**Code Archival Project**

Siddharth Singh

Program - 1

1. Question -

Display the following series : (Preferably by arithmetic method).

½ + ¾ + ⅚ + ⅞ + ……+

0, 3, 7, 15, 24, ….n terms

1! + 4! + 7! + ….n terms

1. Program -

import java.util.\*;

public class prog1

{

public static void series1() //First Series

{

Scanner sc = new Scanner(System.in);

int num = 1;

int den = 2;

int i;

int n;

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

n = sc.nextInt();

System.out.print(num + "/" + den); //Output

num+=2;

den+=2;

for(i=2; i<=n; i++)

{

System.out.print(" + " + num + "/" + den); //Output

num+=2;

den+=2;

}

}

public static void series2() //Second Series

{

Scanner sc = new Scanner(System.in);

int d = 0;

int a = -1;

int i;

int n;

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

n = sc.nextInt();

System.out.print(d); //Output

for(i = 2; i<=n; i++)

{

if(i%2==0)

{

a = a + 4;

d = d + a;

}

else

{

a = a + 1;

d = d + a;

}

System.out.print(", " + d); //Output

}

}

public static void series3() //Third series

{

Scanner sc = new Scanner(System.in);

int n;

int fact = 1;

int x = 4;

int sum = 0;

int i;

int j;

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

n = sc.nextInt();

System.out.print(1); //Output

for(i=2; i<=n; i++)

{

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

{

fact =(fact \* j);

}

x+=3;

System.out.print(" + " + fact); //Output

fact = 1;

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Function for the first series

Step 3 - Initialize int num = 1, int den = 2

Step 4 - Declare int i, int n

Step 5 - Print “Enter the number of terms”

Step 6 - Assign the input given by the user to n

Step 7 - Print num + "/" + den

Step 8 - num = num + 2, den = den + 2

Step 9 - Function for the first series ends

Step 10 - Function for the second series

Step 11 - Initialize int d = 0, int a = -1

Step 12 - Declare int i, int n

Step 13 - Print "Enter the number of terms"

Step 14 - Assign the input given by the user to n

Step 15 - Print d

Step 16 - for(i = 2; i<=n; i++)

if(i%2==0)

a = a + 4;

d = d + a;

else

a = a + 1;

d = d + a;

Step 17 - Print ", " + d

Step 18 - Function for the second series ends

Step 19 - Function for the third series

Step 20 - Initialize int fact = 1, int x = 4, int sum = 0

Step 21 - Declare int n, int i, int j

Step 22 - Print “Enter the number of terms”

Step 23 - Assign the input given by the user to n

Step 24 - Print 1

Step 25 - for(i=2; i<=n; i++)

{

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

{

fact =(fact \* j);

}

x+=3;

Print " + " + fact

fact = 1;

}

Step 26 - Function for the third series ends

Step 27 - Stop

1. Output -

Enter the number of terms

3

1/2 + 3/4 + 5/6

Enter the number of terms

3

0, 3, 7

Enter the number of terms

3

1 + 24 + 5040

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| num | int | It stores the numerator of the series. |
| den | int | It stores the denominator of the series. |
| i | int | Counter variable in the loops. |
| j | int | Counter variable in the loops. |
| n | int | Stores the value input by the user. |
| d | int | It stores the digit to be displayed. |
| a | int | It stores the number which will be added to the digit. |
| fact | int | It stores the factorial. |
| x | int | It stores the upper limit for the for loop containing j as its counter variable. |
| sum | int | It stores the sum. |

Program - 2

1. Question -

Using Scanner class, write a program to accept a sentence and display the number of words beginning with character ‘ch’ (input from user).

       eg.Sample Input :MRS. ARORA IS AN ELECTIVE ENGLISH TEACHER.

        ch= ‘E’

       Sample Output : 2 WORDS BEGIN WITH CHARACTER ‘E’

1. Program -

import java.util.\*;

public class prog2

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the sentence"); //input

String s = sc.nextLine();

System.out.println("Enter the letter"); //input

char a = sc.next().charAt(0);

int len = s.length();

int occ = 0;

char ch;

for(int i = 0; i<len; i++) //for loop

{

ch = s.charAt(i);

if(ch==' ')

{

if(s.charAt(i+1)==a)

{

occ++;

}

}

}

System.out.println(occ + " words begin with character " + "'" + a + "'"); //output

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print "Enter the sentence"

Step 3 - Initialize String s to the sentence input by the user

Step 4 - Print "Enter the letter"

Step 5 - Initialize char a to the character input by the user

Step 6 - Initialize int len = s.length()

Step 7 - Initialize int occ = 0

Step 8 - Declare char ch

Step 9 - for(int i = 0; i<len; i++)

{

ch = s.charAt(i);

if(ch==' ')

{

if(s.charAt(i+1)==a)

{

occ++;

}

}

Step 10 - Print occ + " words begin with character " + "'" + a + "'"

Step 11 - Stop

1. Output -

Enter the sentence

He teaches his students multiple subjects

Enter the letter

s

2 words begin with character 's'

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| s | String | It stores the sentence input by the user. |
| a | char | It stores the character input by the user. |
| len | int | It stores the length of the string s. |
| occ | int | It stores the number of occurrences of the character a in the string s. |
| ch | char | It is used to extract and store each and every character of the string s. |
| i | int | Counter variable for the for loop. |

Program - 3

1. Question -

WAP to check whether a number is a Pronic Number or not .

Pronic Number : A pronic number, oblong number, rectangular number or heteromecic number, is a number which is the product of two consecutive integers, that is,

n (n + 1). 0, 2, 6, 12, 20, 30, 42, 56, 72, 90, 110, 132, 156, 182, 210, 240, 272, 306, 342, 380, 420, 462 … etc.

1. Program -

import java.util.\*;

public class prog3

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number"); //input

int n = sc.nextInt();

int sq = (int) Math.sqrt(n); //calculations

int prod = sq \* (sq + 1);

if (prod ==n)

{

System.out.println("The number input is a Pronic Number"); //output

}

else

{

System.out.println("The number input is not a Pronic Number"); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print "Enter the number"

Step 3 - Initialize int n to the number input by the user

Step 4 - Initialize int sq = (int) Math.sqrt(n)

int prod = sq \* (sq + 1)

Step 5 - if prod==n

Print "The number input is a Pronic Number"

else

Print "The number input is not a Pronic Number"

Step 6 - Stop

1. Output -

Enter the number

72

The number input is a Pronic Number

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| n | int | It stores the number input by the user. |
| sq | int | It stores the square root of n. |
| prod | int | It stores the product of sq ans sq + 1. |

Program - 4

1. Question -

Input a number and check whether the number entered by the user is an Evil number or not. An Evil number is one whose Binary counterpart has even number of ones. E.g. 10, as the binary of 10 is 1010 which has 2 ones (hence Evil), whereas numbers that has Odd number of 1’s are called Odious numbers.

1. Program -

import java.util.\*;

public class prog4

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number"); //input

int n = sc.nextInt();

String s = Integer.toBinaryString(n);

System.out.println("Binary = " + s);

int occ = 0;

char ch;

int len = s.length();

for(int i =0; i<len; i++) //for loop

{

ch = s.charAt(i);

if(ch=='1')

{

occ++;

}

}

if(occ%2==0)

{

System.out.println("It is an evil number"); //output

}

else

{

System.out.println("It is not an evil number"); ///output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print "Enter the number"

Step 3 - Initialize int n to the value input by the user

Step 4 - Initialize String s = Integer.toBinaryString(n)

Step 5 - Print "Binary = " + s

Step 6 - Initialize int occ = 0

Step 7 - Declare char ch

Step 8 - for(int i =0; i<len; i++)

{

ch = s.charAt(i);

if(ch=='1')

{

occ++;

}

}

Step 9 - if(occ%2==0)

Print "It is an evil number"

else

Print "It is not an evil number"

Step 10 - Stop

1. Output -

Enter the number

10

Binary = 1010

It is an evil number

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| n | int | It stores the number input by the user. |
| s | String | It stores the binary form of n. |
| occ | int | It stores the number of occurrences of the digit 1 in the binary form. |
| ch | char | It is used to extract and store every digit of the binary form. |
| len | int | It stores the length of s. |
| i | int | It acts as a counter variable for the for loop. |

Program - 5

1. Question -

Display the following patterns:

1. Half Pyramid
2. Inverted Half Pyramid
3. Hollow Half Pyramid
4. Full Pyramid
5. Hollow Full Pyramid
6. Hollow Inverted Half Pyramid
7. Program -

import java.util.\*;

public class prog5

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter 1 for a half pyramid");

System.out.println("Enter 2 for an inverted half pyramid");

System.out.println("Enter 3 for a hollow half pyramid");

System.out.println("Enter 4 for a full pyramid");

System.out.println("Enter 5 for a hollow full pyramid");

System.out.println("Enter 6 for a hollow inverted half pyramid");

int a = sc.nextInt();

switch(a)

{

case(1): //half pyramid

{

for(int i = 1; i<=5; i++)

{

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

{

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

}

System.out.println();

}

break;

}

case(2): // inverted half pyramid

{

for(int i = 5; i>=1; i--)

{

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

{

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

}

System.out.println();

}

break;

}

case(3): //hollow half pyramid

{

for(int i = 1; i <= 5; i++)

{

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

{

if (j == 1 || j == i || i == 5)

System.out.print(j);

else

System.out.print(" ");

}

System.out.println();

}

break;

}

case(4): //full pyramid

{

int c = 0;

int c1 = 0;

int k = 0;

for(int i = 1; i <= 5; ++i)

{

for(int j = 1; j <= 5-i; ++j)

{

System.out.print(" ");

++c;

}

while(k != 2\*i-1)

{

if (c <= 5-1)

{

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

++c;

}

else

{

++c1;

System.out.print((i+k-2\*c1) + " ");

}

++k;

}

c1 = c = k = 0;

System.out.println();

}

break;

}

case(5): //hollow full pyramid

{

for(int i = 1; i <= 5; i++)

{

for(int j = i; j < 5; j++)

{

System.out.print(" ");

}

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

{

if(j == 1 || i == 5)

{

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

}

else

{

System.out.print(" ");

}

}

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

{

if(j == i-1 && j < 5-1)

{

System.out.print(j+1);

}

else

{

System.out.print(" ");

}

}

System.out.println();

}

break;

}

case(6): //hollow inverted half pyramid

{

for(int i = 1; i <= 5; i++)

{

for(int j = i; j <= 5; j++)

{

if (i == 1 || j == 5 || j == i)

System.out.print(j);

else

System.out.print(" ");

}

System.out.println();

}

break;

}

default: //default case if the user enters any other value

{

System.out.println("Wrong Choice");

}

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print "Enter 1 for a half pyramid"

"Enter 2 for an inverted half pyramid"

"Enter 3 for a hollow half pyramid"

"Enter 4 for a full pyramid"

"Enter 5 for a hollow full pyramid"

"Enter 6 for a hollow inverted half pyramid"

Step 3 - Initialize int a to the value input by the user

Step 4 - switch(a)

{

case(1):

{

for(int i = 1; i<=5; i++)

{

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

{

Print (j + " ");

}

Print a blank line

}

break;

}

case(2):

{

for(int i = 5; i>=1; i--)

{

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

{

Print(j + " ")

}

Print a blank line

}

break;

}

case(3): {

for(int i = 1; i <= 5; i++)

{

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

{

if (j == 1 || j == i || i == 5)

Print(j)

else

Print(" ")

}

Print a blank line

}

break;

}

case(4):

