1.Create a class named 'Student' with String variable 'name' and integer variable 'roll\_no'. Assign the value of roll\_no as '2' and that of name as "John" by creating an object of the class Student.

**public** **class** Students {

String name;

**int** roll\_no;

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

Students s1= **new** Students();

s1.name="John";

s1.roll\_no=2;

}

}

Out put = Jhon

2

2.Assign and print the roll number, phone number and address of two students having names "Sam" and "John" respectively by creating two objects of class 'Student'.

**public** **class** Students {

String name;

**int** roll\_no;

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

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

Students s1= **new** Students();

Students s2= **new** Students();

s1.name="John";

s1.roll\_no=2;

s2.name="Sam";

s2.roll\_no=3;

System.***out***.println(s1.roll\_no+" "+s1.name);

System.***out***.print(s2.roll\_no+" "+s2.name);

}

}

Out put:

2 John

3 Sam

3. Write a program to print the area and perimeter of a triangle having sides of 3, 4 and 5 units by creating a class named 'Triangle' without any parameter in its constructor.

**package** assign;

**public** **class** Perimeter {

**int** side1 = 3;

**int** side2 = 4;

**int** side3 = 5;

**int** peri = side1+side2+side3;

**void** display() {

System.***out***.println("Perimeter of triangle is :" + peri);

System.***out***.println("Area of triangle is: "+ (0.5\*side2\*side3));

}

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

perimeter obj = **new** perimeter();

obj.display();

}

}

Output = Perimeter of triangle is :12

Area of triangle is: 10.0

4.Write a program to print the area and perimeter of a triangle having sides of 3, 4 and 5 units by creating a class named 'Triangle' with constructor having the three sides as its parameters.

**package** assign;

**public** **class** Perimeter\_cons {

**int** side1 ;

**int** side2 ;

**int** side3 ;

Perimeter\_cons(**int** s1, **int** s2 , **int** s3){

side1 = s1;

side2 = s2;

side3 = s3;

}

**void** display() {

System.***out***.println("Area of triangle is: "+(0.5\*side2\*side3));

System.***out***.println("Perimeter of triangle is: "+(side1+side2+side3));

}

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

Perimeter\_cons obj = **new** Perimeter\_cons(3,4,5);

obj.display();

}

}

Output = Area of triangle is: 10.0

Perimeter of triangle is: 12

5. Write a program to print the area of two rectangles having sides (4,5) and (5,8) respectively by creating a class named 'Rectangle' with a method named 'Area' which returns the area and length and breadth passed as parameters to its constructor.

**package** assign;

**public** **class** Rectangle {

**int** l;

**int** b;

Rectangle(**int** len , **int** bre){

l = len ;

b = bre ;

}

**int** area() {

**int** a = l\*b;

**return** a;

}

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

Rectangle obj = **new** Rectangle(4,5);

Rectangle obj1 = **new** Rectangle(5,8);

System.***out***.println("Area of Rectangle 1 st is: "+ obj.area());

System.***out***.println("Area of rectangle 2nd is :" +obj1.area());

}

}

Out put is::

Area of Rectangle 1 st is: 20

Area of rectangle 2nd is :40

6. Write a program to print the area of a rectangle by creating a class named 'Area' taking the values of its length and breadth as parameters of its constructor and having a method named 'returnArea' which returns the area of the rectangle. Length and breadth of rectangle are entered through keyboard.

import java.util.\*;

public class Area {

int length;

int breadth;

int a;

Area(int l , int b){

length = l;

breadth = b;

}

void area() {

a = length\*breadth;

}

int returnArea(){

return a;

}

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

int obj = sc.nextInt();

int obj3 = sc.nextInt();

Area obj1 = new Area(obj,obj3);

obj1.area();

System.out.println("Area of rectangle is :"+obj1.returnArea());

}

}

Out put is::

5

5

Area of rectangle is :25

7. Print the sum, difference and product of two complex numbers by creating a class named 'Complex' with separate methods for each operation whose real and imaginary parts are entered by user.

