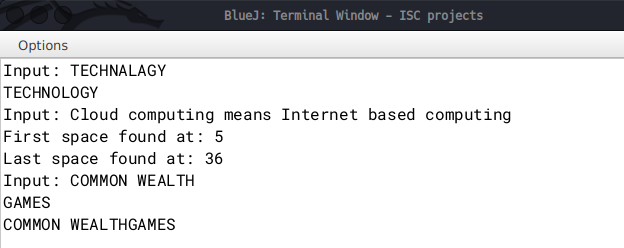
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable | Type | Description |
| ans | String | stores the string to be showed to the user after operation |
| i,j | int | loop variable |
| tmp | char | temporarily stores the character |
| s, s1, s2 | String | method parameters |
| ch1, ch2 | char | method parameters |

**Output**



**Q12)**

Write a menu driven program to sort the elements of an integer array using bubble sort, selection sort, insertion sort

**Algorithm**

1. Start

2. Declare a class named Sorting

3. Declare a one dimensional array as a, declare int n

4. Declare a parameterised constructor and take argument as int nn

5. Declare a readArray method

6. Run a loop as for(int i=0;i<n;i++)

7. Inside take input for the array

8. Declare a print method to print the array by running a loop as in LINE 6

9. Declare a method named bubblesort. Inside initialise int temp=0. Run a loop as in LINE 6. Inside the loop run another loop as for(int j=1; j < (n-i); j++).

10. Inside the inner loop check if a[j-1] > a[j]. If yes the set temp= a[j-1] and a[j-1] = a[j] and a[j] = temp

11. Declare a method named selectionsort and inside run a loop as in LINE 6 and inside initialise int index =i.

12 Inside run another loop as for (int j = i + 1; j < a.length; j++). Inside check if a[j] < a[index]. If yes, set index=j.

13. Outer the inner loop, initialise temp= a[index] and a[index]=a[i] and a[i]=temp

14. Declare a method named insertionsort. Inside initialise int sortvalue=0. Inside set sortvalue=a[i] and initialise int j. Inside the inner loop check if a[j]>sortvalue. If yes the set a[j+1]=a[j] else break the inner loop. Outside the inner loop set a[j+1]=sortvalue.

15. Declare a main method and inside take user input in int n for length of the array.Initialise an object of class Sorting. Print the options to user. Inside a switch case call the appropriate function depending on the user's choice.

16. End

**Source Code**

import java.util.Scanner;

public class Sorting{

int a[];

int n;

Sorting(int nn)

{

n=nn;

a=new int[n];

}

void readarray()

{

Scanner sc = new Scanner(System.in);

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

{

a[i]=sc.nextInt();

}

}

void print()

{

System.out.println("Array elements:");

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

{

System.out.println(a[i]);

}

}

void bubblesort()

{

int temp = 0;

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

{

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

{

if(a[j-1] > a[j])

{

//swap elements

temp = a[j-1];

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

a[j] = temp;

}

}

}

}

void selectionsort()

{

for (int i = 0; i < a.length - 1; i++)

{

int index = i;

for (int j = i + 1; j < a.length; j++){

if (a[j] < a[index]){

index = j;//searching for lowest index

}

}

int temp= a[index];

a[index] = a[i];

a[i] = temp;

}

}

void insertionsort()

{

int sortvalue=0;

for(int i=1; i<a.length; i++)

{

sortvalue=a[i];

int j;

for(j=i-1; j>=0; j--)

{

if(a[j]>sortvalue)

{

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

}

else

{

break;

}

}

a[j+1]=sortvalue;

}

}

public static void main(String args[])

{

Scanner sc = new Scanner(System.in);

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

int n=sc.nextInt();

Sorting s=new Sorting(n);

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

s.readarray();

s.print();

System.out.println("Select sorting technique:");

System.out.println("1. for bubble sort");

System.out.println("2. for selection sort");

System.out.println("3. for insertion sort");

int c=sc.nextInt();

switch(c)

{

case 1:

s.bubblesort();

s.print();

break;

case 2:

s.selectionsort();

s.print();

break;

case 3:

s.insertionsort();

s.print();

break;