{

int c = 0;

int c1 = 0;

int k = 0;

for(int i = 1; i <= 5; ++i)

{

for(int j = 1; j <= 5-i; ++j)

{

Print(" ")

++c;

}

while(k != 2\*i-1)

{

if (c <= 5-1)

{

Print(i+k + " ")

++c;

}

else

{

++c1;

Print((i+k-2\*c1) + " ")

}

++k;

}

c1 = c = k = 0;

Print a blank line

}

break;

}

case(5):

{

for(int i = 1; i <= 5; i++)

{

for(int j = i; j < 5; j++)

{

Print(" ")

}

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

{

if(j == 1 || i == 5)

{

Print(j + " ")

}

else

{

Print(" ")

}

}

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

{

if(j == i-1 && j < 5-1)

{

Print(j+1)

}

else

{

Print(" ")

}

}

System.out.println();

}

break;

}

case(6):

{

for(int i = 1; i <= 5; i++)

{

for(int j = i; j <= 5; j++)

{

if (i == 1 || j == 5 || j == i)

Print(j)

else

Print(" ")

}

Print a blank line

}

break;

}

default:

{

Print("Wrong Choice")

}

}

Step 5 - Stop

1. Output -
2. Enter 1 for a half pyramid

Enter 2 for an inverted half pyramid

Enter 3 for a hollow half pyramid

Enter 4 for a full pyramid

Enter 5 for a hollow full pyramid

Enter 6 for a hollow inverted half pyramid

1

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

1. Enter 1 for a half pyramid

Enter 2 for an inverted half pyramid

Enter 3 for a hollow half pyramid

Enter 4 for a full pyramid

Enter 5 for a hollow full pyramid

Enter 6 for a hollow inverted half pyramid

2

1 2 3 4 5

1 2 3 4

1 2 3

1 2

1

1. Enter 1 for a half pyramid

Enter 2 for an inverted half pyramid

Enter 3 for a hollow half pyramid

Enter 4 for a full pyramid

Enter 5 for a hollow full pyramid

Enter 6 for a hollow inverted half pyramid

3

1

12

1 3

1 4

12345

1. Enter 1 for a half pyramid

Enter 2 for an inverted half pyramid

Enter 3 for a hollow half pyramid

Enter 4 for a full pyramid

Enter 5 for a hollow full pyramid

Enter 6 for a hollow inverted half pyramid

4

1

2 3 2

3 4 5 4 3

4 5 6 7 6 5 4

5 6 7 8 9 8 7 6 5

1. Enter 1 for a half pyramid

Enter 2 for an inverted half pyramid

Enter 3 for a hollow half pyramid

Enter 4 for a full pyramid

Enter 5 for a hollow full pyramid

Enter 6 for a hollow inverted half pyramid

5

1

1 2

1 3

1 4

1 2 3 4 5

1. Enter 1 for a half pyramid

Enter 2 for an inverted half pyramid

Enter 3 for a hollow half pyramid

Enter 4 for a full pyramid

Enter 5 for a hollow full pyramid

Enter 6 for a hollow inverted half pyramid

6

12345

2 5

3 5

45

5

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| a | int | It is used to store the number of user’s choice. |
| i | int | Counter variable for the for loops. |
| j | int | Counter variable for the for loops. |
| k | int | It stores the output. |
| c | int | It acts as a counter variable. |
| c1 | int | It acts as a counter variable. |

Program - 6

1. Question -

o check whether the number is a magic number or not.  
[ e.g.  172 = 1+7+2 =10  
            10 = 1+0 = 1 ]  
( If the ultimate result is 1 then it is a magic number).

1. Program -

import java.util.\*;

public class prog6

{

public static void main()

{

Scanner sc = new Scanner(System.in);

int sum = 0;

int sum1 = 0;

int rem;

int rem1;

int n;

System.out.println("Enter the number"); //input

n = sc.nextInt();

while(n!=0) //while loop for the sum of the digits of the number entered

{

rem = n % 10;

sum = sum + rem;

n = n/10;

}

while(sum!=0) //while loop for the sum of the digits of the first sum

{

rem1 = sum % 10;

sum1 = sum1 + rem1;

sum = sum/10;

}

if(sum==1||sum1==1)

{

System.out.println("It is a magic number"); //output

}

else

{

System.out.println("It is not a magic number"); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Initialize int sum = 0, int sum1 = 0

Step 3 - Declare int rem, int rem1, int n

Step 4 - Print "Enter the number"

Step 5 - Assign the number entered by the user to n

Step 6 - while(n!=0)

{

rem = n % 10;

sum = sum + rem;

n = n/10;

}

Step 7 - while(sum!=0)

{

rem1 = sum % 10;

sum1 = sum1 + rem1;

sum = sum/10;

}

Step 8 - if(sum==1||sum1==1)

{

Print("It is a magic number");

}

else

{

Print("It is not a magic number");

}

Step 9 - Stop

1. Output -

Enter the number

271

It is a magic number

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| sum | int | It stores the sum of the digits of the number entered in n by the user. |
| sum1 | int | It stores the sum of the digits of the number in the variable sum. |
| rem | int | It stores the value of n%10. |
| rem1 | int | It stores the value of sum%10. |
| n | int | It stores the number input by the user. |

Program - 7

1. Question -

Write a program to input a sentence and display the largest and smallest word present in it. (first to be encountered)

 Sample input: Australia is the largest continent in the world.

                      Sample output: largest: Australia

                                                 smallest: is

1. Program -

import java.util.\*;

public class prog7

{

public static void main()

{

Scanner sc = new Scanner(System.in);

String s;

System.out.println("Enter the sentence"); //input

s = sc.nextLine();

int len = s.length();

int si = 0;

int ei = 0;

int minlen = len;

int maxlen = 0;

int minstartindex = 0;

int maxstartindex = 0;

while (ei <= len)

{

if (ei < len && s.charAt(ei) != ' ')

{

ei++;

}

else

{

int curlen = ei - si; //current length

if (curlen < minlen)

{

minlen = curlen;

minstartindex = si;

}

if (curlen > maxlen)

{

maxlen = curlen;

maxstartindex = si;

}

ei++;

si = ei;

}

}

String smallest = s.substring(minstartindex, minstartindex + minlen); //assignment of the words to their corresponding variables

String largest = s.substring(maxstartindex, maxstartindex + maxlen);

System.out.println("Largest: " + largest); //output

System.out.println("Smallest: " + smallest); //output

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Declare String s

Step 3 - Print "Enter the sentence"

Step 4 - Store the sentence entered by the user in the variable s

Step 5 - Initialize int len = s.length(), int si = 0, int ei = 0, int minlen = len, int maxlen = 0, int minstartindex = 0, int maxstartindex = 0

Step 6 - while (ei <= len)

{

if (ei < len && s.charAt(ei) != ' ')

{

ei++;

}

else

{

int curlen = ei - si; //current length

if (curlen < minlen)

{

minlen = curlen;

minstartindex = si;

}

if (curlen > maxlen)

{

maxlen = curlen;

maxstartindex = si;

}

ei++;

si = ei;

}

}

Step 7 - Initialize String smallest = s.substring(minstartindex, minstartindex + minlen), String largest = s.substring(maxstartindex, maxstartindex + maxlen)

Step 8 - Print("Largest: " + largest)

Print("Smallest: " + smallest);

Step 9 - Stop

1. Output -

Enter the sentence

I study in DPS Newtown

Largest: Newtown

Smallest: I

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| s | String | It stores the sentence input by the user. |
| len | int | It stores the length of the string s. |
| si | int | It stores the starting index of the word. |
| ei | int | It stores the ending index of the word. |
| minlen | int | It stores the length of the smallest word. |
| maxlen | int | It stores the length of the largest word. |
| minstartindex | int | It stores the starting index of the smallest word. |
| maxstartindex | int | It stores the starting index of the largest word. |
| curlen | int | It stores the current length. |
| smallest | String | It stores the smallest word. |
| largest | String | It stores the largest word. |

Program - 8

1. Question -

Write a program that encodes a word into Piglatin. To translate a word into a Piglatin word, convert the word into uppercase and then place the first vowel of the original word as the start of the new word along with the remaining alphabets. The alphabets present before the vowel being shifted towards the end followed by “AY”.  
Sample Input (1) : London, Sample Output (1) : ONDONLAY  
Sample Input (2) : Olympics, Sample Output (2) : OLYMPICSAY

1. Program -

import java.util.\*;

public class prog8

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the word"); //input

String s = sc.nextLine();

String suc = s.toUpperCase(); //conversion of word to uppercase

int len = suc.length();

char ch;

int si = 0;

for(int i = 0; i<len; i++) //for loop

{

ch = suc.charAt(i);

if(ch=='A'||ch=='E'||ch=='I'||ch=='O'||ch=='U')

{

si = i;

break;

}

else

{

continue;

}

}

if(si!=0)

{

System.out.print(suc.substring(si, len)); //output

System.out.print(suc.substring(0, si));

}

else

{

System.out.print(suc.substring(si, len)); //output

}

System.out.print("AY"); //output

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print "Enter the word"

Step 3 - Initialize String s to the value input by the user

Step 4 - Initialize String suc = s.toUpperCase()

Step 5 - Initialize int len = suc.length()

Step 6 - Declare char ch

Step 7 - Initialize int si = 0

Step 8 - for(int i = 0; i<len; i++)

{

ch = suc.charAt(i);

if(ch=='A'||ch=='E'||ch=='I'||ch=='O'||ch=='U')

{

si = i;

break;

}

else

{

continue;

}

}

Step 9 - if(si!=0)

{

Print(suc.substring(si, len))

Print(suc.substring(0, si))

}

else

{

Print(suc.substring(si, len))

}

Step 10 - Print "AY"

Step 11 - Stop

1. Output -

Enter the word

london

ONDONLAY

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| s | String | It stores the word input by the user. |
| suc | String | It stores the word stores in s in the uppercase form. |
| len | int | It stores the length of the string suc. |
| ch | char | It is used to extract and store every character of the string suc. |
| si | int | It stores the starting index. |
| i | int | It acts as a counter variable for the for loop. |

Program - 9

1. Question -

WAP that outputs the results of the following evaluations based on the choice entered by the user:

1. Prints the factors of a number taken as input by the user
2. Print the factorial of the same number
3. Check if the same number is prime or not
4. Check if the same number is perfect or not
5. Check if the same number is palindrome or not
6. Program -

import java.util.\*;

public class prog9

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number"); //input

int n = sc.nextInt();

System.out.println("Enter 1 for factors of the number"); //input menu for the user

System.out.println("Enter 2 for factorial of the number");

System.out.println("Enter 3 to check if the number is a prime number");

System.out.println("Enter 4 to check if the number is a perfect number");

System.out.println("Enter 5 to check if the number is a palindrome or not");

int c = sc.nextInt();

switch(c)

{

case(1): //factors of the numbers

{

System.out.print("The factors of " + n + " are: ");

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

{

if(n%i==0)

{

System.out.print(i + ", "); //output

}

}

break;

}

case(2): //factorial of the number

{

int fact = 1;

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

{

fact = fact \* i;

}

System.out.println("The factorial of " + n + " is: " + fact); //output

break;

}

case(3): //to check if the number is prime or not

{

int f = 0;

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

{

if(n%i==0)

{

f++;

}

}

if(f==2)

{

System.out.println("It is a prime number"); //output

}

else

{

System.out.println("It is not a prime number"); //output

}

break;

}

case(4): // to check if the number is perfect or not

{

int sum = 0;

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

{

if(n%i==0)

{

sum = sum + i;

