1a)class DefaultValues {

byte defaultByte;

short defaultShort;

int defaultInt;

long defaultLong;

float defaultFloat;

double defaultDouble;

char defaultChar;

boolean defaultBoolean;

public static void main(String[] args) {

DefaultValues obj = new DefaultValues();

System.out.println("Default values of primitive data types:");

System.out.println("byte: " + obj.defaultByte);

System.out.println("short: " + obj.defaultShort);

System.out.println("int: " + obj.defaultInt);

System.out.println("long: " + obj.defaultLong);

System.out.println("float: " + obj.defaultFloat);

System.out.println("double: " + obj.defaultDouble);

System.out.println("char: [" + obj.defaultChar + "]");

System.out.println("boolean: " + obj.defaultBoolean);

}

}

output:

Default values of primitive data types:

byte: 0

short: 0

int: 0

long: 0

float: 0.0

double: 0.0

char: []

boolean: false

1b)import java.util.Scanner;

public class QuadraticEquation {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter coefficient a: ");

double a = scanner.nextDouble();

System.out.print("Enter coefficient b: ");

double b = scanner.nextDouble();

System.out.print("Enter coefficient c: ");

double c = scanner.nextDouble();

double D = b \* b - 4 \* a \* c;

System.out.println("Discriminant (D) = " + D);

if (D > 0) {

System.out.println("The equation has two real and distinct roots.");

double root1 = (-b + sqrt(D)) / (2 \* a);

double root2 = (-b - sqrt(D)) / (2 \* a);

System.out.println("Root 1: " + root1);

System.out.println("Root 2: " + root2);

} else if (D == 0) {

System.out.println("The equation has one real and repeated root.");

double root = -b / (2 \* a);

System.out.println("Root: " + root);

} else {

System.out.println("The equation has two complex roots.");

double realPart = -b / (2 \* a);

double imaginaryPart = sqrt(-D) / (2 \* a);

System.out.println("Root 1: " + realPart + " + " + imaginaryPart + "i");

System.out.println("Root 2: " + realPart + " - " + imaginaryPart + "i");

}

scanner.close();

}

public static double sqrt(double number) {

double t;

double squareRoot = number / 2;

do {

t = squareRoot;

squareRoot = (t + (number / t)) / 2;

} while ((t - squareRoot) != 0);

return squareRoot;

}

}

output:

Enter coefficient a: 1

Enter coefficient b: -3

Enter coefficient c: 2

Discriminant (D) = 1.0

The equation has two real and distinct roots.

Root 1: 2.0

Root 2: 1.0

2a)public class SimpleClass{

public void PrintMessage(){

System.out.println("Hello World");

}

public static void main(String[] args){

SimpleClass obj=new SimpleClass();

obj.PrintMessage();

}}

output:

Hello World

2b)public class MethodOverWriting{

public int add(int a,int b)

{

return a+b;

}

public int add(int a,int b,int c){

return a+b+c;

}

public double add(double a,double b){

return a+b;

}

public static void main(String[] args)

{

MethodOverWriting obj=new MethodOverWriting();

System.out.println("sum of 5 and 10:"+obj.add(5,10));

System.out.println("sum of 2,3 and 7:"+obj.add(2,3,7));

System.out.println("sum of 10.5 and 6.5:"+obj.add(10.5,6.5));

}}

output:

sum of 5 and 10:15

sum of 2,3 and 7:12

sum of 10.5 and 6.5:17.0

2c)class ConExample{

int i;

int j;

ConExample(){

i=34;

j=23;

}

public void Display(){

System.out.println("i value:"+i+"and j value:"+j);

}

public static void main(String[] args){

ConExample obj=new ConExample();

obj.Display();

}}

output:

i value:34 and j value:23

3a)class Demo11

{

Demo11()

{

System.out.println("Empty constructor");

}

Demo11(int a)

{

System.out.println("a value is:"+a);

}

Demo11(int a,int b)

{

System.out.println("a value is:"+a+"b value is:"+b);

}

}

class ConstructorOverloading

{

public static void main (String args[])

{

Demo11 ob1=new Demo11(28,4);

}

}

output:

a value is:28b value is:4

3b)import java.util.Arrays;

import java.util.Scanner;

public class BinarySearch {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input: List size and elements

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

int n = scanner.nextInt();

int[] arr = new int[n];

System.out.println("Enter " + n + " elements (space-separated): ");

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

arr[i] = scanner.nextInt();

}

// Sorting the array

Arrays.sort(arr);

System.out.println("Sorted array: " + Arrays.toString(arr));

// Input: Element to search

System.out.print("Enter the element to search: ");

int key = scanner.nextInt();

// Perform binary search

int result = binarySearch(arr, key);

// Output result

if (result == -1) {

System.out.println("Element not found in the list.");

} else {

System.out.println("Element found at index: " + result);

}

scanner.close();

}

// Binary search method

public static int binarySearch(int[] arr, int key) {

int left = 0, right = arr.length - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

// Check if key is at mid

if (arr[mid] == key) {

return mid;

}

// If key is smaller than mid, search left subarray

if (key < arr[mid]) {

right = mid - 1;

} else { // Search right subarray

left = mid + 1;

}

}

// Key not found

return -1;

}

}

output:

Enter the number of elements: 5

Enter 5 elements (space-separated):

23

55

22

2

74

Sorted array: [2, 22, 23, 55, 74]

Enter the element to search: 22

Element found at index: 1

3c)import java.util.Scanner;

class BubbleSort

{

public static void main(String args[])

{

Scanner scanner=new Scanner(System.in);

System.out.println("enter the no.of elements into array:");

int n=scanner.nextInt();

int[] arr=new int[n];

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

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

{

arr[i]=scanner.nextInt();

}

for(int i=0;i<n-1;i++)

{

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

{

if(arr[j]>arr[j+1])

{

int temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

}

System.out.println("sorted array:");

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

{

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

}

}

}

output:

enter the no.of elements into array:

5

enter the elements:

21

32

25

6

41

sorted array:

6

21

25

32