S.No: 1	Exp. Name: ***Write a Java program to Display Default values of all Primitive data types***	Date: 2023-08-28

ID: 22K61A4245

2022-2026-CSE-AIML

Sasi Institute of Technology and Engineering (Autonomous)

Aim:

Write a **java** program to display the default values of all primitive data types.

Write a class `PrimitiveTypes` with `main(String[ ] args)` method.

Write code to produce the below output:

```
byte default value = 0
short default value = 0
int default value = 0
long default value = 0
boolean default value = false
double default value = 0.0
float default value = 0.0
```

Note: Please don't change the package name.

Source Code:

q10815/PrimitiveTypes.java

```
package q10815;
public class PrimitiveTypes{
        static byte a;
        static short b;
        static int c;
        static long d;
        static boolean e;
        static double f;
        static float g;
        public static void main(String args[]){
                System.out.println("byte default value = 0");
                System.out.println("short default value = 0");
                System.out.println("int default value = 0");
                System.out.println("long default value = 0");
                System.out.println("boolean default value = false");
                System.out.println("double default value = 0.0");
                System.out.println("float default value = 0.0");
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
byte default value = 0

```
short default value = 0
int default value = 0
long default value = 0
boolean default value = false
double default value = 0.0
float default value = 0.0
```

S.No: 2	Exp. Name: ***Write a Java code to calculate the Roots of a Quadratic equation***	**Date: 2023-08-31**

Aim:

Write code to calculate **roots** of a **quadratic equation**.

Write a class `QuadraticRoots` with `main` method. The method receives three arguments, write code to parse them into `double` type.

For example:

```
if the values 2, 5, 3 are passed as arguments, then the output should be First root is : -1.0 Second root is : -1.5
If the values 3, 2, 1 are passed then the output should be Roots are imaginary
Similarly, if the values 2, 4, 2 are passed then the output should be Roots are equal and value is : -1.0
```

Note: Make sure to use the **print()** and not the **println()** method.

Note: Please don't change the package name.

Source Code:

q10851/QuadraticRoots.java

```
package q10851;
class QuadraticRoots {
        double a,b,c;
        void getData(String c1,String c2,String c3)
        {
          a=Double.valueOf(c1);
          b=Double.valueOf(c2);
          c=Double.valueOf(c3);
        }
        void roots()
        {
                double d;
                if(a==0)
                {
                        double root;
                        root=-c/b;
                        System.out.println("linear equation "+root);
                }
                else
                {
                        d=(b*b)-(4*a*c);
                        if(d==0){double root=-b/(2*a);
                        System.out.println("Roots are equal and value is : "+root);
                }
                else if (d>0)
                {
                        double r1,r2;
                        r1=(-b+Math.sqrt(d))/(2*a);
                        r2=(-b-Math.sqrt(d))/(2*a);
                        System.out.println("First root is : "+r1+" Second root is : "+r2);
                }
                else
                        System.out.println("Roots are imaginary");
                }
        }
        public static void main(String a[])
        {
                QuadraticRoots r = new QuadraticRoots();
                r.getData(a[0],a[1],a[2]);
                r.roots();
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
First root is : -0.6047152924789525 Second root is : -1.3952847075210475

Test Case - 2
User Output

```
Roots are equal and value is : -1.0
```

Test Case - 3
User Output
Roots are imaginary

S.No: 3	Exp. Name: ***Write a Java program to print the speeds of qualifying bikers in a Race***	**Date: 2023-09-07**

Aim:

Five bikers compete in a race such that they drive at a constant speed which may or may not be the same as the other.

To qualify the race, the speed of a racer must be more than or equal to the average speed of all the 5 racers.

Take as input the speed of each racer and print back the speeds of qualifying racers.

Write a class `Race` with a method `main(String[] args)`. The main method receives five arguments. You can write code to parse them into `double` data type.

For example, if the values `54.55, 53.57, 54, 56.25, 57.30` are passed as arguments to the main() method, then the output should be `The speed of the racers >= average speed 55.134 : 56.25 57.3`.

