1. **package** codecamp;

**import** java.util.Comparator;

**public** **class** Student

{

String firstName;

String lastName;

**int** grade;

Student()

{}

Student(String firstName,String lastName,**int** grade)

{

**this**.firstName=firstName;

**this**.lastName=lastName;

**this**.grade=grade;

}

}

**package** codecamp;

**import** java.util.ArrayList;

**import** java.util.Comparator;

**import** java.util.List;

**public** **class** StudentTester {

**public** **static** **void** main(String[] args)

{

Student s=**new** Student("Raj","Kumar",1);

Student s1=**new** Student("Rama","Rao",2);

List<Integer> al=**new** ArrayList<Integer>();

al.add(s.grade);

al.add(s1.grade);

}

**class** Grade1 **implements** Comparator<Stud>

{

// Used for sorting in ascending order of

**public** **int** compare(Student a, Student b)

{

**return** a.grade - b.grade;

}

@Override

**public** **int** compare(Stud o1, Stud o2) {

// **TODO** Auto-generated method stub

**return** 0;

}

}

1. **package** codecamp;

**import** java.util.Scanner;

**abstract** **class** Shape

{

**abstract** **void** getPerimeter(**float** value);

**abstract** **void** getArea(**float** value);

}

**class** Square **extends** Shape

{

@Override

**void** getPerimeter(**float** side)

{

**float** Perimeter=2\*side;

System.***out***.println("Perimeter of the Square "+Perimeter);

}

@Override

**void** getArea(**float** side)

{

**float** area=side\*side;

System.***out***.println("Area of the Square "+area);

}

}

**class** Pentagon **extends** Shape

{

@Override

**void** getPerimeter(**float** side)

{

}

@Override

**void** getArea(**float** side)

{

**double** area=(Math.*sqrt*(5\*(5+2 \* Math.*sqrt*(5)))\* side \* side)/4;

System.***out***.println("Area of the pentagon "+area);

}

}

**class** Circle **extends** Shape

{

**private** **static** **final** **float** ***PI*** = (**float**) 3.14;

**float** radius;

@Override

**void** getPerimeter(**float** radius)

{

**float** perimeter= 2\****PI***\*radius;

System.***out***.println("Perimeter of the circle "+perimeter);

}

@Override

**void** getArea(**float** radius)

{

**float** area=***PI***\*radius\*radius;

System.***out***.println("Area of circle "+area);

}

}

**public** **class** Shapedemo

{

**private** **static** **final** **float** ***PI*** = (**float**) 3.14;

**public** **static** **void** main(String[] args)

{

Scanner sc=**new** Scanner(System.***in***);

Shape s=**new** Circle();

Square sq=**new** Square();

Pentagon p=**new** Pentagon();

System.***out***.println("Enter radius of the circle");

**float** radius=sc.nextFloat();

s.getPerimeter(radius);

s.getArea(radius);

System.***out***.println("Enter side of the square ");

**float** side=sc.nextFloat();

sq.getArea(side);

sq.getPerimeter(side);

System.***out***.println("Enter side of the pentagon");

**float** side1=sc.nextFloat();

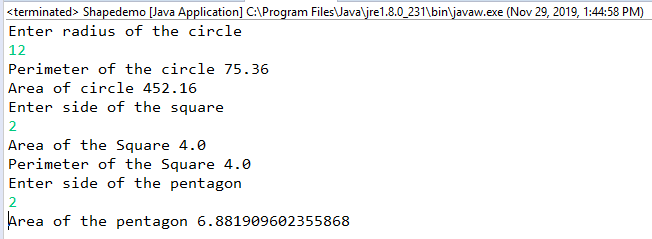
p.getArea(side1);

p.getPerimeter(side1);

}

}

Output:



3.