}

}

if(sum==n)

{

System.out.println("It is a perfect number"); //output

}

else

{

System.out.println("It is not a perfect number"); //output

}

break;

}

case(5): // to check if the number is a palindrome or not

{

int pal = 0;

int rem;

int n1 = n;

while(n>0)

{

rem = n%10;

pal = pal\*10 + rem;

n = n/10;

}

if(n1==pal)

{

System.out.println("It is a palindrome number"); //output

}

else

{

System.out.println("It is not a palindrome number"); //output

}

break;

}

default: //default case if the user enters wrong input

{

System.out.println("Wrong Choice"); //output

}

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print "Enter the number"

Step 3 - Initialize int n to the value input by the user

Step 4 - Print("Enter 1 for factors of the number")

("Enter 2 for factorial of the number");

("Enter 3 to check if the number is a prime number");

("Enter 4 to check if the number is a perfect number");

("Enter 5 to check if the number is a palindrome or not");

Step 5 - Initialize int c to the value input by the user

Step 6 - switch(c)

{

case(1):

{

Print("The factors of " + n + " are: ");

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

{

if(n%i==0)

{

Print(i + ", ");

}

}

break;

}

case(2):

{

int fact = 1;

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

{

fact = fact \* i;

}

Print("The factorial of " + n + " is: " + fact);

break;

}

case(3):

{

int f = 0;

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

{

if(n%i==0)

{

f++;

}

}

if(f==2)

{

Print("It is a prime number");

}

else

{

Print("It is not a prime number");

}

break;

}

case(4):

{

int sum = 0;

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

{

if(n%i==0)

{

sum = sum + i;

}

}

if(sum==n)

{

Print("It is a perfect number");

}

else

{

Print("It is not a perfect number");

}

break;

}

case(5):

{

int pal = 0;

int rem;

int n1 = n;

while(n>0)

{

rem = n%10;

pal = pal\*10 + rem;

n = n/10;

}

if(n1==pal)

{

Print("It is a palindrome number");

}

else

{

Print("It is not a palindrome number");

}

break;

}

default:

{

Print("Wrong Choice");

}

}

Step 7 - Stop

1. Output -
2. Enter the number

121

Enter 1 for factors of the number

Enter 2 for factorial of the number

Enter 3 to check if the number is a prime number

Enter 4 to check if the number is a perfect number

Enter 5 to check if the number is a palindrome or not

1

The factors of 121 are: 1, 11, 121,

1. Enter the number

4

Enter 1 for factors of the number

Enter 2 for factorial of the number

Enter 3 to check if the number is a prime number

Enter 4 to check if the number is a perfect number

Enter 5 to check if the number is a palindrome or not

2

The factorial of 4 is: 24

1. Enter the number

17

Enter 1 for factors of the number

Enter 2 for factorial of the number

Enter 3 to check ifthe number is a prime number

Enter 4 to check if the number is a perfect number

Enter 5 to check if the number is a palindrome or not

3

It is a prime number

1. Enter the number

6

Enter 1 for factors of the number

Enter 2 for factorial of the number

Enter 3 to check if the number is a prime number

Enter 4 to check if the number is a perfect number

Enter 5 to check if the number is a palindrome or not

4

It is a perfect number

1. Enter the number

121

Enter 1 for factors of the number

Enter 2 for factorial of the number

Enter 3 to check if the number is a prime number

Enter 4 to check if the number is a perfect number

Enter 5 to check if the number is a palindrome or not

5

It is a palindrome number

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| n | int | It stores the number input by the user. |
| c | int | It stores the choice input by the user. |
| i | int | It acts as a counter variable in all for loops. |
| fact | int | It stores the factorial. |
| f | int | It stores the number of factors. |
| sum | int | It stores the sum of the factors. |
| pal | int | It stores the reverse of the number stored in n. |
| rem | int | It stores the value of n%10. |
| n1 | int | It stores the original value of n for comparison with the variable pal later. |

Program - 10

1. Question -

Design a class overloading a function calculate () as follows:

1. void calculate( int m, char ch) to check whether the number is divisible by 7 or not, if ch is 's' otherwise, it checks the last digit of the integer argument contains 7 or not.
2. void calculate (int a, int b, char ch) which displays the greater of integer arguments if ch is 'g' otherwise, display smaller of the integer arguments.
3. Program -

public class prog10

{

void calculate(int m, char ch) //input

{

if(ch=='s') //nested if-else

{

if(m%7==0)

{

System.out.println(m + " is divisible by 7"); //output

}

else

{

System.out.println(m + " is not divisible by 7"); //output

}

}

else

{

if(m%10==7)

{

System.out.println("The last digit of " + m + " contains 7"); //output

}

else

{

System.out.println("The last digit of " + m + " does not cantain 7"); //output

}

}

}

void calculate(int a, int b, char ch) //input

{

if(ch=='g') //nested if-else

{

if(a>b)

{

System.out.println("The greater number is " + a); //output

}

else

{

System.out.println("The greater number is " + b); //output

}

}

else

{

if(a<b)

{

System.out.println("The smaller number is " + a); //output

}

else

{

System.out.println("The smaller number is " + b); //output

}

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Start method void calculate(int m, char ch)

Step 3 - if(ch=='s')

{

if(m%7==0)

{

Print(m + " is divisible by 7");

}

else

{

Print(m + " is not divisible by 7");

}

}

else

{

if(m%10==7)

{

Print("The last digit of " + m + " contains 7");

}

else

{

Print("The last digit of " + m + " does not cantain 7");

}

}

Step 4 - Start method void calculate(int a, int b, char ch)

Step 5 - if(ch=='g')

{

if(a>b)

{

Print("The greater number is " + a);

}

else

{

Print("The greater number is " + b);

}

}

else

{

if(a<b)

{

Print("The smaller number is " + a);

}

else

{

Print("The smaller number is " + b);

}

}

Step 6 - Stop

1. Output -
2. Input - m = 49, ch = ‘s’

49 is divisible by 7

1. Input - a = 22, b = 64, ch = ‘g’

The greater number is 64

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| m | int | It stores the number input by the user in the first function. |
| ch | char | It stores the character input by the user in both the functions. |
| a | int | It stores the first number entered by the user in the second function. |
| b | int | It stores the second number entered by the user in the second function. |

Program - 11

1. Question -

Write a program in JAVA to find the Prime factors of a number.

Prime factors of a number are those factors which are prime in nature and by which the number itself is completely divisible (1 will not be taken as prime number).

Few such numbers are:  
Prime Factors of 24 are 2, 2, 2, 3  
Prime Factors of 6 are 2, 3

1. Program -

import java.util.\*;

public class prog11

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number"); //input

int n = sc.nextInt();

System.out.print("Prime factors of " + n + " are "); //output

int f = 0;

for(int i = 2; i<n; i++) //for loop

{

if(n%i==0)

{

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

{

if(i%j==0)

{

f++;

}

}

if(f==2) //a prime number has 2 factors

{

while(n%i==0)

{

System.out.print(i + ", "); //output

n = n/i;

}

}

}

}

if(n!=1)

{

System.out.print(n); //output, last prime factor

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print "Enter the number"

Step 3 - Initialize int n to the value input by the user

Step 4 - Print "Prime factors of " + n + " are "

Step 5 - Initialize int f = 0

Step 6 - for(int i = 2; i<n; i++)

{

if(n%i==0)

{

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

{

if(i%j==0)

{

f++;

}

}

if(f==2)

{

while(n%i==0)

{

Print(i + ", ")

n = n/i;

}

}

}

}

Step 7 - if(n!=1)

Print n

Step 8 - Stop

1. Output -

Enter the number

48

Prime factors of 48 are 2, 2, 2, 2, 3

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| n | int | It stores the number input by the user. |
| f | int | It stores the number of factors of the numbers which are factors of the number stored In n. |
| i | int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |

Program - 12

1. Question -

Mr.X has a savings bank account in BOI. He wants to make transaction from his account either by depositing some money or by making withdrawals.

Write a program to perform the above task with the following details.

Class name                :    SavingsAccount

Data members            :

    String name, ac\_type

    intacno

    floatbalance

Methods                :

    SavingsAccount(intbal)    :    assign bal to balance

    void deposit(intamt)                :    deposits amt and update balance

    void withdraw(intamt)    :    withdraws amt and maintains balance. However, if the balance becomes less than Rs. 500 then the function should display an error message “insufficient balance” and should not allow the                        user to withdraw.

void display()                  :    displays balance of the account holder.

1. Program -

import java.util.\*;

public class prog12

{

String name; //data members

String ac\_type;

int acno;

float balance;

void SavingsAccount(int bal) //input

{

balance = bal;

}

void deposit(int amt) //input

{

balance = balance + amt;

}

void withdraw(int amt) //input

{

if(balance>=500) //if condition

{

balance = balance - amt;

}

else

{

System.out.println("Insufficient Balance");

}

}

void display()

{

System.out.println("Balance: " + balance);//output

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Declare String name, String ac\_type, int acno, float balance

Step 3 - void SavingsAccount(int bal)

{

balance = bal;

}

Step 4 - void deposit(int amt)

{

balance = balance + amt;

}

Step 5 - void withdraw(int amt)

{

if(balance>=500)

{

balance = balance - amt;

}

else

{

Print("Insufficient Balance");

}

}

Step 6 - void display()

{

Print("Balance: " + balance);

}

Step 7 - Stop

1. Output -

Input - int bal = 10000, int amt = 5000(deposit), int amt = 3000(withdraw)

Balance: 12000.0

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| name | String | It stores the name of the user. |
| ac\_type | String | It stores the account type. |
| acno | int | It stores the account number. |
| balance | float | It stores the balance in the account. |
| bal | int | It stores the balance in the account. |
| amt | int | It stores the amount to be deposited. |
| amt | int | It stores the amount to be withdrawn. |

Program - 13

1. Question -

Define a class ‘Telephone’ having the following description:

Instance Variables / Data Members:

intprv    - to store the previous and present meter reading int call

intpre        - to store the calls made (i.e. pre – prv)

String name    - to store name of the customer

doubleamt    - to store the amount

doubletotal    - to store the total amount to be paid

Member Methods:

void input ( )    - to input the previous reading, present reading and name of the customer

voidcal ( )    - to calculate the amount and total amount to be paid

void display ( ) - to display the name of the customer, calls made, amount and total amount to be paid in the following format:

    Name          Calls Made             Amount         Total Amount

    ……..             …………..             ………..         ………………

Write a program to compute the monthly bill to be paid according to the given conditions:

 Calls made                                 Rate

    Up to 100 calls                             No charge

    For the next 100 calls                         90 paise per call

    For the next 200 calls                         80 paise per call

    More than 400 calls                        70 paise per call

However every customer has to pay Rs. 180 per month as monthly rent for availing the service.