}

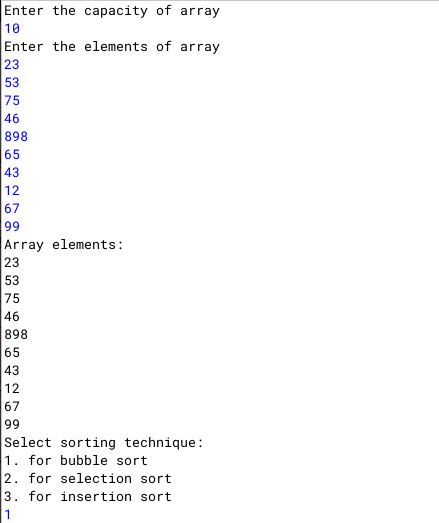
}

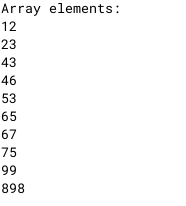
}

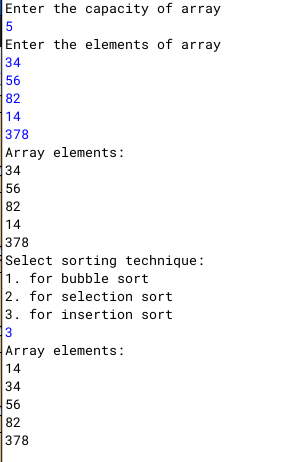
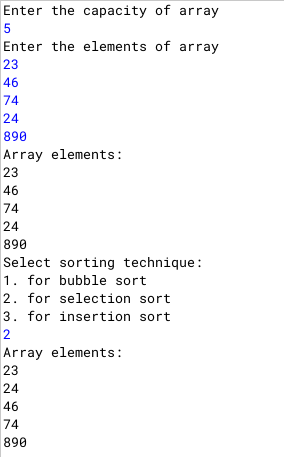
**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable | Type | Description |
| i,j | int | loop variables |
| a | Integer array | stores the unsorted array |
| n | int | stores the length of array |
| temp | int | stores temporary numbers for checking |
| index | int | stores the index of array |
| sortvalue | int | stores the element to be swapped |

**Output**







**Q13)**

Write a menu driven program in java to convert a binary to decimal and vice versa

**Algorithm**

1. START

2. Declare a class DecToBin

3. Declare a method "binary\_to\_decimal" and using recursion inside check if the argument passed is 0. If not zero return n%10+2\*binary\_to\_decimal(n/10). If argument is zero then return 0.

4.

4. Declare a method "decimal\_to\_binary" and using recursion inside check if argument passed is zero.If argument is not zero return (n%2 +10 \* decimal\_to\_binary(n/2)). If argument is zero, return 0.

5. Declare a main method print out the choice to user and input the number.

6. Check the users choice and accordingly call the methods and display the result.

7. END

**Source Code**

import java.util.Scanner;

public class DecToBin{

int binary\_to\_decimal(int n){

if(n==0)

return 0;

return n%10+2\*binary\_to\_decimal(n/10);

}

int decimal\_to\_binary(int n){

if(n==0)

return 0;

else

return (n%2 +10 \* decimal\_to\_binary(n/2));

}

public static void main(String arg[]){

Scanner nrt=new Scanner(System.in);

System.out.println("1: Binary to Decimal\n2:Decimal to Binary");

int choice=nrt.nextInt();

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

int num=nrt.nextInt();

DecToBin A=new DecToBin();

if(choice==1){

System.out.println("Decimal version of "+num+" is "+(A.binary\_to\_decimal(num)));

}

else if (choice == 2){

System.out.println("Binary version of "+num+" is "+(A.decimal\_to\_binary(num)));

}

nrt.close();

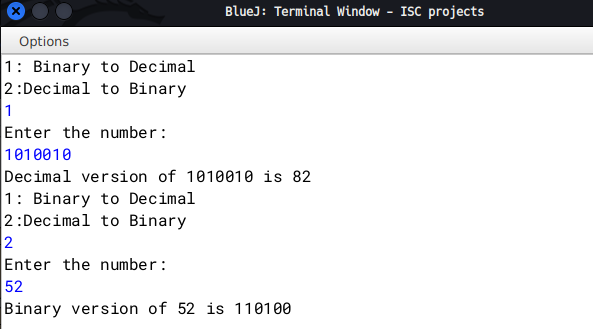
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable | Type | Description |
| n | int | method parameter |
| choice | int | stores user choice |
| num | int | stores user input |

**Output**



**Q14)**

Write a program to check if a number is Disarium or not

**Algorith**m

1. START

2. Declare a class named Disarium and initialize variables int num,size.

3. Declare a parameterised constructor taking int n as argument and inside set num=n and size=0.

4. Declare a method countDigits and inside initialize variables as int copy=num,count=0.

5. Run a while loop as long as copy is greater than 0 and post increment count. Outside the loop set size = count.

6. Declare a method SumOfDigits taking arguments as int n,int p.

7. Check if n is 0 then return 0 else return (int)(Math.pow(n%10,p))+SumOfDigits(n/10,p-1).

8. Declare a main method and take user input.

9. Initialize a object of class Disarium and call countDigits and check methods.

10. END

**Source Code**

import java.util.Scanner;

public class Disarium{

int num,size;

Disarium(int n){

num=n;

size=0;

}

void countDigits(){

int copy=num,count=0;

while(copy>0){

copy/= 10;

count++;

}

size=count;