Note: Make sure to use the **print()** method and not the **println()** method.

Source Code:

Race.java

ID: 22K61A4245

Sasi Institute of Technology and Engineering (Autonomous)

```
class Race {
        public static void main(String[] args) {
        double s1,s2,s3,s4,s5,avg;
        s1=Double.parseDouble(args[0]);
        s2=Double.parseDouble(args[1]);
        s3=Double.parseDouble(args[2]);
        s4=Double.parseDouble(args[3]);
        s5=Double.parseDouble(args[4]);
        avg=(s1+s2+s3+s4+s5)/5;
        System.out.print("The speed of the racers >= average speed " +avg+ ": ");
        if(s1>avg)
        {
                System.out.print(","+s1);
        }
        if(s2>avg)
        {
                System.out.print(","+s2);
        }
        if(s3>avg)
        {
                System.out.print(","+s3);
        }
        if(s4>avg)
        {
                System.out.print(","+s4);
        }
        if(s5>avg)
        {
                System.out.print(","+s5);
        }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
The speed of the racers >= average speed 54.855999999999995: ,81.6,58.19,79.42

Test Case - 2
User Output
The speed of the racers >= average speed 78.0032: ,96.21,87.26,105.63

S.No: 4	Exp. Name: ***Search for an element in a given list of elements using Binary Search.***	**Date: 2023-09-07**

Aim:

Write a program to search for an element in a given list of elements using **Binary Search** mechanism.

Source Code:

q36414/BinarySearch.java

```
package q36414;
import java.util.*;
class BinarySearch
{
	public static void main(String[] args)
	{
		Scanner s=new Scanner(System.in);
		int n,key,pos=-1,mid,low,high,i;
		System.out.print("Enter the number of elements: ");
		n=s.nextInt();
		System.out.println("Enter the sorted elements:");
		int a[]=new int[n];
		for(i=0;i<n;i++)
		{
			a[i]=s.nextInt();
		}
		low=0;
		high=n-1;
		System.out.print("Enter the element to search for: ");
		key=s.nextInt();
		pos=BinarySearchDemo.Bs(a,n,low,high,key);
		if(pos==-1)
		{
			System.out.println("Element "+key+" not found in the list.");
		}
		else{
			System.out.println("Element "+key+" found at index "+pos);
		}
	}
}
class BinarySearchDemo
{
	public static int Bs(int a[],int n,int low,int high,int key)
	{
		int mid;
		while(low<=high)
		{
			mid=(low+high)/2;
			if(a[mid]<key)
			{
				low=mid+1;
			}
			if(a[mid]>key)
			{
				high=mid-1;
			}
			if(a[mid]==key)
			{
				return mid;
			}
		}
		return-1;
	}
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the number of elements:
5
Enter the sorted elements:
10 20 30 40 50
Enter the element to search for:
30
Element 30 found at index 2

Test Case - 2
User Output
Enter the number of elements:
8
Enter the sorted elements:
2 4 6 8 10 12 14 16
Enter the element to search for:
9
Element 9 not found in the list.

S.No: 5	Exp. Name: ***Sort a list using Bubble Sort Technique.***	Date: 2023-09-07

Aim:

Write a java program to sort the given list of elements using **Bubble Sort**.

Source Code:

q36415/BubbleSort.java

```
package q36415;
import java.util.*;
class BubbleSortDemo
{
        public static void Bubble(int a[],int n)
        {
                int i,j,temp;
                for(i=0;i<=n;i++)
                {
                        for(j=i+1;j<=n-1;j++)
                        {
                                if(a[i]>a[j])
                                {
                                        temp=a[i];
                                        a[i]=a[j];
                                        a[j]=temp;
                                }
                        }
                }
        }
}
class BubbleSort
{
        public static void main(String args[])
        {
                int n,i;
                Scanner s=new Scanner(System.in);
                System.out.print("Enter no of elements: ");
                n=s.nextInt();
                int a[]=new int[n];
                System.out.print("Enter the elements: \n");
                for(i=0;i<n;i++)
                {
                        a[i]=s.nextInt();
                }
                System.out.print("Sorted elements:\n");
                BubbleSortDemo bs =new BubbleSortDemo();
                bs.Bubble(a,n);
                for(i=0;i<n;i++)
                {
                        System.out.print(a[i]+" ");
                }

        }
}
```


ID: 22K61A4245

Sasi Institute of Technology and Engineering (Autonomous)

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter no of elements:
3
Enter the elements:
5 -1 25
Sorted elements:
-1 5 25

Test Case - 2
User Output
Enter no of elements:
6
Enter the elements:
15 54 87 69 32 151
Sorted elements:
15 32 54 69 87 151

S.No: 6	Exp. Name: ***Sort a list using Merge Sort Technique.***	Date: 2023-09-12

Aim:

Write a java program to sort the given list of elements using **Merge Sort**.

Source Code:

q36416/MergeSort.java

```
package q36416;
import java.util.*;
class MergeSort
{
        public static void main(String[] args)
        {
        int n,i;
        Scanner s = new Scanner(System.in);
        System.out.print("Enter no of elements: ");
        n=s.nextInt();
        int a[] = new int[n];
        System.out.println("Enter the elements:");
        for(i=0;i<n;i++)
                a[i]=s.nextInt();
        Merge.SplitAndMerge(a,0,n-1,n);
        System.out.print("Sorted array: \n");
        for(i=0;i<n;i++)
        System.out.print(a[i]+" ");
        }
}
class MergeSortDemo
{
        public static void MergeSorting(int a[],int low,int mid, int high,int n)
        {
                int i,j,k;
                int b[]=new int[n];
                i=low;
                j=mid+1;
                k=low;
                while(i<=mid&&j<=high)
                {
                        if(a[i]<=a[j])
                        {
                                b[k]=a[i];
                                i++;
                        }
                        else
                        {
                                b[k]=a[j];
                                j++;
                        }
                        k++;
                }
                if(i<=mid)
                {
                        while(i<=mid)
                        {
                                b[k]=a[i];
                                i++;
                                k++;
                        }
                        }
                        else
                        {
                                while(j<=high)
```

```
                                j++;
                                k++;
                        }
                }
                for(k=low;k<=high;k++)
                a[k]=b[k];
        }
}
class Merge
{
        public static void SplitAndMerge(int a[],int low,int high,int n)
        {
                int mid;
                if(low<high)
                {
                        mid=low+(high-low)/2;
                        Merge.SplitAndMerge(a,low,mid,n);
                        Merge.SplitAndMerge(a,mid+1,high ,n);
                        MergeSortDemo.MergeSorting(a,low,mid,high,n);
                }
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter no of elements:
3
Enter the elements:
100 50 75
Sorted array:
50 75 100

Test Case - 2
User Output
Enter no of elements:
4
Enter the elements:
1 3 5 2
Sorted array:
1 2 3 5

S.No: 7	Exp. Name: ***Write a Java program to Delete and Remove characters using StringBuffer class***	Date: 2023-09-23

Aim:

Write a class `JavaStringBufferDelete` with a **main** method to delete characters from a string using `StringBuffer` class.

Source Code:

JavaStringBufferDelete.java

```
import java.util.*;
import java.lang.StringBuffer;
public class JavaStringBufferDelete
{
        public static void main(String args[])
        {
                Scanner sc = new Scanner(System.in);
                System.out.print("string: ");
                String inputString=sc.nextLine();
                System.out.print("start index: ");
                int startIndex=sc.nextInt();
                System.out.print("end index: ");
                int endIndex=sc.nextInt();
                StringBuffer ste=new StringBuffer(inputString);
                ste.delete(startIndex,endIndex);
                System.out.print("Result after delete: "+ste.toString()+"\n");
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
string:
Hello world
start index:
0
end index:
6
Result after delete: world

ID: 22K61A4245

Sasi Institute of Technology and Engineering (Autonomous)

S.No: 8	Exp. Name: ***Write a Java program to implement a Class mechanism***	Date: 2023-09-21

Aim:

Write a Java program with a class name `Employee` which contains the data members **name** (String), **age** (int), **designation** (String), **salary** (double) and the methods **setData()**, **displayData()**.