1. Program -

import java.util.\*;

public class Telephone

{

int prv; //data members

int pre;

String name;

double amt;

double total;

void input() //input

{

Scanner sc = new Scanner(System.in); //input

System.out.println("Enter your name");

name = sc.nextLine();

System.out.println("Enter the previous reading");

prv = sc.nextInt();

System.out.println("Enter the present reading");

pre = sc.nextInt();

}

void cal() //calculations

{

if(pre<=100)

{

amt = 0;

total = 0 + 180;

}

else if((pre>100)&&(pre<=200))

{

amt = 0.9 \* (pre-100);

total = (0.9 \* (pre-100)) + 180;

}

else if((pre>200)&&(pre<=400))

{

amt = (0.9 \* 100) + (0.8 \* (pre-200));

total = ((0.9 \* 100) + (0.8 \* (pre-200))) + 180;

}

else if(pre>400)

{

amt = (0.9 \* 100) + (0.8 \* 200) + (0.7 \* (pre-400));

total = ((0.9 \* 100) + (0.8 \* 200) + (0.7 \* (pre-400))) + 180;

}

}

void display() //display

{

System.out.println("Name: " + name); //output

System.out.println("Calls made: " + pre);

System.out.println("Amount: " + amt);

System.out.println("Total Amount: " + total);

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Declare int prv, int pre, String name, double amt, double total

Step 3 - void input()

{

Scanner sc = new Scanner(System.in);

Print("Enter your name");

name = sc.nextLine();

Print("Enter the previous reading");

prv = sc.nextInt();

Print("Enter the present reading");

pre = sc.nextInt();

}

Step 4 - void cal()

{

if(pre<=100)

{

amt = 0;

total = 0 + 180;

}

else if((pre>100)&&(pre<=200))

{

amt = 0.9 \* (pre-100);

total = (0.9 \* (pre-100)) + 180;

}

else if((pre>200)&&(pre<=400))

{

amt = (0.9 \* 100) + (0.8 \* (pre-200));

total = ((0.9 \* 100) + (0.8 \* (pre-200))) + 180;

}

else if(pre>400)

{

amt = (0.9 \* 100) + (0.8 \* 200) + (0.7 \* (pre-400));

total = ((0.9 \* 100) + (0.8 \* 200) + (0.7 \* (pre-400))) + 180;

}

}

Step 5 - void display()

{

Print("Name: " + name);

Print("Calls made: " + pre);

Print("Amount: " + amt);

Print("Total Amount: " + total);

}

Step 6 - Stop

1. Output -

void input() -

Enter your name

Siddharth Singh

Enter the previous reading

200

Enter the present reading

500

(void cal() is run)

void display() -

Name: Siddharth Singh

Calls made: 500

Amount: 320.0

Total Amount: 500.0

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| prv | int | It stores the previous reading. |
| pre | int | It stores the present reading. |
| name | String | It stores the name of the user. |
| amt | double | It stores the amount of call charges. |
| total | double | It stores the total amount to be paid by the user. |

Program - 14

1. Question -

Write a program in Java to read two numbers and check if they are co-prime numbers or not. Use the following functions in your program-

int cacl\_hcf(int, int)    - finds HCF of two numbers

int prime (int)              - returns 0 if not prime otherwise 1

[Note: A co-prime numbers are numbers which are both prime and their HCF is 1]

1. Program -

import java.util.\*;

public class prog14

{

int cacl\_hcf(int a, int b) //input

{

int hcf = 0;

for(int i = 1; i<=a&&i<=b; i++) //for loop

{

if((a%i==0)&&(b%i==0))

{

if(i>hcf)

{

hcf = i;

}

}

}

return (hcf);

}

int prime(int n) //input

{

int f = 0;

for(int i = 1; i<=n; i++) //for loop

{

if(n%i==0)

{

f++;

}

}

if(f==2)

{

return (1);

}

else

{

return(0);

}

}

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the first number"); //input

int x = sc.nextInt();

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

int y = sc.nextInt();

prog14 obj = new prog14();

int l = obj.cacl\_hcf(x, y);

int m = obj.prime(x);

int n = obj.prime(y);

if((l==1)&&(m==1)&&(n==1))

{

System.out.println("The two numbers are co-prime"); //output

}

else

{

System.out.println("The two numbers are not co-prime"); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - int cacl\_hcf(int a, int b)

{

int hcf = 0;

for(int i = 1; i<=a&&i<=b; i++)

{

if((a%i==0)&&(b%i==0))

{

if(i>hcf)

{

hcf = i;

}

}

}

return (hcf);

}

Step 3 - int prime(int n)

{

int f = 0;

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

{

if(n%i==0)

{

f++;

}

}

if(f==2)

{

return (1);

}

else

{

return(0);

}

}

Step 4 - public static void main()

{

Scanner sc = new Scanner(System.in);

Print("Enter the first number");

int x = sc.nextInt();

Print("Enter the second number");

int y = sc.nextInt();

prog14 obj = new prog14();

int l = obj.cacl\_hcf(x, y);

int m = obj.prime(x);

int n = obj.prime(y);

if((l==1)&&(m==1)&&(n==1))

{

Print("The two numbers are co-prime");

}

else

{

Print("The two numbers are not co-prime");

}

}

Step 5 - Stop

1. Output -

Enter the first number

17

Enter the second number

19

The two numbers are co-prime

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| a | int | It stores the first number input by the user. |
| b | int | It stores the second number input by the user. |
| hcf | int | It stores the hcf of the two numbers a and b. |
| n | int | It stores the number input by the user. |
| f | int | It stores the number of factors of the number n. |
| i | int | It acts as a counter variable. |
| x | int | It stores the first number input by the user. |
| y | int | It stores the second number input by the user. |
| l | int | It stores the value of hcf returned from int cacl\_hcf(int a, int b) by giving x and y as an input. |
| m | int | It stores the value returned from int prime(int n) by giving x as an input. |
| n | int | It stores the value returned from int prime(int n) by giving y as an input. |

Program - 15

1. Question -

An ISBN ( InternationalStanadar 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 whehter ISBN is correct or not. Each of the first nine digits of the code can take a value between 0 and 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 and so on until we add 1 time the last digit. If the final number leaves no remainder when divided by 11, the code is a valid ISBN.   
For example:   
1. 02011003311 = 10\*0 + 9\*2 + 8\*0 + 7\*1 + 6\*1 + 5\*0 + 4\*3 + 3\*3 + 2\*1 + 1\*1 = 55   
Since 55 leaves no remainder when divisible by 11, hence it is a valid ISBN.   
Design a program to accept a ten digit code from the user. For an invalid inout, display an   
appropriate message. Verify the code for its validity in the format specified below:   
Test your program with sample data and some random data.   
Example 1   
INPUT CODE : 0201530821   
OUTPUT : SUM = 99   
LEAVES NO REMAINDER - VALID ISBN CODE

1. Program -

import java.util.\*;

public class prog15

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the ISBN number"); //input

long isbn = sc.nextLong();

long sum = 0;

for(int i = 1; i<=10; i++) //for loop

{

long rem = isbn%10;

sum = sum + (rem \* i);

isbn = isbn/10;

}

System.out.println("Sum: " + sum);

if(sum%11==0)

{

System.out.println("Valid ISBN Code"); //output

}

else

{

System.out.println("Invalid ISBN Code"); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print "Enter the ISBN number"

Step 3 - Initialize long isbn to the number in put by the user

Step 4 - Initialize long sum = 0

Step 5 - for(int i = 1; i<=10; i++)

{

long rem = isbn%10;

sum = sum + (rem \* i);

isbn = isbn/10;

}

Step 6 - Print("Sum: " + sum);

Step 7 - if(sum%11==0)

{

Print("Valid ISBN Code");

}

else

{

Print("Invalid ISBN Code");

}

Step 8 - Stop

1. Output -

Enter the ISBN number

0201530821

Sum: 99

Valid ISBN Code

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| isbn | long | It stores the ISBN number entered by the user. |
| sum | long | It stores the sum of the pattern given to validate an ISBN number. |
| i | int | It acts as a counter variable. |
| rem | long | It stores the value of isbn%10. |

Program - 16

1. Question -

Write a program to accept 15 integers from the keyboard, assuming that no integer entered is a zero. perform Selection Sort on the integers and then print them in ascending order

1. Program -

import java.util.\*;

public class prog16

{

public static void main()

{

Scanner sc=new Scanner(System.in);

int i;

int temp;

int j;

int min;

int arr[]=new int[15]; //array storing the numbers

for(i=0;i<15;i++) //for loop

{

System.out.println("Enter the Numbers"); //input

arr[i]=sc.nextInt();

}

for(i=0;i<14;i++) //for loop

{

min=i;

for(j=i+1;j<15;j++) //for loop

{

if(arr[j]<arr[min])

min=j;

}

temp= arr[i];

arr[i]= arr[min];

arr[min]=temp;

}

System.out.println("The numbers arranged in ascending order are:"); //output

for(i=0;i<15;i++)

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

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Declare int I, int temp, int j, int min

Step 3 - Create array int arr[]=new int[15]

Step 4 - for(i=0;i<15;i++) //for loop

{

Print("Enter the Numbers");

arr[i]=sc.nextInt();

}

Step 5 - for(i=0;i<14;i++)

{

min=i;

for(j=i+1;j<15;j++)

{

if(arr[j]<arr[min])

min=j;

}

temp= arr[i];

arr[i]= arr[min];

arr[min]=temp;

}

Step 6 - Print("The numbers arranged in ascending order are:");

for(i=0;i<15;i++)

Print(arr[i]);

Step 7 - Stop

1. Output -

Enter the Numbers

21

Enter the Numbers

44

Enter the Numbers

36

Enter the Numbers

71

Enter the Numbers

88

Enter the Numbers

30

Enter the Numbers

96

Enter the Numbers

105

Enter the Numbers

226

Enter the Numbers

43

Enter the Numbers

22

Enter the Numbers

75

Enter the Numbers

10

Enter the Numbers

8

Enter the Numbers

62

The numbers arranged in ascending order are:

8

10

21

22

30

36

43

44

62

71

75

88

96

105

226

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| i | int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |
| temp | int | It is used to exchange the values of arr[i] and arr[min]. |
| min | int | It stores the position of the minimum value in the array. |

Program - 17

1. Question -

Design a Calculator class that performs calculations only on integer values. Some functions and data members are given below:

Data member: result

Default constructor: to Create an object with default value of result.

Copy constructor: Creates a duplicate of an existing object of the class.

inputValue(int x): to enter the first operand

display(): to display the result

add(int x): to add argument to the result

multiply(int x): to multiply the argument with the result

setZero(): to set the result to value 0.

1. Program -

public class Calculator

{

int result;//data member

public Calculator() //default constructor

{

}

public Calculator(Calculator obj) //copy constructor

{

Calculator obj1 = new Calculator();

}

void inputValue(int x) //input

{

result = x;

}

void display()

{

System.out.println("Result: " + result); //output

}

void add(int x) //input number to be added to result

{

result = result + x;

}

void multiply(int x) //input number to be multiplied to result

{

result = result \* x;

}

void setZero() //set result to 0

{

result = 0;

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Declare int result

Step 3 - public Calculator()

{

}

Step 4 - public Calculator(Calculator obj)

{

Calculator obj1 = new Calculator();

}

Step 5 - void inputValue(int x)

{

result = x;

}

Step 6 - void display()

{

Print("Result: " + result);

}

Step 7 - void add(int x)

{

result = result + x;

}

Step 8 - void multiply(int x)

{

result = result \* x;

}

Step 9 - void setZero() //set result to 0

{

result = 0;

}

Step 10 - Stop

1. Output -

Input - void inputValue(int x), int x = 200

void multiply(int x), int x = 5

void display()

Result: 1000

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| result | int | It stores the result. |
| x | int | It stores the number input by the user. |

Program - 18

1. Question -

Write a menu driven program to accept a number and check and display whether it is a prime number, automorphic number or perfect number.