**package** codecamp;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** Arraylist

{

**public** **static** **void** main(String[] args)

{

**int** key;

List<Integer> al=**new** ArrayList<Integer>();

al.add(23);

al.add(35);

al.add(123);

al.add(273);

al.add(1223);

key=273;

al.contains(key);

**if**(al.contains(key)==**true**)

{

System.***out***.println(key+" found in Array List");

}

**else**

System.***out***.println(key+" Not found in Array List");

}

**boolean** contains(**int** key)

{

**return** **true**;

}

}

Output:



4.

**package** codecamp;

**import** java.util.Scanner;

**public** **class** Employee

{

**static** Scanner *sc*=**new** Scanner(System.***in***);

**public** **static** **void** main(String[] args)

{

**int** empId[]=**new** **int**[20];

empId=*input*();

*sort*(empId);

}

**static** **int**[] input()

{

**int**[] empId=**new** **int**[20];

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

{

empId[i]=*sc*.nextInt();

}

**return** empId;

}

//sorting in descending order

**static** **void** sort(**int**[] empId)

{

**int** temp;

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

{

**for**(**int** j=i+1;j<empId.length;j++)

{

**if**(empId[i]<=empId[j])

{

temp=empId[i];

empId[i]=empId[j];

empId[j]=temp;

}

}

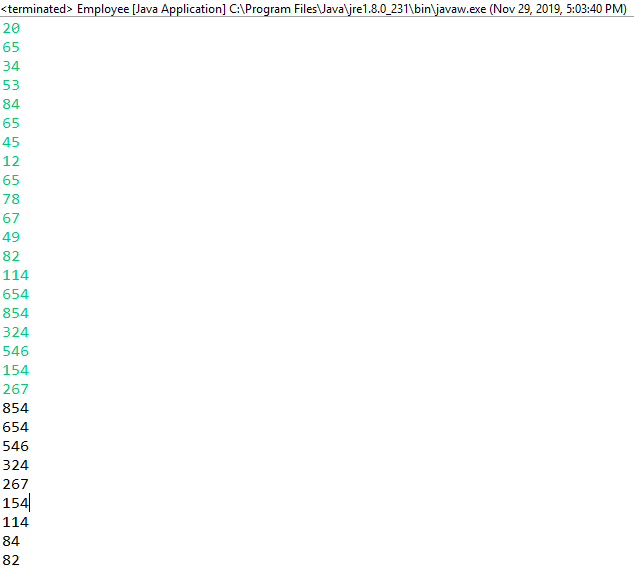
System.***out***.println(empId[i]);

}

}

}

Output:



6. **package** codecamp;

**import** java.io.IOException;

**import** java.util.Scanner;

**public** **class** Car {

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

{

**int** n=0,m=0;

Scanner sc=**new** Scanner(System.***in***);

**int**[] evenNumber=**new** **int**[10];

**int**[] oddNumber=**new** **int**[10];

**int**[] carNumber=**new** **int**[10];

System.***out***.println("Enter car numbers");

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

{

carNumber[i]=sc.nextInt();

}

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

{

**int** temp=carNumber[i];

**if**(temp%2==0)

{

evenNumber[n]=carNumber[i];

System.***out***.println("Even Car Number "+evenNumber[n]);

n++;

}

**else**

{

oddNumber[m]=carNumber[i];

System.***out***.println("Odd Car Number "+oddNumber[m]);

m++;

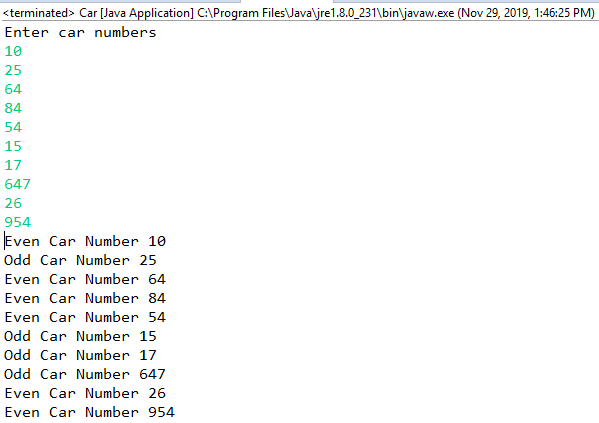
}

}

}

}

Output:



7.