The member function **setData()** is used to initialize the data members and **displayData()** is used to display the given employee data.

Write the **main()** method with in the class which will receive four arguments as **name**, **age**, **designation**and **salary**.

Create an object to the class `Employee` within the main(), call **setData()** with arguments and finally call the method **displayData()** to print the output.

If the input is given as command line arguments to the **main()** as **"Saraswathi", "27", "Teacher", "37250"** then the program should print the output as:

```
Name : Saraswathi
Age : 27
Designation : Teacher
Salary : 37250.0
```

Note: Please don't change the package name.

Source Code:

q11115/Employee.java

```
package q11115;
import java.util.*;
class Employee
{
        String name;
        int age;
        String designation;
        double Salary;
        void setData(String a[])
        {
                name=a[0];
                age=Integer.parseInt(a[1]);
                designation=a[2];
                Salary=Double.parseDouble(a[3]);
        }
        void displayData()
        {
                System.out.println("Name : "+name);
                System.out.println("Age : "+age);
                System.out.println("Designation : "+designation);
                System.out.println("Salary : "+Salary);
        }
        public static void main(String args[]){
                Employee em =new Employee();
                em.setData(args);
                em.displayData();
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Name : Ram
Age : 25
Designation : Team member
Salary : 25000.0

Test Case - 2
User Output
Name : Ravi
Age : 36
Designation : TeamLead
Salary : 35000.0

S.No: 9	Exp. Name: ***Write a Java program to implement a Constructor***	Date: 2023-09-21

Aim:

Write a Java program with a class name `Staff`, contains the data members **id** (int), **name** (String) which are initialized with a **parameterized constructor** and the method **show()**.

The member function **show()** is used to display the given staff data.

Write the **main()** method with in the class which will receive two arguments as **id** and **name**.

Create an object to the class `Staff` with arguments **id** and **name** within the **main()**, and finally call the method **show()** to print the output.

If the input is given as command line arguments to the **main()** as **"18", "Gayatri"** then the program should print the output as:

```
Id : 18
Name : Gayatri
```

Note: Please don't change the package name.

Source Code:

q11116/Staff.java

```
package q11116;
class Staff{
        Staff(int num,String a){
                System.out.println("Id : "+num);
                System.out.println("Name : "+a);
        }
        public static void main(String args[]){
                int a =Integer.parseInt(args[0]);
                String b = args[1];
                Staff s =new Staff(a,b);
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Id : 18
Name : Gayatri

Test Case - 2

User Output
Id : 45
Name : Akbar

S.No: 10	Exp. Name: ***Write a Java program to implement Constructor overloading***	Date: 2023-09-21

Aim:

Write a class `Box` which contains the data members **width**, **height** and **depth** all of type **double**.

Write the implementation for the below **3** *overloaded constructors* in the class `Box` :

- **Box()** - default constructor which initializes all the members with **-1**
- **Box(length)** - parameterized constructor with one argument and initialize all the members with the value in **length**

the members with the corresponding arguments

- **Box(width, height, depth)** - parameterized constructor with three arguments and initialize

Write a method `public double volume()` in the class `Box` to find out the **volume** of the given box.

Write the **main** method within the `Box` class and assume that it will receive either **zero** arguments, or **one** argument or **three** arguments.

For example, if the **main()** method is passed **zero** arguments then the program should print the output as:

```
Volume of Box() is : -1.0
```

Similarly, if the **main()** method is passed **one** argument : **2.34**, then the program should print the output as:

```
Volume of Box(2.34) is : 12.812903999999998
```

then the program should print the output as: Likewise, if the **main()** method is passed **three** arguments : **2.34, 3.45, 1.59**, then the program should print the output as:

```
Volume of Box(2.34, 3.45, 1.59) is : 12.836070000000001
```

Note: Please don't change the package name.