Prime number: a number is said to be a prime number if it is divisible only by 1 and itself and no other number. Example: 3, 5, 7, 11, 13, etc.…

Automorphic number: An automorphic number is the number which is contained in the last digits of its square. Ex: 25 is automorphic as its contained in its square 625

Perfect number: Perfect number is a number if it equals to the sum of its factor other than the number itself. Example 6=1+2+3

1. Program -

import java.util.\*;

public class prog18

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter 1 to check if the number is a prime number or not"); //menu for the user to input his choice

System.out.println("Enter 2 to check if the number is an automorphic number or not");

System.out.println("Enter 3 to check if the number is a perfect number or not");

int n = sc.nextInt();

switch(n)

{

case(1): //to check if the number is a prime number or not

{

System.out.println("Enter the number"); //input

int a = sc.nextInt();

int f = 0;

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

{

if(a%i==0)

{

f++;

}

}

if(f==2)

{

System.out.println("The number is a prime number"); //output

}

else

{

System.out.println("The number is not a prime number"); //output

}

break;

}

case(2): //to check if the number is an automorphic number or not

{

System.out.println("Enter the number"); //input

int a = sc.nextInt();

int sqr = a \* a;

String sa = Integer.toString(a);

String ssqr = Integer.toString(sqr);

if(ssqr.endsWith(sa))

{

System.out.println("The number is an automorphic number"); //output

}

else

{

System.out.println("The number is not an automorphic number"); //output

}

break;

}

case(3): //to check if the number is a perfect number or not

{

System.out.println("Enter the number"); //input

int a = sc.nextInt();

int sum = 0;

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

{

if(a%i==0)

{

sum = sum + i;

}

}

if(sum==a)

{

System.out.println("It is a perfect number"); //output

}

else

{

System.out.println("It is not a perfect number"); //output

}

break;

}

default: //default case if the user enters a wrong choice

{

System.out.println("Wrong Choice"); //output

}

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print("Enter 1 to check if the number is a prime number or not");

("Enter 2 to check if the number is an automorphic number or not");

("Enter 3 to check if the number is a perfect number or not");

Step 3 - Initialize int n to the value input by the user

Step 4 - switch(n)

{

case(1):

{

Print("Enter the number");

int a = sc.nextInt();

int f = 0;

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

{

if(a%i==0)

{

f++;

}

}

if(f==2)

{

Print("The number is a prime number");

}

else

{

Print("The number is not a prime number");

}

break;

}

case(2):

{

Print("Enter the number");

int a = sc.nextInt();

int sqr = a \* a;

String sa = Integer.toString(a);

String ssqr = Integer.toString(sqr);

if(ssqr.endsWith(sa))

{

Print("The number is an automorphic number");

}

else

{

Print("The number is not an automorphic number");

}

break;

}

case(3):

{

Print("Enter the number");

int a = sc.nextInt();

int sum = 0;

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

{

if(a%i==0)

{

sum = sum + i;

}

}

if(sum==a)

{

Print("It is a perfect number");

}

else

{

Print("It is not a perfect number");

}

break;

}

default:

{

Print("Wrong Choice");

}

}

Step 5 - Stop

1. Output -
2. Enter 1 to check if the number is a prime number or not

Enter 2 to check if the number is an automorphic number or not

Enter 3 to check if the number is a perfect number or not

1

Enter the number

19

The number is a prime number

1. Enter 1 to check if the number is a prime number or not

Enter 2 to check if the number is an automorphic number or not

Enter 3 to check if the number is a perfect number or not

2

Enter the number

25

The number is an automorphic number

1. Enter 1 to check if the number is a prime number or not

Enter 2 to check if the number is an automorphic number or not

Enter 3 to check if the number is a perfect number or not

3

Enter the number

6

It is a perfect number

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| n | int | It stores the choice input by the user. |
| a | int | It stores the number input by the user in all the cases. |
| f | int | It stores the number of factors. |
| i | int | It acts as a counter variable in the for loop. |
| sqr | int | It stores the square of the number stored in a. |
| sa | String | It stores the variable a converted to String. |
| ssqr | String | It stores the variable sqr converted to String. |
| sum | int | It stores the sum of the factors of the number stored in variable a . |

Program - 19

1. Question -

Create a class called Points with the following specifications-

    Class name            : Points

Data members

private x1,y1,x2,y2        : variables to store values of the coordinates

Methods

public Points()        : default constructor

public Points (int x1, int y1)    : assigns value to x1 and y1 coordinates

public Points (int x1, int y1, int x2, int y2):  assigns x2 and y2 coordinates directly

                and x1 and y1 coordinates by calling Points(int x1,

                int y1) constructor

public long distance()    : calculates and returns the distance between two

                points using formula

distance =

public int getX1()        : returns the value of x1 coordinate

public void showPoints()    : displays Points in the following format

                Point1 (x1,y1) , Point2 (x2, y2)

Write main() to call the methods.

1. Program -

public class Points

{

private int x1; //data members

private int y1;

private int x2;

private int y2;

public Points() //default constructor

{

}

public Points(int x1, int y1)

{

this.x1 = x1;

this.y1 = y1;

}

public Points(int x1, int y1, int x2, int y2)

{

this.x1=x1;

this.y1 = y1;

this.x2 = x2;

this.y2 = y2;

}

public long distance()

{

long distance = (long) Math.sqrt((Math.pow((x1-x2),2)) + (Math.pow((y1-y2),2))); //distance between the points

return(distance);

}

public int getX1()

{

return(x1);

}

public void showPoints()

{

System.out.println("Point 1(" + x1 + ", " + y1 + "), Point 2(" + x2 + ", " + y2 + ")"); //output

}

public static void main()

{

Points obj1 = new Points();

long a = obj1.distance(); //callings methods with the help of object

int b = obj1.getX1();

obj1.showPoints();

System.out.println("Distance: " +a); //output

System.out.println("X1: " + b);

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Declare private int x1, private int y1, private int x2, private int y2

Step 3 - public Points()

{

}

Step 4 - public Points(int x1, int y1)

{

this.x1 = x1;

this.y1 = y1;

}

Step 5 - public Points(int x1, int y1, int x2, int y2)

{

this.x1=x1;

this.y1 = y1;

this.x2 = x2;

this.y2 = y2;

}

Step 6 - public long distance()

{

long distance = (long) Math.sqrt((Math.pow((x1-x2),2)) + (Math.pow((y1-y2),2)));

return(distance);

}

Step 7 - public int getX1()

{

return(x1);

}

Step 8 - public void showPoints()

{

Print("Point 1(" + x1 + ", " + y1 + "), Point 2(" + x2 + ", " + y2 + ")");

}

Step 9 - public static void main()

{

Points obj1 = new Points();

long a = obj1.distance();

int b = obj1.getX1();

obj1.showPoints();

Print("Distance: " +a);

Print("X1: " + b);

}

Step 10 - Stop

1. Output -

Input - public Points(int x1, int y1, int x2, int y2), x1 = 6, y1 = 4, x2 = 3, y2=0.

long distance()

distance = 5

int getX1()

x1 = 6

public void showPoints()

Point 1(6, 4), Point 2(3, 0)

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| x1 | int | Stores the first x coordinate. |
| y1 | int | Stores the first y coordinate. |
| x2 | int | Stores the second x coordinate. |
| y2 | int | Stores the second y coordinate. |
| distance | long | Stores the distance between the two points. |
| a | long | It stores the distance in the main method. |
| b | int | It stores the value of x1 in the main method. |

Program - 20

1. Question -

Write a program in Java to find the Roman equivalent of any Decimal number entered by the user. [The number entered should be within the Range 1-100]

Brief Note on Roman Numerals: The Roman numerals follow this basic pattern,

100 = C, 90 = XC, 50 = L, 40 = XL, 10 = X, 9 = IX, 5 = V, 4 = IV, 1 = I

1. Program -

import java.util.\*;

public class prog20

{

public static void main()

{

Scanner sc = new Scanner(System.in); //input

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

int n = sc.nextInt();

String str = ""; //stores the roman form

while(n>=100) //while loops

{

str = str + "C";

n = n-100;

}

while(n>=90)

{

str = str + "XC";

n = n-90;

}

while(n>=50)

{

str = str + "L";

n = n-50;

}

while(n>=40)

{

str = str + "XL";

n = n-40;

}

while(n>=10)

{

str = str + "X";

n = n-10;

}

while(n>=9)

{

str = str + "IX";

n = n-9;

}

while(n>=5)

{

str = str + "V";

n = n-5;

}

while(n>=4)

{

str = str + "IV";

n = n-4;

}

while(n>=1)

{

str = str + "I";

n = n-1;

}

System.out.println("Roman Form: " + str); //output

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Print "Enter the number"

Step 3 - Initialize int n to the value input by the user

Step 4 - Initialize String str = "";

Step 5 - while(n>=100) //while loops

{

str = str + "C";

n = n-100;

}

Step 6 - while(n>=90)

{

str = str + "XC";

n = n-90;

}

Step 7 - while(n>=50)

{

str = str + "L";

n = n-50;

}

Step 8 - while(n>=40)

{

str = str + "XL";

n = n-40;

}

Step 9 - while(n>=10)

{

str = str + "X";

n = n-10;

}

Step 10 - while(n>=9)

{

str = str + "IX";

n = n-9;

}

Step 11 - while(n>=5)

{

str = str + "V";

n = n-5;

}

Step 12 - while(n>=4)

{

str = str + "IV";

n = n-4;

}

Step 13 - while(n>=1)

{

str = str + "I";

n = n-1;

}

Step 14 - Print "Roman Form: " + str

Step 15 - Stop

1. Output -

Enter the number

194

Roman Form: CXCIV

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| n | int | It stores the number input by the user. |
| str | String | It stores the roman form the number stored in the variable n. |

Program - 21

1. Question -

Write a program to read a string and convert it to uppercase. Count and

output the consecutive letter pairs that exist in the string.

Eg- IT WAS NOT TOUGH FOR HIM TO RESIDE ABOVE THE HILL

OUTPUT: NOT, TOUGH, HIM, RESIDE, ABOVE, HILL

1. Program -

import java.util.\*;

public class prog21

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the string"); //input

String s = sc.nextLine();

s = s.toUpperCase();

String arr[] = s.split(" ");

System.out.println("The words with consecutive letter pairs are:"); //output

for(int i = 0; i<arr.length; i++) //nested for loop

{

for(int j = 0; j<(arr[i].length())-1; j++)

{

if((arr[i].charAt(j))+1==arr[i].charAt(j+1))

{

System.out.println(arr[i]); //output

}

}

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Initialize String s to the value input by the user

s = s.toUpperCase();

Initialize String arr[] = s.split(" ");

Step 3 - System.out.println("The words with consecutive letter pairs are:");

Step 4 - for(int i = 0; i<arr.length; i++)

{

for(int j = 0; j<(arr[i].length())-1; j++)

{

if((arr[i].charAt(j))+1==arr[i].charAt(j+1))

{

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

}

}

}

Step 5 - Stop

1. Output -

Enter the string

IT WAS NOT TOUGH FOR HIM TO RESIDE ABOVE THE HILL

The words with consecutive letter pairs are:

NOT

TOUGH

HIM

RESIDE

ABOVE

HILL

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| s | String | It stores the string input by the user. |
| arr[] | String | It stores the string input by the user in an array. |
| i | int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |

Program - 22

1. Question -

The English Teacher of a primary school wants her children to identify the

letters used in a paragraph uniquely and arrange them in alphabetical order. Write

a Java program that reads a string from the user and generates a set of unique

letters by removing the duplicates. The program should print the unique set of

letters alphabetically.

Ex: Input- assessment

Output-

Unique letters – asemnt

Alphabetically – aemnst

1. Program -

import java.util.\*;

public class prog22

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the string"); //input

String s = sc.nextLine();

char arr[] = s.toCharArray(); //conversion of string to character array

String s1 = "";

for(int i = 0; i<arr.length; i++) //for loop

{

char ch = arr[i];

if(s1.indexOf(ch)==-1)

{

s1 = s1 + ch;

}

}

System.out.println("Unique letters - " + s1); //output

char arr1[] = s1.toCharArray(); //conversion of string to character array

Arrays.sort(arr1); //sorting the array alphabetically

System.out.print("Alphabetically - "); //output

for(int i = 0; i<arr1.length; i++) //for loop

{

System.out.print(arr1[i]); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Initialize String s to the value input by the user

Initialize char arr[] = s.toCharArray();

Initialize String s1 = "";

Step 3 - for(int i = 0; i<arr.length; i++)

{

char ch = arr[i];

if(s1.indexOf(ch)==-1)

{

s1 = s1 + ch;

}

}

Step 4 - System.out.println("Unique letters - " + s1);

Step 5 - Initialize char arr1[] = s1.toCharArray();

Step 6 - Arrays.sort(arr1);

Step 7 - System.out.print("Alphabetically - ");

Step 8 - for(int i = 0; i<arr1.length; i++)

{

System.out.print(arr1[i]);

}

Step 9 - Stop

1. Output -

Enter the string

racecar

Unique letters - race

Alphabetically - acer

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| s | String | It stores the String input by the user. |
| arr[] | char | It stores the converted form of String s to a character array. |
| s1 | String | It stores the unique letters. |
| i | int | It acts as a counter variable in the for loop. |
| ch | char | It stores the value of arr[i]. |
| arr1[] | char | It stores the converted form of String s1 to a character array. |

Program - 23

1. Question -

Designs functions:

• int factorial(int x): to find and return the factorial of number and stored in

x.

• void IsSpecial(int arr[]): the function gets an array arr[] and by invoking

function factorial() print only special numbers from the array.

Write a main function to input 20 integers in an array and invoke the functions.

(A number is said to be special if sum of factorials of each digit of the number is

equal to that number)

1. Program -

import java.util.\*;

public class prog23

{

Scanner sc = new Scanner(System.in);

int factorial(int x) //int factorial(int x) function

{

int fact = 1;

for(int i = 1; i<=x; i++) //for loop

{

fact = fact \* i;

}

return (fact);

}

void IsSpecial(int arr[]) //void IsSpecial(int arr[]) function

{

System.out.println("Special Numbers: "); //output

for(int i = 0; i<20; i++) //for loop

{

prog23 obj = new prog23();

int s = 0;

int c = arr[i];

while(arr[i]!=0) //while loop

{

int rem = arr[i] % 10;

int f = obj.factorial(rem); //calling int factorial(int x) function

s = s + f;

arr[i] = arr[i] / 10;

}

if(c==s)

{

System.out.println(c); //output

}

}

}

public void main() //main function

{

int arr1[] = new int[20];

System.out.println("Enter the numbers"); //input

for(int i = 0; i<20; i++) //for loop

{

arr1[i] = sc.nextInt();

}

prog23 obj1 = new prog23();

obj1.IsSpecial(arr1); //calling void IsSpecial(int arr[]) function

}

}

1. Algorithm -

Step 1 - Start

Step 2 - int factorial(int x)

{

int fact = 1;

for(int i = 1; i<=x; i++)

{

fact = fact \* i;

}

return (fact);

}

Step 3 - void IsSpecial(int arr[])

{

System.out.println("Special Numbers: ");

for(int i = 0; i<20; i++)

{

prog23 obj = new prog23();

int s = 0;

int c = arr[i];

while(arr[i]!=0)

{

int rem = arr[i] % 10;

int f = obj.factorial(rem);

s = s + f;

arr[i] = arr[i] / 10;

}

if(c==s)

{

System.out.println(c);

}

}

}

Step 4 - public void main()

{

int arr1[] = new int[20];

System.out.println("Enter the numbers");

for(int i = 0; i<20; i++)

{

arr1[i] = sc.nextInt();

}

prog23 obj1 = new prog23();

obj1.IsSpecial(arr1);

}

Step 5 - Stop

1. Output -

Enter the numbers

0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

145

Special Numbers:

0

1

2

145

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| x | int | It stores the value input by the user. |
| fact | int | It stores the factorial of the number. |
| i | int | It acts as a counter variable in the for loop. |
| arr[] | int | It stores the array input by the user. |
| s | int | It stores the sum of the factorials. |
| c | int | It stores the value of arr[i]. |
| rem | int | It stores the value of arr[i] % 10. |
| f | int | It stores the factorial of rem. |
| arr1[] | int | It stores the array input by the user. |

Program - 24

1. Question -

The editor of a news paper company needs to edit write-ups sent by the

correspondents frequently. Help her to device a method in Java that reads a string

and a subpart1 from the user and search if the subpart1 is found in the main string.

It reads another string, subpart2 which will replace subpart1 in the main string.

The method should display a suitable message if the subpart1 is not found in the

main string.

Eg. Input – mimicry is a special mistery art in mindful people

Subpart 1: mi

Subpart 2: mo

Output: momocry is a special mostery art in mondful people

1. Program -

import java.util.\*;

public class prog24

{

public static void main()

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter the main string"); //input

String s = sc.nextLine();

System.out.println("Enter the subpart 1");

String sub1 = sc.nextLine();

System.out.println("Enter the subpart 2");

String sub2 = sc.nextLine();

String snew = s.replace(sub1, sub2); //replacing subpart 1 with subpart 2

System.out.println(snew); //output

}

}

1. Algorithm -

Step 1 - Start

Start 2 - Scanner sc = new Scanner(System.in);

System.out.println("Enter the main string");

String s = sc.nextLine();

System.out.println("Enter the subpart 1");

String sub1 = sc.nextLine();

System.out.println("Enter the subpart 2");

String sub2 = sc.nextLine();

String snew = s.replace(sub1, sub2);

System.out.println(snew);

Step 3 - Stop

1. Output -

Enter the main string

The boy owns a bow

Enter the subpart 1

bo

Enter the subpart 2

co

The coy owns a cow

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| s | String | It stores the main string input by the user. |
| sub1 | String | It stores the subpart 1 input by the user. |
| sub2 | String | It stores the subpart 2 input by the user. |
| snew | String | It stores the new string. |

Program - 25

1. Question -

Write a program to identify common elements or numbers between two given

arrays.

Example:

int[] arr1 = {4,7,3,9,2};

int[] arr2 = {3,2,12,9,40,32,4};

Output:

4

3

9

2

1. Program -

public class prog25

{

public static void main(int arr[], int arr1[]) //input through parameters

{

for(int i = 0; i<arr.length; i++) //nested for loop

{

for(int j =0; j<arr1.length; j++)

{

if(arr[i]==arr1[j])

{

System.out.println(arr[i]); //output

}

}

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - public static void main(int arr[], int arr1[])

{

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

{

for(int j =0; j<arr1.length; j++)

{

if(arr[i]==arr1[j])

{

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

}

}

}

}

Step 3 - Stop

1. Output -

Input - arr[] = {2,4,6,8,10,12,14,16,18,20}

arr1[] = {3,6,9,12,15,18,21,24,27,30}

6

12

18

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| arr[] | int | It stores the first array input by the user. |
| arr1[] | int | It stores the second array input by the user. |
| i | int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |

Program - 26

1. Question -

Write a program in Java to store the numbers in a 3\*4 matrix in a DDA. Find

the sum of all the numbers of the matrix and display the sum using an input

statement.

Sample Output:

The numbers of the matrix are:

1 2 3 4

5 6 7 8

9 10 11 12

The sum of the elements in the matrix is 78.

1. Program -

import java.util.\*;

public class prog26

{

public static void main()

{

int arr[][] = new int [3][4];

System.out.println("Enter the numbers"); //input

Scanner sc = new Scanner(System.in);

for(int i = 0; i<3; i++) //nested for loop

{

for(int j = 0; j<4; j++)

{

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

}

}

int sum = 0;

for(int i = 0; i<3; i++) //nested for loop

{

for(int j = 0; j<4; j++)

{

sum = sum + arr[i][j];

}

}

System.out.println("The numbers of the matrix are:"); //output

for(int i = 0; i<3; i++) //nested for loop

{

for(int j = 0; j<4; j++)

{

System.out.print(arr[i][j] + " "); //output

}

System.out.println();

}

System.out.println("The sum of the elements in the matrix is " + sum); //output

}

}

1. Algorithm -

Step 1 - Start

Step 2 - int arr[][] = new int [3][4];

System.out.println("Enter the numbers");

Scanner sc = new Scanner(System.in);

Step 3 - for(int i = 0; i<3; i++)

{

for(int j = 0; j<4; j++)

{

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

}

}

Step 4 - Initialize int sum = 0

Step 5 - for(int i = 0; i<3; i++)

{

for(int j = 0; j<4; j++)

{

sum = sum + arr[i][j];

}

}

Step 6 - System.out.println("The numbers of the matrix are:");

Step 7 - for(int i = 0; i<3; i++)

{

for(int j = 0; j<4; j++)

{

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

}

System.out.println();

}

Step 8 - System.out.println("The sum of the elements in the matrix is " + sum);

Step 9 - Stop

1. Output -

Enter the numbers

11

12

13

14

15

16

17

18

19

20

21

22

The numbers of the matrix are:

11 12 13 14

15 16 17 18

19 20 21 22

The sum of the elements in the matrix is 198

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| arr[][] | int | It stores the array input by the user. |
| i | int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |
| sum | int | It stores the sum of the numbers of the matrix. |

Program - 27

1. Question -

Design a class to overload print() as follows:

void print(int x, int y) - prints odd number between x and y

void print(char ch1, char ch2) - prints all the consonants between ch1 and ch2

void print(char ch, int n) -to print the character ch for n number of times in the

given

format. Ex- ch=&amp; and n =4

& & & &

& & &

& &

&

1. Program -

import java.util.\*;

public class prog27

{

void print(int x, int y) //void print(int x, int y) function

{

System.out.println("The odd numbers between " + x + " and " + y + " are: "); //output

for(int i = x; i<=y; i++) //for loop

{

if(i%2==1)

{

System.out.println(i); //output

}

}

}

void print(char ch1, char ch2) //void print(char ch1, char ch2) function

{

System.out.println("The consonants between " + ch1 + " and " + ch2 + " are: "); //output

for(char i = ch1; i<=ch2; i++) //for loop

{

if((i!='a')&&(i!='A')&&(i!='e')&&(i!='E')&&(i!='i')&&(i!='I')&&(i!='o')&&(i!='O')&&(i!='u')&&(i!='U'))

{

System.out.println(i); //output

}

}

}

void print(char ch, int n) //void print(char ch, int n) function

{

for(int i = 1; i<=n; i++) //nested for loop

{

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

{

System.out.print(ch + " "); //output

}

System.out.println();

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - void print(int x, int y)

{

System.out.println("The odd numbers between " + x + " and " + y + " are: ");

for(int i = x; i<=y; i++)

{

if(i%2==1)

{

System.out.println(i);

}

}

}

Step 3 - void print(char ch1, char ch2)

{

System.out.println("The consonants between " + ch1 + " and " + ch2 + " are: ");

for(char i = ch1; i<=ch2; i++)

{

if((i!='a')&&(i!='A')&&(i!='e')&&(i!='E')&&(i!='i')&&(i!='I')&&(i!='o')&&(i!='O')&&(i!='u')&&(i!='U'))

{

System.out.println(i);

}

}

}

Step 4 - void print(char ch, int n)

{

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

{

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

{

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

}

System.out.println();