Code:

import java.util.\*;

class Complex{

int real;

int imag;

public Complex(int r, int i){

real = r;

imag = i;

}

public static Complex add(Complex a, Complex b){

return new Complex((a.real+b.real),(a.imag+b.imag));

}

public static Complex diff(Complex a, Complex b){

return new Complex((a.real-b.real),(a.imag-b.imag));

}

public static Complex product(Complex a, Complex b){

return new Complex(((a.real\*b.real)-(a.imag\*b.imag)),((a.real\*b.imag)+(a.imag\*b.real)));

}

public void printComplex(){

if(real == 0 && imag!=0){

System.out.println(imag+"i");

}

else if(imag == 0 && real!=0){

System.out.println(real);

}

else{

System.out.println(real+"+"+imag+"i");

}

}

}

class Two{

public static void main(String[] args){

Complex c = new Complex(1,2);

Complex d = new Complex(34);

Complex e = Complex.add(c,d);

Complex f = Complex.diff(c,d);

Complex g = Complex.product(c,d);

e.printComplex();

f.printComplex();

g.printComplex();

}

}

Out put is::

4+6i

-2+-2i

-5+10i

8.Write a program that would print the information (name, year of joining, salary, address) of three employees by creating a class named 'Employee'. The output should be as follows:  
Name        Year of joining        Address  
Robert            1994                64C- WallsStreat  
Sam                2000                68D- WallsStreat  
John                1999                26B- WallsStreat

Code::

**package** assign;

**public** **class** Employe {

String name;

**int** Year\_of\_joining;

String Address;

**void** details(String x ,**int** y,String z){

name=x;

Year\_of\_joining=y;

Address=z;

}

**void** display(){

System.***out***.println(name+"\t\t"+Year\_of\_joining+"\t\t"+Address);

}

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

Employe arr[]= **new** Employe[3];

arr[0]=**new** Employe();

arr[1]=**new** Employe();

arr[2]=**new** Employe();

arr[1].details("Robert",1994,"64C-WallsStreet");

arr[2].details("Sam",2000,"68C-wallzSt");

arr[0].details("Jhon",1999,"56B-mountSt");

System.***out***.println("NAME\t YEAR OF JOINING\tADDRESS");

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

arr[i].display();

}

}

}

Out put::

NAME YEAR OF JOINING ADDRESS

Jhon 1999 56B-mountSt

Robert 1994 64C-WallsStreet

Sam 2000 68C-wallzSt

9. The Matrix class has methods for each of the following:  
1 - get the number of rows  
2 - get the number of columns  
3 - set the elements of the matrix at given position (i,j)  
4 - adding two matrices. If the matrices are not addable, "Matrices cannot be added" will be displayed.  
5 - multiplying the two matrices

Code ::

class Matrix{

int row;

int column;

int[][] a;

public Matrix(int r, int c){

row = r;

column = c;

a = new int[row][column];

}

public int getRows(){

return row;

}

public int getColumns(){

return column;

}

public int getElement(int r, int c){

return a[r][c];

}

public void setElement(int r, int c, int element){

a[r][c] = element;

}

public static Matrix add(Matrix x, Matrix y){

if((x.row == y.row) && (x.column == y.column)){

Matrix m = new Matrix(x.row,x.column);

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

for(int j = 0;j<m.column;j++){

m.setElement(i,j,(x.getElement(i,j)+y.getElement(i,j)));

}

}

return m;

}

else{

System.out.println("Matrices can not be added");

return new Matrix(0,0);

}

}

public static Matrix product(Matrix x, Matrix y){

Matrix m = new Matrix(x.row,y.column);

for(int j = 0;j<x.row;j++){

for(int i = 0;i<y.column;i++){

int sum = 0;

for(int k = 0;k<x.column;k++){

sum = sum+(x.getElement(j,k)\*y.getElement(k,i));

}

m.setElement(j,i,sum);

}

}

return m;

}

public void printMatrix(){

System.out.println("Matrix is :");

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

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

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

}

System.out.println("");

}

}

}

class Two{

public static void main(String[] args){

Matrix m = new Matrix(3,3);

Matrix n = new Matrix(3,3);

int k = 1;

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

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

m.setElement(i,j,k);

k++;

n.setElement(i,j,k);

k++;

}

}

m.printMatrix();

n.printMatrix();

Matrix o = Matrix.add(m,n);

o.printMatrix();

Matrix p = Matrix.product(m,n);

p.printMatrix();

}

}

Out put is::

Matrix is :

1 3 5

7 9 11

13 15 17

Matrix is :

2 4 6

8 10 12

14 16 18

Matrix is :

3 7 11

15 19 23

27 31 35

Matrix is :

96 114 132

240 294 348

384 474 564

10. Write a program to print the names of students by creating a Student class. If no name is passed while creating an object of Student class, then the name should be "Unknown", otherwise the name should be equal to the String value passed while creating object of Student class.

Code:

**package** assign;

**public** **class** Student {

String stud ;

Student(){

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

}

Student(String str){

stud=str;

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

}

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

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

Student s1=**new** Student();

Student s2= **new** Student("Axis");

}

}

Output is::

null

Axis

11. Will the following code snippet compile successfully? If yes, what is the output of the following program?

public class Myclass

{

private int x = 10;

static int m1() {

int y = x;

return y;

}

public static void main(String[] args) {

m1();

}

}

Out put is::

No,it will not run gives error because non static variable can not be reffrenced by static context.

12.Identify the error in the following code snippet. If there is no error then what will be the output of the program?

public class Myclass

{

private int x = 10;

static int m1()

{

Myclass obj = new Myclass();

int y = obj.x;

return y;

}

public static void main(String[] args) {

System.out.println(m1());

}

}

Output is::

yes, it will run and output is 10.

13.what is the output

public class Myclass

{

static int a = 20;

static int b = 30;

static int c = 40;

Myclass()

{

a = 200;

}

static void m1() {

b = 300;

}

static {

c = 400;

}

public static void main(String[] args) {

System.out.println(a);

System.out.println(b);

System.out.println(c);

}

}

Out put is::

yes,it will run and output is a=20,b=30,c=400.

14.whats the output

public class Myclass {

static int a = 20;

Myclass() {

a = 200;

}

public static void main(String[] args) {

new Myclass();

System.out.println(a);

}

}

Out put::

yes,it will run and output is 200;

15.Whats the error in the code

public class Myclass {

static int a = 20;

Myclass() {

a++;

}

void m1() {

a++;

System.out.println(a);

}

public static void main(String[] args)

{

Myclass obj = new Myclass();

Myclass obj2 = new Myclass();

Myclass obj3 = new Myclass();

obj3.m1();

}

Out put is::

No Error.o/p is 24.

16.Will this program execute what will be the output

public class Test

{

Test(Test t) {

m1();

System.out.println("Constructor");

}

void m1() {

m2();

System.out.println("Instance method");

}

static void m2() {

System.out.println("Static method");

}

public static void main(String[] args)

{

new Test(null);

}

}

Output is::

Yes..o/p= Static method

Instance method

Constructor

17.whats the output

import java.util.Scanner;

class Figure

{

final int length = 5;

final int bredth = 4;

final void area()

{

int a = length \* bredth;

System.out.println("Area:"+a);

}

}

class Rectangle extends Figure

{

final void rect()

{

System.out.println("This is rectangle");

}

}

final public class Final\_Use extends Rectangle

{

public static void main(String[] args)

{

Final\_Use obj = new Final\_Use();

obj.rect();

obj.area();

}

}

Output is::

This is rectangle

Area:20

18. Create a class with a method that prints "This is parent class" and its subclass with another method that prints "This is child class". Now, create an object for each of the class and call  
a - method of parent class by object of parent class  
b - method of child class by object of child class  
c - method of parent class by object of child class

Code::

**class** Method{

**void** set\_method() {

System.***out***.println("This is Parent class");

}

}

**class** Method\_m1 **extends** Method{

**void** get(){

System.***out***.println("This is child");

}

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

Method obj=**new** Method();

Method\_m1 obj1=**new** Method\_m1();

obj.set\_method();

obj1.get();

obj1.set\_method();

}

}

Out put is::

This is Parent class

This is child

This is Parent class

19.  
Create a class named 'Member' having the following members:  
Data members  
1 - Name  
2 - Age  
3 - Phone number  
4 - Address  
5 - Salary  
It also has a method named 'printSalary' which prints the salary of the members.  
Two classes 'Employee' and 'Manager' inherits the 'Member' class. The 'Employee' and 'Manager' classes have data members 'specialization' and 'department' respectively. Now, assign name, age, phone number, address and salary to an employee and a manager by making an object of both of these classes and print the same.

Code::

**package** assign;

**public** **class** Memeber {

String name;

**int** age;

**long** PH;

String Address;

**int** Salary;

**void** printSalary() {

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

}

}

**public** **class** Manager {

String specialization ;

String Departement;

}

**public** **class** Employes **extends** Memeber{

String specialization ;

String Departement;

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

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

Employes e= **new** Employes();

e.name="Abhi";

e.age=21;

e.Address="Delh nagar";

e.PH=784577575;

e.Salary=24000;

Memeber a=**new** Memeber();

a.name="Ashish";

a.age=23;

a.Address="Jbp";

a.PH=877676765;

a.Salary=500000;

e.printSalary();

a.printSalary();

}

}

Output is::

24000

500000

20.