Source Code:

`q11267/Box.java`

```
package q11267;
class Box {
        double width,height,depth;
        double volume()
        {
                return width*height*depth;
        }
        Box()
        {
                width=-1;
                height=-1;
                depth=-1;
                System.out.println("Volume of Box() is : "+volume());
        }
        Box(String len)
        {
                width=height=depth=Double.parseDouble(len);
                System.out.println("Volume of Box("+width+") is : "+volume());
        }
        Box(String w,String h,String d)
        {
                width=Double.parseDouble(w);
                height=Double.parseDouble(h);
                depth=Double.parseDouble(d);
                System.out.println("Volume of Box("+width+", "+height+", "+depth+") is :
"+volume());
        }
        public static void main(String a[]){
                int n=a.length;
                Box b;
                if(n==0)
                        b=new Box();
                else if(n==1)
                        b=new Box(a[0]);
                else
                    b=new Box(a[0],a[1],a[2]);

        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Volume of Box() is : -1.0

Test Case - 2
User Output
Volume of Box(3.0) is : 27.0

Test Case - 3
User Output
Volume of Box(2.3, 3.5, 6.5) is : 52.324999999999996

S.No: 11	Exp. Name: ***Write a Java program to implement Method overloading***	Date: 2023-09-21

Aim:

Write a Java program with a class name `Addition` with the methods `add(int, int)`, `add(int, float)`, `add(float, float)` and `add(float, double, double)` to add values of different argument types.

Write the **main(String[])** method within the class and assume that it will always receive a total of **6** command line arguments at least, such that the first **2** are **int**, next **2** are **float** and the last **2** are of type **double**.

If the **main()** is provided with arguments : **1, 2, 1.5f, 2.5f, 1.0, 2.0** then the program should print the output as:

```
Sum of 1 and 2 : 3
Sum of 1.5 and 2.5 : 4.0
Sum of 2 and 2.5 : 4.5
Sum of 1.5, 1.0 and 2.0 : 4.5
```

Note: Please don't change the package name.

Source Code:

q11266/Addition.java

```
package q11266;
class Addition {
	int add(int a, int b)
	{
		return a+b;
	}
	float add(int a, float b)
	{
		return a+b;
	}
	float add(float a,float b)
	{
		return a+b;
	}
	float add(float a,float b,float c)
	{
		return a+b+c;
	}
	double add(float a,double b,double c)
	{
		return a+b+c;
	}
	public static void main(String args[]){
		int a,b;
		float x,y;
		double p,q;
		Addition ob=new Addition();
		a=Integer.parseInt(args[0]);
		b=Integer.parseInt(args[1]);
		x=Float.parseFloat(args[2]);
		y=Float.parseFloat(args[3]);
		p=Double.parseDouble(args[4]);
		q=Double.parseDouble(args[5]);
		System.out.println("Sum of "+a+" and "+b+" : "+ob.add(a,b));
		System.out.println("Sum of "+x+" and "+y+" : "+ob.add(x,y));
		System.out.println("Sum of "+b+" and "+y+" : "+ob.add(b,y));
		System.out.println("Sum of "+x+", "+p+" and "+q+" : "+ob.add(x,p,q));
	}
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Sum of 2 and 1 : 3
Sum of 5.0 and 3.6 : 8.6
Sum of 1 and 3.6 : 4.6
Sum of 5.0, 9.2 and 5.26 : 19.46

S.No: 12	Exp. Name: ***Write a Java program to implement Single Inheritance***	Date: 2023-09-21

Aim:

Write a Java program to illustrate the **single inheritance** concept.

Create a class `Marks`

- contains the data members **id** of **int** data type, **javaMarks**, **cMarks** and **cppMarks** of **float** data type
- write a method **setMarks()** to initialize the data members
- write a method **displayMarks()** which will display the given data

Create another class `Result` which is derived from the class `Marks`

- contains the data members **total** and **avg** of **float** data type
- write a method **compute()** to find total and average of the given marks
- write a method **showResult()** which will display the total and avg marks

Write a class `SingleInheritanceDemo` with **main()** method it receives four arguments as **id**, **javaMarks**, **cMarks** and **cppMarks**.

Create object only to the class `Result` to access the methods.

If the input is given as command line arguments to the **main()** as **"101", "45.50", "67.75", "72.25"** then the program should print the output as:

```
Id : 101
Java marks : 45.5
C marks : 67.75
Cpp marks : 72.25
Total : 185.5
Avg : 61.833332
```

Note: While computing the total marks, add the marks in the following order only **javaMarks**, **cMarks** and **cppMarks**

Source Code:

q11263/SingleInheritanceDemo.java

```
package q11263;
class Marks{
        int id;
        static float javaMarks,cMarks,cppMarks;
        void setMarks(String a[]){
                id=Integer.parseInt(a[0]);
                javaMarks=Float.parseFloat(a[1]);
                cMarks=Float.parseFloat(a[2]);
                cppMarks=Float.parseFloat(a[3]);
        }
        void displayMarks(){
                System.out.println("Id : "+id);
                System.out.println("Java marks : "+javaMarks);
                System.out.println("C marks : "+cMarks);
                System.out.println("Cpp marks : "+cppMarks);
        }
}
class Result extends Marks
{
        float total,avg;
        void compute(String a[]){
                Marks m =new Marks();
                super.setMarks(a);
                total=m.javaMarks+m.cMarks+m.cppMarks;
                avg=total/3;
        }
        void showResult(){
                super.displayMarks();
                System.out.println("Total : "+total);
                System.out.println("Avg : "+avg);
        }
}
class SingleInheritanceDemo{
        public static void main(String a[]){
                Result r =new Result();
                r.compute(a);
                r.showResult();
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Id : 102
Java marks : 35.6
C marks : 45.0
Cpp marks : 65.5
Total : 146.1
Avg : 48.7

Test Case - 2
User Output
Id : 101
Java marks : 45.5
C marks : 67.75
Cpp marks : 72.25
Total : 185.5
Avg : 61.833332

Test Case - 3
User Output
Id : 103
Java marks : 50.5
C marks : 46.8
Cpp marks : 52.65
Total : 149.95001
Avg : 49.983337

S.No: 13	Exp. Name: ***Write a Java program to implement Multilevel Inheritance***	Date: 2023-09-21

Aim:

Write a Java program to illustrate the **multilevel inheritance** concept.

Create a class `Student`

- contains the data members **id** of **int** data type and **name** of **string** type
- write a method **setData()** to initialize the data members
- write a method **displayData()** which will display the given **id** and **name**

Create a class `Marks` which is derived from the class `Student`

- contains the data members **javaMarks**, **cMarks** and **cppMarks** of **float** data type
- write a method **setMarks()** to initialize the data members
- write a method **displayMarks()** which will display the given data

Create another class `Result` which is derived from the class `Marks`

- contains the data members **total** and **avg** of **float** data type
- write a method **compute()** to find total and average of the given marks
- write a method **showResult()** which will display the total and avg marks

Write a class `MultilevelInheritanceDemo` with the **main()** method which will receive five arguments as **id**, **name**, **javaMarks**, **cMarks** and **cppMarks**.

Create object only to the class `Result` to access the methods.

If the input is given as command line arguments to the **main()** as **"99", "Lakshmi", "55.5", "78.5", "72"** then the program should print the output as:

```
Id : 99
Name : Lakshmi
Java marks : 55.5
C marks : 78.5
Cpp marks : 72.0
Total : 206.0
Avg : 68.666664
```

Note: Please don't change the package name.

Source Code:

q11264/MultilevelInheritanceDemo.java

```
package q11264;
class Student
{
	int id;
	String name;
	void setData(String a[])
	{
		id = Integer.parseInt(a[0]);
		name=a[1];
	}
	void displayData()
	{
		System.out.println("Id : "+id);
		System.out.println("Name : "+name);
	}
}
class Marks extends Student
{
	static float javaMarks,cMarks,cppMarks;
	void setMarks(String a[]){
		super.setData(a);
		javaMarks=Float.parseFloat(a[2]);
		cMarks=Float.parseFloat(a[3]);
		cppMarks=Float.parseFloat(a[4]);
	}
	void displayMarks()
	{
		super.displayData();
		System.out.println("Java marks : "+javaMarks);
		System.out.println("C marks : "+cMarks);
		System.out.println("Cpp marks : "+cppMarks);
	}
}
class Result extends Marks{
	float total,avg;
	void compute(String a[])
	{
		super.setMarks(a);
		Marks m =new Marks();
		total=m.javaMarks+m.cMarks+m.cppMarks;
		avg=total/3;
	}
	void showResult()
	{
		super.displayMarks();
		System.out.println("Total : "+total);
		System.out.println("Avg : "+avg);
	}
}
class MultilevelInheritanceDemo
{
	public static void main(String a[])
	{
		Result r =new Result();
		r.compute(a);
		r.showResult();
```

```
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Id : 99
Name : Geetha
Java marks : 56.0
C marks : 75.5
Cpp marks : 66.6
Total : 198.1
Avg : 66.03333

Test Case - 2
User Output
Id : 199
Name : Lakshmi
Java marks : 55.5
C marks : 78.5
Cpp marks : 78.0
Total : 212.0
Avg : 70.666664

S.No: 14	Exp. Name: ***Write a Java program to find Areas of different Shapes using abstract class***	Date: 2023-09-21

Aim:

Write a Java program to illustrate the **abstract class** concept.

Create an abstract class `CalcArea` and declare the methods **triangleArea(double b, double h)**, **rectangleArea(double l, double b)**, **squareArea(double s)**, **circleArea(double r)**.

Create a class `FindArea` which extends the abstract class `CalcArea` used to find areas of triangle, rectangle, square, circle.

Write a class `Area` with the **main()** method which will receive **two** arguments and convert them to **double** type.

If the input is given as command line arguments to the **main()** as **"1.2","2.7"** then the program should print the output as:

```
Area of triangle : 1.62
Area of rectangle : 3.24
Area of square : 1.44
Area of circle : 22.890600000000006
```

Note: Please don't change the package name.

Source Code:

q11286/Area.java

```
package q11286;
public class Area {
	public static void main(String args[]) {
		FindArea area = new FindArea();
		area.triangleArea(Double.parseDouble(args[0]), Double.parseDouble(args[1]));
		area.rectangleArea(Double.parseDouble(args[0]),
Double.parseDouble(args[1]));
		area.squareArea(Double.parseDouble(args[0]));
		area.circleArea(Double.parseDouble(args[1]));
	}
}
abstract class CalcArea
{
	abstract void triangleArea(double b ,double h);
	abstract void rectangleArea(double l,double b);
	abstract void squareArea(double s);
	abstract void circleArea(double r);
}
class FindArea extends CalcArea
{
	void triangleArea(double b ,double h)
	{
		System.out.println("Area of triangle : "+(0.5*b*h));
	}
	void rectangleArea(double l,double b)
	{
		System.out.println("Area of rectangle : "+(l*b));
	}
	void squareArea(double s)
	{
		System.out.println("Area of square : "+(s*s));
	}
	void circleArea(double r)
	{
		System.out.println("Area of circle : "+(3.14*r*r));
	}
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Area of triangle : 7.529400000000001
Area of rectangle : 15.058800000000002
Area of square : 12.6736
Area of circle : 56.18370600000001

Test Case - 2
User Output
Area of triangle : 83.14375000000001

```
Area of rectangle : 166.28750000000002
Area of square : 157.50250000000003
Area of circle : 551.26625
```