}

}

Step 5 - Stop

1. Output -

void print(int x, int y), x = 1, y = 10

The odd numbers between 1 and 10 are:

1

3

5

7

9

void print(char ch1, char ch2), ch1 = ‘a’, ch2 = ‘b’

The consonants between a and z are:

b

c

d

f

g

h

j

k

l

m

n

p

q

r

s

t

v

w

x

y

z

void print(char ch, int n), ch = ‘&’, n = 6

& & & & & &

& & & & &

& & & &

& & &

& &

&

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| x | int | It stores the value input by the user. |
| y | int | It stores the value input by the user. |
| i | int | It acts as a counter variable in the for loop. |
| ch1 | char | It stores the character input by the user. |
| ch2 | char | It stores the character input by the user. |
| i | char | It stores the character to be printed. |
| ch | char | It stores the character input by the user. |
| n | int | It stores the value input by the user. |

Program - 28

1. Question -

Write a program in Java to create a 3\*3 Square Matrix and store numbers in

it. The programmer should check whether the Matrix is Symmetric or not. A

Square Matrix is said to be Symmetric if the element of the ith row and jth column

is equal to the element of the jth row an ith column. E.g., A Symmetric Matrix:

1 2 3

2 4 5

3 5 6

1. Program -

import java.util.\*;

public class prog28

{

public static void main()

{

int arr[][] = new int[3][3];

System.out.println("Enter the numbers"); //input

Scanner sc = new Scanner(System.in);

for(int i = 0; i<3; i++) //nested for loop

{

for(int j = 0; j<3; j++)

{

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

}

}

if((arr[0][1]==arr[1][0])&&(arr[0][2]==arr[2][0])&&(arr[1][2]==arr[2][1]))

{

System.out.println("The matrix is symmetrical"); //output

}

else

{

System.out.println("The matrix is not symmetrical"); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - int arr[][] = new int[3][3];

System.out.println("Enter the numbers");

Scanner sc = new Scanner(System.in);

Step 3 - for(int i = 0; i<3; i++)

{

for(int j = 0; j<3; j++)

{

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

}

}

Step 4 - if((arr[0][1]==arr[1][0])&&(arr[0][2]==arr[2][0])&&(arr[1][2]==arr[2][1]))

{

System.out.println("The matrix is symmetrical"); //output

}

else

{

System.out.println("The matrix is not symmetrical"); //output

}

Step 5 - Stop

1. Output -

Enter the numbers

1

2

3

2

4

5

3

5

6

The matrix is symmetrical

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| arr[][] | int | It stores the array input by the user. |
| i | Int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |

Program - 29

1. Question -

WAP to transpose a matrix of 3\*3 accepted by user. The transpose of a

matrix is a new matrix whose rows are the columns of the original. (This

makes the columns of the new matrix the rows of the original). Here is a

matrix and its transpose: A new matrix whose rows are the columns of the original matrix is called Transpose of a matrix.

1. Program -

import java.util.\*;

public class prog29

{

public static void main()

{

int arr[][] = new int[3][3];

System.out.println("Enter the numbers"); //input

Scanner sc = new Scanner(System.in);

for(int i = 0; i<3; i++) //nested for loop

{

for(int j = 0; j<3; j++)

{

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

}

}

System.out.println("Matrix:"); //output

for(int i = 0; i<3; i++) //nested for loop

{

for(int j = 0; j<3; j++)

{

System.out.print(arr[i][j] + " "); //output

}

System.out.println();

}

int a;

int b;

int c;

a = arr[0][1]; //transposing the matrix

arr[0][1] = arr[1][0];

arr[1][0] = a;

b = arr[0][2];

arr[0][2] = arr[2][0];

arr[2][0] = b;

c = arr[1][2];

arr[1][2] = arr[2][1];

arr[2][1] = c;

System.out.println("New Transposed Matrix:"); //output

for(int i = 0; i<3; i++)

{

for(int j = 0; j<3; j++)

{

System.out.print(arr[i][j] + " "); //output

}

System.out.println();

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - int arr[][] = new int[3][3];

System.out.println("Enter the numbers");

Scanner sc = new Scanner(System.in);

Step 3 - for(int i = 0; i<3; i++)

{

for(int j = 0; j<3; j++)

{

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

}

}

Step 4 - System.out.println("Matrix:");

Step 5 - for(int i = 0; i<3; i++)

{

for(int j = 0; j<3; j++)

{

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

}

System.out.println();

}

Step 6 - Declare int a

Declare int b

Declare int c

Step 7 - a = arr[0][1];

arr[0][1] = arr[1][0];

arr[1][0] = a;

b = arr[0][2];

arr[0][2] = arr[2][0];

arr[2][0] = b;

c = arr[1][2];

arr[1][2] = arr[2][1];

arr[2][1] = c;

Step 8 - System.out.println("New Transposed Matrix:");

Step 9 - for(int i = 0; i<3; i++)

{

for(int j = 0; j<3; j++)

{

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

}

System.out.println();

}

Step 10 - Stop

1. Output -

Enter the numbers

1

2

3

4

5

6

7

8

9

Matrix:

1 2 3

4 5 6

7 8 9

New Transposed Matrix:

1 4 7

2 5 8

3 6 9

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| arr[][] | int | It stores the array input by the user. |
| i | int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |
| a | int | It is used to transpose the matrix. |
| b | int | It is used to transpose the matrix. |
| c | int | It is used to transpose the matrix. |

Program - 30

1. Question -

Write a program to store the numbers in a 4\*4 matrix in a DDA. Find the

highest and lowest numbers of the matrix by using an input statement. Sample

Output:

The numbers of the matrix are:

4 9 6 2

12 48 57 25

75 61 2 84

18 97 54 62

The lowest number in the array is 2

the highest number in the array is 97

1. Program -

import java.util.\*;

public class prog30

{

public static void main()

{

Scanner sc = new Scanner(System.in);

int arr[][] = new int[4][4];

System.out.println("Enter the numbers"); //input

for(int i = 0; i<4; i++) //nested for loop

{

for(int j = 0; j<4; j++)

{

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

}

}

int max = arr[0][0];

int min = arr[0][0];

System.out.println("The numbers of the matrix are:"); //output

for(int i = 0; i<4; i++) //nested for loop

{

for(int j = 0; j<4; j++)

{

System.out.print(arr[i][j] + " "); //output

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

{

max = arr[i][j];

}

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

{

min = arr[i][j];

}

}

System.out.println();

}

System.out.println("The lowest number in the array is " + min); //output

System.out.println("The highest number in the array is " + max);

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Scanner sc = new Scanner(System.in);

int arr[][] = new int[4][4];

System.out.println("Enter the numbers");

Step 3 - for(int i = 0; i<4; i++)

{

for(int j = 0; j<4; j++)

{

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

}

}

Step 4 - int max = arr[0][0];

int min = arr[0][0];

System.out.println("The numbers of the matrix are:");

Step 5 - for(int i = 0; i<4; i++)

{

for(int j = 0; j<4; j++)

{

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

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

{

max = arr[i][j];

}

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

{

min = arr[i][j];

}

}

System.out.println();

}

Step 6 - System.out.println("The lowest number in the array is " + min);

System.out.println("The highest number in the array is " + max);

Step 7 - Stop

1. Output -

Enter the numbers

21

62

73

45

89

95

82

31

26

18

22

28

66

64

31

20

The numbers of the matrix are:

21 62 73 45

89 95 82 31

26 18 22 28

66 64 31 20

The lowest number in the array is 18

The highest number in the array is 95

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| arr[][] | int | It stores the array input by the user. |
| i | int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |
| max | int | It stores the maximum number in the array. |
| min | int | It stores the minimum number in the array. |

Program - 31

1. Question -

Write a program to accept the year of graduation from school as an integer value from the user. Using the binary search technique on the sorted array of integers given below, output the message “Record exists” if the value input is located in the array. If not, output the message “Record does not exist”

Sample Input:

n[0] n[1] n[2] n[3] n[4] n[5] n[6] n[7] n[8] n[9]

1982 1987 1993 1996 1999 2003 2006 2007 2009 2010

1. Program -

import java.util.\*;

public class prog31

{

public static void main()

{

Scanner sc= new Scanner(System.in);

int arr[] = {1982,1987,1993,1996,1999,2003,2006,2007,2009,2010}; //array containing the years

System.out.println("Enter the year of graduation"); //input

int g = sc.nextInt();

int f = 0;

int l = 0;

int u = 9;

int m = 0;

while(l<=u) //while loop

{

m = (l+u)/2;

if(g>arr[m])

{

l = m+1;

}

else if(g<arr[m])

{

u = m-1;

}

else

{

f = 1;

break;

}

}

if(f==1)

{

System.out.println("Record exists"); //output

}

else

{

System.out.println("Record does not exist"); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Scanner sc= new Scanner(System.in);

int arr[] = {1982,1987,1993,1996,1999,2003,2006,2007,2009,2010};

System.out.println("Enter the year of graduation");

int g = sc.nextInt();

Step 3 - Declare int f = 0;

Declare int l = 0;

Declare int u = 9;

Declare int m = 0;

Step 4 - while(l<=u)

{

m = (l+u)/2;

if(g>arr[m])

{

l = m+1;

}

else if(g<arr[m])

{

u = m-1;

}

else

{

f = 1;

break;

}

}

Step 5 - if(f==1)

{

System.out.println("Record exists");

}

else

{

System.out.println("Record does not exist");

}

Step 6 - Stop

1. Output -

Enter the year of graduation

2006

Record exists

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| arr[] | int | It stores the array containing the years. |
| g | int | It stores the value input by the user. |
| f | int | It acts as a flag which tells us whether the record exists or not |
| l | int | It stores the lower limit. |
| u | int | It stores the upper limit. |
| m | int | It stores the middle position. |

Program - 32

1. Question -

Write a program to accept a list of 20 integers. Sort the first 10 numbers in

ascending order and next the 10 numbers in descending order by using ‘Bubble Sort’ technique. Finally, print the complete list of integers.

1. Program -

import java.util.\*;

public class prog32

{

public static void main()

{

Scanner sc = new Scanner(System.in);

int arr[] = new int[20];

int temp;

int temp1;

System.out.println("Enter the numbers"); //input

for(int i = 0; i<20; i++) //for loop

{

arr[i] = sc.nextInt();

}

for(int i=0;i<9;i++) //nested for loop ascending order

{

for(int j=i+1;j<10;j++)

{

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

{

temp= arr[i];

arr[i]= arr[j];

arr[j]=temp;

}

}

}

for(int i=10;i<19;i++) //nested for loop descending order

{

for(int j=i+1;j<20;j++)

{

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

{

temp1= arr[i];

arr[i]= arr[j];

arr[j]=temp1;

}

}

}

System.out.println("New Arranged Array:"); //output

for(int i = 0; i<20; i++) //for loop

{

System.out.println(arr[i]); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Scanner sc = new Scanner(System.in);

int arr[] = new int[20];

Declare int temp;

Declare int temp1;

System.out.println("Enter the numbers");

Step 3 - for(int i = 0; i<20; i++)

{

arr[i] = sc.nextInt();

}

Step 4 - for(int i=0;i<9;i++)

{

for(int j=i+1;j<10;j++)

{

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

{

temp= arr[i];

arr[i]= arr[j];

arr[j]=temp;

}

}

}

Step 5 - for(int i=10;i<19;i++)

{

for(int j=i+1;j<20;j++)

{

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

{

temp1= arr[i];

arr[i]= arr[j];

arr[j]=temp1;

}

}

}

Step 6 - System.out.println("New Arranged Array:");

Step 7 - for(int i = 0; i<20; i++)

{

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

}

Step 8 - Stop

1. Output -

Enter the numbers

3

7

4

9

2

10

1

5

8

6

14

19

11

20

13

16

12

15

17

18

New Arranged Array:

1

2

3

4

5

6

7

8

9

10

20

19

18

17

16

15

14

13

12

11

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| arr[] | int | It is used to store the 20 numbers input by the user. |
| temp | int | It is used to swap the values of arr[i] and arr[j] in the ascending order for loop. |
| temp1 | int | It is used to swap the values of arr[i] and arr[j] in the descending order for loop. |
| i | int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |

Program - 33

1. Question -

Write a program in JAVA that reads the following list of countries and their respective cities into two separate one-dimensional arrays. The program should accept the name of a country as input and give the name of the corresponding city as an output. The program should be designed to give an error message where a city is asked for a country whose name is not given in the list. To stop the program, “XXX” is to be entered as input.

GERMANY BERLIN

NEPAL KATMANDU

JAPAN TOKYO

CANADA MONTREAL

IRAQ BAGHDAD

1. Program -

import java.util.\*;

public class prog33

{

public static void main()

{

Scanner sc = new Scanner(System.in);

String country[] = {"GERMANY","NEPAL","JAPAN","CANADA","IRAQ"}; //array containing the name of the countries

String city[] = {"BERLIN","KATMANDU","TOKYO","MONTREAL","BAGHDAD"}; //array containing the name of the cities

System.out.println("Enter the name of the country"); //input

System.out.println("Enter XXX to stop the program");

String s = sc.nextLine();

int a = 0;

if(s.equals("XXX"))

{

System.out.println("The program has stopped"); //output

System.exit(0); //terminates the program

}

else

{

for(int i = 0; i<5; i++) //for loop

{

if(s.equals(country[i]))

{

a = i;

break;

}

}

}

if(a>=0)

{

System.out.println("City: " + city[a]); //output

}

else

{

System.out.println("Wrong Choice"); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Scanner sc = new Scanner(System.in);

String country[] = {"GERMANY","NEPAL","JAPAN","CANADA","IRAQ"};

String city[] = {"BERLIN","KATMANDU","TOKYO","MONTREAL","BAGHDAD"};

System.out.println("Enter the name of the country");

System.out.println("Enter XXX to stop the program");

String s = sc.nextLine();

Initialize int a = 0;

Step 3 - if(s.equals("XXX"))

{

System.out.println("The program has stopped");

System.exit(0);

}

else

{

for(int i = 0; i<5; i++)

{

if(s.equals(country[i]))

{

a = i;

break;

}

}

}

Step 4 - if(a>=0)

{

System.out.println("City: " + city[a]);

}

else

{

System.out.println("Wrong Choice");

}

Step 5 - Stop

1. Output -

Input - JAPAN

Enter the name of the country

Enter XXX to stop the program

JAPAN

City: TOKYO

Input - XXX

Enter the name of the country

Enter XXX to stop the program

XXX

The program has stopped

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| country[] | String | It stores the array containing the name of the countries. |
| city[] | String | It stores the array containing the name of the cities. |
| s | String | It stores the String input by the user. |
| a | int | It stores the index number of the country in the array country[] entered by the user, and is used to find its city in the array city[]. |

Program - 34

1. Question -

Write a JAVA program to accept the temperature of any 10 cities in degrees Fahrenheit. Convert temperature to degree centigrade using the given formula: Centigrade = (Fahrenheit - 32) X 5/9 Display the information in the given format. Also at the end print the total number of cities where the temperature is more then 35 degree centigrade and the city name with maximum temperature.

City Name Fahrenheit Temperature Centigrade Temperature

--------- ---------------------- ----------------------

Number of cities more then 35-degree centigrade temperature:

Name of the city with maximum temperature:

1. Program -

import java.util.\*;

public class prog34

{

public static void main()

{

Scanner sc = new Scanner(System.in);

double f[] = new double[10];

double c[] = new double[10];

String n[] = new String[10];

int count = 0;

String max = "";

double maxtemp = 0.0;

System.out.println("Enter the name of the cities"); //input

for(int i = 0; i<10; i++) //for loop

{

n[i] = sc.nextLine();

}

System.out.println("Enter the temperatures in fahrenheit"); //input

for(int i = 0; i<10; i++) //for loop

{

f[i] = sc.nextDouble();

c[i] = (f[i] - 32) \* 5/9;

if(c[i]>35)

{

count++;

}

if(c[i]>maxtemp)

{

max = n[i];

maxtemp = c[i];

}

}

System.out.println("City Name \t Fahrenheit Temperature \t Centigrade Temperature"); //output

for(int i = 0; i<10; i++)

{

System.out.println(n[i] + "\t\t\t" + f[i] + "\t\t\t" + c[i]); //output

}

System.out.println(); //output

System.out.println("Number of cities with more than 35 degree centigrade temperature: " + count);

System.out.println("Name of the city with maximum temperature: " + max);

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Scanner sc = new Scanner(System.in);

double f[] = new double[10];

double c[] = new double[10];

String n[] = new String[10];

Initialize int count = 0;

Initialize String max = "";

Initialize double maxtemp = 0.0;

System.out.println("Enter the name of the cities");

Step 3 - for(int i = 0; i<10; i++)

{

n[i] = sc.nextLine();

}

Step 4 - System.out.println("Enter the temperatures in fahrenheit");

Step 5 - for(int i = 0; i<10; i++)

{

f[i] = sc.nextDouble();

c[i] = (f[i] - 32) \* 5/9;

if(c[i]>35)

{

count++;

}

if(c[i]>maxtemp)

{

max = n[i];

maxtemp = c[i];

}

}

Step 6 - System.out.println("City Name \t Fahrenheit Temperature \t Centigrade Temperature");

Step 7 - for(int i = 0; i<10; i++)

{

System.out.println(n[i] + "\t\t\t" + f[i] + "\t\t\t" + c[i]);

}

Step 8 - System.out.println();

System.out.println("Number of cities with more than 35 degree centigrade temperature: " + count);

System.out.println("Name of the city with maximum temperature: " + max);

Step 9 - Stop

1. Output -

Enter the name of the cities

Chennai

Delhi

Kolkata

Kochi

Dehradun

Agra

Shimla

Mumbai

Hyderabad

Bengaluru

Enter the temperatures in fahrenheit

105

91

86

71

75

79

88

81

99

94

City Name Fahrenheit Temperature Centigrade Temperature

Chennai 105.0 40.55555555555556

Delhi 91.0 32.77777777777778

Kolkata 86.0 30.0

Kochi 71.0 21.666666666666668

Dehradun 75.0 23.88888888888889

Agra 79.0 26.11111111111111

Shimla 88.0 31.11111111111111

Mumbai 81.0 27.22222222222222

Hyderabad 99.0 37.22222222222222

Bengaluru 94.0 34.44444444444444

Number of cities with more than 35 degree centigrade temperature: 2

Name of the city with maximum temperature: Chennai

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| f[] | double | It stores the temperatures in fahrenheit input by the user. |
| c[] | double | It stores the temperatures in celsius. |
| n[] | String | It stores the name of the cities input by the user. |
| count | int | It store the number of cities with temperature more than 35 degree centigrade. |
| max | String | It stores the name of the city with the maximum temperature. |
| maxtemp | double | It stores the maximum temperature in celsius. |
| i | int | It acts as a counter variable in the for loop. |

Program - 35

1. Question -

Write a program in JAVA to accept the name and marks in computer science of forty students in an array and then print the name and marks of students according to their merit (decreasing order of marks).

1. Program -

import java.util.\*;

public class prog35

{

public static void main()

{

Scanner sc = new Scanner(System.in);

String n[] = new String[40];

String n1[] = new String[40];

int m[] = new int[40];

int m1[] = new int[40];

int temp;

System.out.println("Enter the name of the students"); //input

for(int i = 0; i<40; i++) //for loop

{

n[i] = sc.nextLine();

}

System.out.println("Enter the marks of the students in computer science"); //input

for(int i = 0; i<40; i++) //for loop

{

m[i] = sc.nextInt();

}

for(int i = 0; i<40; i++) //for loop

{

m1[i] = m[i];

}

for(int i=0;i<39;i++) //nested for loop

{

for(int j=i+1;j<40;j++)

{

if(m[i]<(m[j]))

{

temp= m[i];

m[i]= m[j];

m[j]=temp;

}

}

}

for(int i = 0; i<40; i++) //nested for loop

{

for(int j = 0; j<40; j++)

{

if(m1[j]==m[i])

{

n1[i] = n[j];

}

}

}

System.out.println("Name\tMarks"); //output

for(int i = 0; i<40; i++) //for loop

{

System.out.println(n1[i] + "\t" + m[i]); //output

}

}

}

1. Algorithm -

Step 1 - Start

Step 2 - Scanner sc = new Scanner(System.in);

String n[] = new String[40];

String n1[] = new String[40];

int m[] = new int[40];

int m1[] = new int[40];

int temp;

System.out.println("Enter the name of the students");

Step 3 - for(int i = 0; i<40; i++)

{

n[i] = sc.nextLine();

}

Step 4 - System.out.println("Enter the marks of the students in computer science");

Step 5 - for(int i = 0; i<40; i++)

{

m[i] = sc.nextInt();

}

Step 6 - for(int i = 0; i<40; i++)

{

m1[i] = m[i];

}

Step 7 - for(int i=0;i<39;i++)

{

for(int j=i+1;j<40;j++)

{

if(m[i]<(m[j]))

{

temp= m[i];

m[i]= m[j];

m[j]=temp;

}

}

}

Step 8 - for(int i = 0; i<40; i++)

{

for(int j = 0; j<40; j++)

{

if(m1[j]==m[i])

{

n1[i] = n[j];

}

}

}

Step 9 - System.out.println("Name\tMarks");

Step 10 - for(int i = 0; i<40; i++)

{

System.out.println(n1[i] + "\t" + m[i]);

}

Step 11 - Stop

1. Output -

Enter the name of the students

a

b

c

d

e

f

g

h

i

j

k

l

m

n

o

p

q

r

s

t

u

v

w

x

y

z

aa

ab

ac

ad

ae

af

ag

ah

ai

aj

ak

al

am

an

Enter the marks of the students in computer science

74

47

62

65

12

67

18

21

4

59

70

45

54

23

43

86

58

77

69

84

26

19

31

42

99

16

76

34

83

85

5

89

37

90

97

11

14

6

25

60

Name Marks

y 99

ai 97

ah 90

af 89

p 86

ad 85

t 84

ac 83

r 77

aa 76

a 74

k 70

s 69

f 67

d 65

c 62

an 60

j 59

q 58

m 54

b 47

l 45

o 43

x 42

ag 37

ab 34

w 31

u 26

am 25

n 23

h 21

v 19

g 18

z 16

ak 14

e 12

aj 11

al 6

ae 5

i 4

1. VDT -

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Description |
| n[] | String | It stores the names of the students input by the user. |
| n1[] | String | It stores the new array of the names of the students arranged according to their marks. |
| m[] | int | It stores the marks input by the user. |
| m1[] | int | It stores a copy of the marks stored in m[], a copy is made so that it can be used later for reference and is used to arrange the names in n1[] when the values in m[] are arranged in descending order. |
| i | int | It acts as a counter variable in the for loop. |
| j | int | It acts as a counter variable in the for loop. |
| temp | int | It is used to swap the values stored in m[i] and m[j]. |