}

int SumOfDigits(int n, int p){

if(n==0)

return 0;

else{

return (int)(Math.pow(n%10,p))+SumOfDigits(n/10,p-1);

}

}

void check(){

int sum=SumOfDigits(num,size);

if(sum==num)

System.out.println(num+" is a Disarium number");

else

System.out.println(num+" is not a Disarium number");

}

public static void main(String h[]){

Scanner nrt=new Scanner(System.in);

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

int user=nrt.nextInt();

Disarium D=new Disarium(user);

D.countDigits();

D.check();

//System.out.println("\nIs "+user+" a disarium number? \n"+(D.isDisarium(user)));

nrt.close();

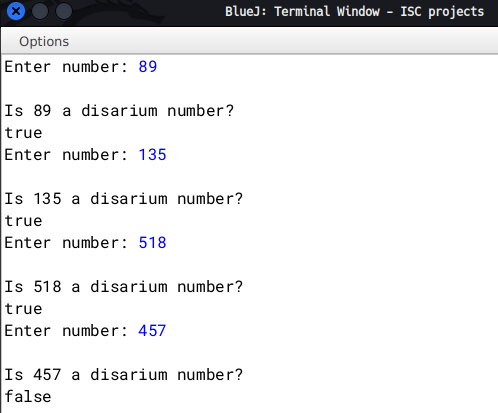
}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable name | type | Description |
| num | int | stores the original number |
| size | int | stores the length of the number |
| copy | int | stores the copy of num |
| count | int | keeps a count while counting the digits |
| user | int | stores user input |
| sum | int | stores the sum of num |

**Output**



**Q15)**

Write a program in Java to perform Binary Search recursively. Take input from user and sort it first using any method.

**Algorithm**

1. Start

2. Declare a class named RecurBinSearch

3. Declare an int array as int a[];

4. Declare a take\_input method to take user input.

5. Inside set a=new int[len], where len is the length of the array as given by the user

6. Run a loop as for(int i=0;i<len;i++), where len is the length of the array 'a'

7. Inside take individual input for the array.

8. Declare a method display to print the array by running a loop as in LINE6

9. Declare a method named insertionsort. Inside initialise int sortvalue=0. Inside set sortvalue=a[i] and initialise int j. Inside the inner loop check if a[j]>sortvalue. If yes the set a[j+1]=a[j] else break the inner loop. Outside the inner loop set a[j+1]=sortvalue.

10. Declare a method binary\_search with return type of int, taking parameters as array, target element, start position, end position. Inside check start <= end. If true continue else return -1 and end the method there.

11. Inside the if block set int mid=(start+end)/2

12. Check if(target==a[mid]), if true, return (mid+1)

13. Else check if(target > a[mid]), if true then return binary\_search(a,target, mid+1 ,end)

14. Else return binary\_search(a,target, start, mid-1)

15. Declare a main method and inside ask the user for the length of the array and store in user\_len. Call take\_input method and display method. Then call insertionsort method and print array again using the display method. Inside the Print statement call as binary\_search(A.a,t,0,((A.a).length)))

16.End

**Source Code**

import java.util.Scanner;

public class RecurBinSearch{

int a[];

void take\_input(int len){

Scanner nrt=new Scanner(System.in);

a=new int[len];

System.out.println("enter elements...");

for(int i=0;i<len;i++){

a[i]=nrt.nextInt();

}

}

void display(){

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

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

}

System.out.println();

}

void insertionsort(){

int sortvalue=0;

for(int i=1; i<a.length; i++)

{

sortvalue=a[i];

int j;

for(j=i-1; j>=0; j--)

{

if(a[j]>sortvalue)

{

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

}

else

{

break;

}

}

a[j+1]=sortvalue;

}

}

int binary\_search(int a[],int target,int start,int end){

if(start <= end){

int mid=(start+end)/2;

if(target==a[mid])

return mid+1;

else if(target > a[mid])

return binary\_search(a,target, mid+1 ,end);

else

return binary\_search(a,target, start, mid-1);

}

return -1;

}

public static void main(String s[]){

Scanner nrt=new Scanner(System.in);

System.out.println("enter length of array: ");

int user\_len=nrt.nextInt();

RecurBinSearch A=new RecurBinSearch();

A.take\_input(user\_len);

System.out.println("Original array");

A.display();

System.out.println("enter target element");

int t=nrt.nextInt();

A.insertionsort();

System.out.println("after sorting");

A.display();

System.out.println("Element found at : "+(A.binary\_search(A.a,t,0,((A.a).length))));

}

}

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| Variable | Type | Description |
| a | integer array | stores the elements in the array |
| i,j | int | loop variables |
| sortvalue | int | stores the value temporarily that is to be swapped |
| start | int | stores the start index |
| target | int | stores the element that is to be searched for |
| end | int | stores the end index |
| user\_len | int | stores the length of the array |

**Output**