Create a class named 'Rectangle' with two data members 'length' and 'breadth' and two methods to print the area and perimeter of the rectangle respectively. Its constructor having parameters for length and breadth is used to initialize length and breadth of the rectangle. Let class 'Square' inherit the 'Rectangle' class with its constructor having a parameter for its side (suppose s) calling the constructor of its parent class as 'super(s,s)'. Print the area and perimeter of a rectangle and a square.

Code::

**package** assign;

**public** **class** Rectangless {

**int** len,breadth;

Rectangless(**int** l,**int** b){

len=l\*b;

breadth=(2\*l)+(2\*b);

}

**void** area() {

System.***out***.println("Area "+len);

}

**void** perimeter() {

System.***out***.println("perimeter "+breadth);

}

}

**public** **class** Squares **extends** Rectangless{

Squares(**int** l, **int** b) {

**super**(l, b);

// **TODO** Auto-generated constructor stub

}

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

Squares obj=**new** Squares(5,3);

Rectangless obj2=**new** Rectangless(7,9);

obj2.area();

obj2.perimeter();

obj.area();

obj.perimeter();

}

}

Output::

Area63

perimeter32

Area15

perimeter16

21. What is the out put

class One

{

protected void getData()

{

System.out.println("Inside GFG");

}

}

class Two extends One

{

protected void getData()

{

System.out.println("Inside GeeksforGeeks");

}

}

public class Test

{

public static void main(String[] args)

{

One obj = new Two();

obj.getData();

}

}

Output is:: Inside GeeksforGeeks

22. **can we overload main() method?**

Ans:: yes we can overload main method. main method must not be static main method.

**23.what is the output**

**public class Myclass {**

**static int a = 20;**

**Myclass() {**

**a = 200;**

**}**

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

**new Myclass();**

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

**}**

}

Output is::200.

24. what is the output

**class A {**

**void sum(int x, int y){**

**System.out.println("Sum of two numbers: " +(x+y));**

**}**

**void sum(int x, int y, int z){**

**System.out.println("Sum of three numbers: " +(x+y+z));**

**}**

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

**A a = new A();**

**a.sum(20, 30);**

**a.sum(30, 40, 50);**

**}**

**}**

**Output is::** **Sum of two numbers: 50**

**Sum of three numbers: 120**

**25.what is the output**

**class A {**

**void sum(int x, int y){**

**System.out.println("Sum of two numbers: " +(x+y));**

**}**

**void sum(int y, int x){**

**System.out.println("Sum of three numbers: " +(x+y));**

**}**

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

**A a = new A();**

**a.sum(20, 30);**

**}**

**}**

**Out put is::**

**Sum of two numbers: 50**

**26.what is the output**

**class A {**

**void m1(A a){**

**System.out.println("m1 method in class A");**

**}**

**}**

**class B extends A {**

**public void m1(A a){**

**System.out.println("m1 method in class B");**

**}**

**}**

**public class Test{**

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

**A a = new A();**

**a.m1(a);**

**a.m1(new B());**

**B b = new B();**

**b.m1(null);**

**a = b;**

**a.m1(null);**

**a.m1(new A());**

**}**

**}**

**Output is::**

m1 method in class A

m1 method in class A

m1 method in class B

m1 method in class B

m1 method in class B

**27.What is the output**

**import java.io.FileNotFoundException;**

**import java.io.IOException;**

**import javax.sql.SQLException;**

**public class ExceptionInterviewQuestion\_01 {**

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

**try {**

**test();**

**} catch (IOException e) {**

**e.printStackTrace();**

**} catch (FileNotFoundException e) {**

**e.printStackTrace();**

**} catch (SQLException e) {\**

**e.printStackTrace();**

**}**

**}**

**public static void test() throws IOException, SQLException, FileNotFoundException{**

**System.out.println("Inside test() method");**

**}**

**}**

**Out put is::**

Inside test() method

**28.what is the output**

**public class TestException3 {**

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

**try{**

**bar();**

**}catch(NullPointerException e){**

**e.printStackTrace();**

**}catch(Exception e){**

**e.printStackTrace();**

**}**

**foo();**

**}**

**public static void bar(){**

**}**

**public static void foo() throws NullPointerException{**

**}**

**}**

**Out put is:: nothing get printed.**