**package** codecamp;

**public** **class** Matrixmultiplication

{

**public** **static** **void** main(String args[])

{

//creating two matrices

**int** a[][]={{1,2,3},{4,5,6},{7,8,9}};

**int** b[][]={{1,2,1},{2,5,2},{3,8,3}};

//creating another matrix to store the multiplication of two matrices

**int** c[][]=**new** **int**[3][3]; //3 rows and 3 columns

//multiplying and printing multiplication of 2 matrices

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

{

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

{

c[i][j]=0;

**for**(**int** k=0;k<3;k++)

{

c[i][j]+=a[i][k]\*b[k][j];

}

//printing matrix element

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

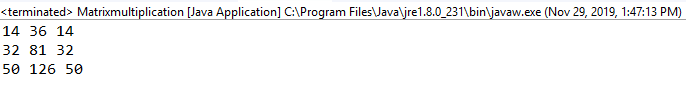
}

System.***out***.println();

}

}}

Output:



8.

**package** codecamp;

**public** **class** Test1 {

**public** **static** **void** main(String[] args)

{

Test1 t=**new** Test1();

**long** value=t.methodA((**byte**) 6, 2.2);

System.***out***.println(value);

}

**long** methodA(**byte** x, **double** y)

{

**return** (**long**) (x / y \* 2);

}

}

Output:



10.

**package** codecamp;

**public** **class** Add {

**public** **static** **void** main(String[] args) {

**int** a = 0, b = 0;

**int** sum=a+b;

System.***out***.println("Enter two numbers to add: ");

System.***out***.println (a);

System.***out***.println (b);

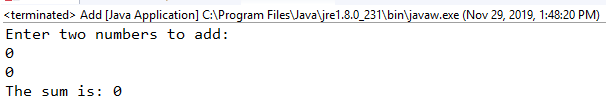
System.***out***.println ("The sum is: "+sum);

}

}

Output:

The declaration is valid as java compiler reads code line by line and the program does not have any syntax errors. Logically the code is not correct as the sum is already done and in the next line it is displaying to enter two numbers and displaying the numbers. The output shows the sum as 0 and values of a, b as 0 because a and b are of integer values whose default values will be 0.



11.The program is behavior platform dependent and prints two new lines spaces in the console.

12.The program goes to infinity loop.

13.The program prints only “Hello World”. After that the program stops executing. To execute the finally block we need to remove System.exit.

**package** codecamp;

**public** **class** HelloWorld {

**public** **static** **void** main(String[] args) {

**try**

{ System.***out***.println("Hello world");

//System.exit(0);

}

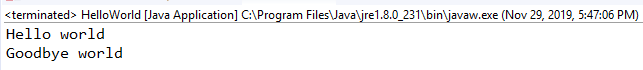
**finally**

{ System.***out***.println("Goodbye world"); }

}

}

Output:



14. The program throws IOException.

**package** codecamp;

**import** java.io.IOException;

**class** San

{

San()**throws** IOException

{}

}

**class** Foundry **extends** San

{

Foundry() **throws** IOException

{}

**public** **static** **void** main(String[] args)

{}

}

15.The program does not show any error.

**package** codecamp;

**class** Test {

**static** **final** **int** ***x*** = 11;

**private** **int** y = 33;

**public** **void** method1(**int** x)

{

Test t = **new** Test();

x = 22;

y = 44;

System.***out***.println("Test.x: " + Test.***x***);

System.***out***.println("t.x: " + t.***x***);

System.***out***.println("t.y: " + t.y);

System.***out***.println("y: " + y);

}

}

**public** **class** Test2

{

**public** **static** **void** main(String args[])

{

Test t = **new** Test();

t.method1(22);

}

}

Output:

