**Q.NO.1.Define the terms class and object and explain about access specifiers**

**Class:**

A class is a group of objects which have common properties. It is a template or blueprint from which objects are created. It is a logical entity. It can't be physical. A class in Java can contain: Fields, Methods, Constructors, Blocks, Nested class and interface

Syntax:

class <class\_name>

{

member variable(s);

member method(s);

}

**Object:** An object is an instance of a class. A class is a template or blueprint from which objects are created. So, an object is the instance(result) of a class.

class\_name objname=new class\_name();

**There are four types of Java access Specifiers:**

1. Private: The access level of a private modifier is only within the class. It cannot be accessed from outside the class.

2. Default: The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default

3. Protected: The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.

4. Public: The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

**Q.No. 2 Draw the class diagram for the following and develop a java program that reads the number of kilograms of type double, through command line arguments, converts it to pounds, displays the result. (HINT : 1 kg = 2.2046 pounds)**

**Class Diagram**

|  |
| --- |
| P1 |

|  |
| --- |
| <<+ kilotopound>> |
| +res:double |
| +main(args:String[]):void  +public kilotopou(d1:double) |

**Program :**

package p1;

public class kilotopound {

public static double res;

public static void kilotopou(double d1)

{

res=d1\*2.2046;

System.out.println("Pounds value "+res);

}

public static void main(String[] args) {

double d1=Double.parseDouble(args[0]);//reading form command line argument.

kilotopou(d1);

}

}

**Q.NO.3 Modularize to class level the following task and draw the class diagram. Develop a static method distance( ) in class geometry which accepts x1,y1,x2,y2 representing the coordinates of two points of type double and then returns the distance between them and call the method from main () n demo class. Assume 4 inputs from command line.(HINT:distance=((x2-x1)^2-(y2-y1)^2)^1/2).**

import java.util.Scanner;

public class Geometry

{

static double x1;

static double x2;

static double y1;

static double y2;

public static void main(String[] args)

{

double dis;

Scanner sc=new Scanner(System.in);

System.out.println("Enter x1 ");

x1=sc.nextInt();

System.out.println("enter y1");

y1=sc.nextInt();

System.out.println("enter x2");

x2=sc.nextInt();

System.out.println("enter y2");

y2=sc.nextInt();

dis=Math.sqrt((x2-x1)(x2-x1) + (y2-y1)(y2-y1));

System.out.println("distance between two point is "+dis);

}

}

**4) Develop a class ProductSales with a static array ‘sales’ of size of 12 where 0th index stores the sales in January, 1st index stores sales in February…so on 11th index stores sales in December. Create a static findMaximumSalesMonth( ) which prints the month in which maximum sales are done, and test this above function main( ) from the same class.**

package q4;

public class ProductSales {

public static int sales[] = {12,3,11,42,53,33,110,543,23,12,10,33};

public static void findMaximumSalesMonth() {

int i,maxIndex=0;

for(i=1;i<12;i++)

if(sales[i] > sales[maxIndex])

maxIndex=i;

if(maxIndex==0)

System.out.println("January");

else if(maxIndex==1)

System.out.println("February");

else if(maxIndex==2)

System.out.println("March");

else if(maxIndex==3)

System.out.println("April");

else if(maxIndex==4)

System.out.println("May");

else if(maxIndex==5)

System.out.println("June");

else if(maxIndex==6)

System.out.println("July");

else if(maxIndex==7)

System.out.println("August");

else if(maxIndex==8)

System.out.println("September");

else if(maxIndex==9)

System.out.println("October");

else if(maxIndex==10)

System.out.println("Novenber");

else if(maxIndex==11)

System.out.println("December");

}

public static void main(String[] args) {

findMaximumSalesMonth();

}

}

**5) Develop a class Utility with three overloaded static methods with the method name findSmallest( ). The first method will take two arguments and find the smallest among them and return that value. The second method will take three arugments and find smallest among them. The third method will accept an array and find the smallest value in the array and ant return it. Call these three overloaded methods and test their functionality from the main( ) of the Demo class which belong to another package.**

package pack3;

public class Utility {

public static int findSmallest(int a,int b)

{

if(a<b)

{

return a;

}

else

{

return(b);

}

}

public static int findSmallest(int a,int b,int c)

{

if(a<b&&a<c)

{

return(a);

}

else if(b<a&&b<c)

{

return(b);

}

else

{

return(c);

}

}

public static int findSmallest(int[] a)

{

int i,min=a[0];

for(i=1;i<a.length;i++)

{

if(a[i]<min)

{

min=a[i];

}

}

return(min);

}

}

package pack4;

import java.util.Scanner;

import pack3.Utility;

public class Demo

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter two values:");

System.out.println(Utility.findSmallest(sc.nextInt( ),

sc.nextInt( )));

System.out.println("Enter three values:");

System.out.println(Utility.findSmallest(sc.nextInt( ),

sc.nextInt( ),sc.nextInt( )));

int a[]=new int[5];

System.out.println("Enter five elements:");

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

{

a[i]=sc.nextInt();

}

System.out.println(Utility.findSmallest(a));

sc.close();

}

}

**6) Some websites impose certain rules for passwords. Suppose the password rules are as follows: a. A password must have at least eight characters. b. A password cannot have space and must have at least 1 alphabet and digit. Modularize the design to class & package levels and draw the class diagrams. Write main( ) method accepts password through command line arguments and a static method isValidPassword( ) that expects a String argument and returns Boolean type after validating the password.**

**Class Level**

package pack1;

public class Password

{

String pass;

public static boolean isValidPassword(String p)

{

if((p.length( )>=8)&&p.matches("[a-zA-Z0-9]\*"))

{

return true;

}

else

{

return false;

}

}

}

package pack1;

public class PasswordRule

{

public static void main(String[] args)

{

boolean res=Password.isValidPassword(args[0]);

if(res==true)

{

System.out.println("password is valid");

}

else

{

System.out.println("password is invalid");

}

}

}

**Package Level**

package pack2;

import pack1.Password;

public class PasswordRulePackage {

public static void main(String[] args)

{

boolean res=Password.isValidPassword(args[0]);

if(res==true)

{

System.out.println("password is valid");

}

else

{

System.out.println("password is invalid");

}

}

}

**7) Develop class diagram and logic as per the following specifications. Develop a class Employee with ID, firstname, middlename and lastname as private attributes. Code the method with hasSameName(Employee e):boolean that returns true if all firstname, middlename and lastname match. otherwise return false. Develop main method that creates an object and test the functionality of the above class**.

-id:int

-firstname:String

-middlename:String

-lastname:String

+hasSameName(Employee:e):boolean

+main(args:String):void

**Employee**

**Solution:**

package pack1;

public class Employee {

private int id;

private String firstname;

private String middlename;

private String lastname;

Employee(int i, String f,String m,String l)

{

id = i;

firstname = f;

middlename = m;

lastname = l;

}

public String getFirstName( ) { return firstname; }

public String getLastName( ) { return lastname; }

public String getMiddleName( ) { return middlename; }

public boolean hasSameName(Empolyee e)

{

if(e.getFirstName( ).equals(firstname) &&

e.getLastName( ).equals(lastname) &&

e.getMiddleName( ).equals(middlename))

{ return true; }

else

{ return false; }

}

public static void main(String[] args)

{

Employee obj1 = new Employee (701,"Ram", "Gopal",

"Tummala");

Employee obj2 = new Empolyee(702, "Lakshman", "kumar",

"Tummala");

if(obj1.hasSameName(obj2))

System.out.println("Names are same");

else

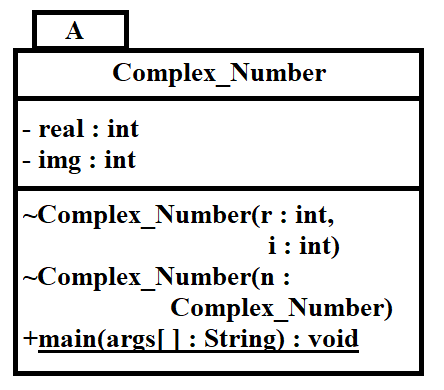
System.out.println("Names are not same");

}

}

**Q.No.8. Draw the class diagram for the following and develop the logic. Define a class complex with real and img as its private attributes. Code a constructor that accepts object of the type complex as an argument and copies the value to the invoking instance.**

**Class Diagram**

****

**Program**

package A;

public class Complex\_Number {

private int real;

private int img;

Complex\_Number(int r,int I)

{

real=r;

img=I;

}

Complex\_Number(Complex\_Number n)

{

real=n.real;

img=n.img;

}

public static void main(String[] args) {

Complex\_Number n=new Complex\_Number(5,6);

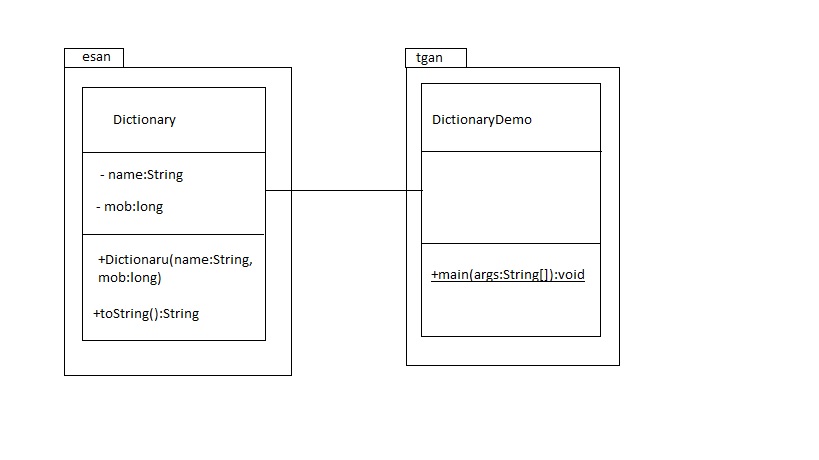
System.out.println("Real number is "+n.real);

System.out.println("Imaginary number is "+n.img);

}

}

**09. Draw the class diagram modularized to package level and develop a class “Dictionary” that contains the following information (a) Name of a person (b) mobile number (both as private attributes). The main() demo class must be able to store details of 10 person and display a menu with the following options. 1. Add new person information 2. Display all data.**



package esan;

publicclass Dictionary {

private String name;

privatelong mob;

public Dictionary(String name,long mob){

this.name=name;

this.mob=mob;

}

public String toString() {

return "Name="+name+" Mobile No="+mob+" \n";

}

}

package tgan;

import java.util.Scanner;

import esan.Dictionary;

publicclass DictionaryDemo {

publicstaticvoid main(String[] args) {

Dictionary[] d=new Dictionary[10];

Scanner sc=new Scanner(System.in);

int size=0;

boolean repeat=true;

while(repeat) {

System.out.println("1. addNewPerson\n2.Display\n others exit");

int option=sc.nextInt();

switch(option)

{

case 1:

d[size]=new Dictionary(sc.next(),sc.nextLong());

size++;

break;

case 2:

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

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

break;

default:

repeat=false;

}

}

sc.close();

}

}

**10. Let s1 and s2 be an ArrayList in main( ) of the class Utility which stores a list of integers. Develop the methods for following task.**

**a) search for an element in ArrayList s1**

**b) store all the elements of s1 into s2**

**c) find the smallest value in s2 ArrayList**

package pack1;

import java.util.Scanner;

import java.util.ArrayList;

public class Utility

{

ArrayList<Integer> iobj=new ArrayList<Integer>( );

Scanner scobj = new Scanner(System.in);

public void searchElement( )

{

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

int search = scobj.nextInt( );

for(Integer tmp: iobj)

{

if(tmp.intValue( ) == search)

{

System.out.println("Element is found");

return;

}

}

System.out.println("Element is not found");

}

public void storeAllElements(Utility obj)

{

for(Integer t:obj.iobj)

this.iobj.add(t.intValue( ));

}

public int findMin( )

{

int min = 99999;

for(Integer tmp: iobj)

{

if(tmp.intValue( ) < min)

{

min = tmp.intValue( );

}

}

return min;

}

public static void main(String[] args)

{

Utility uobj1 = new Utility( );

uobj1.iobj.add(10);

uobj1.iobj.add(20);

uobj1.iobj.add(30);

uobj1.iobj.add(40);

uobj1.iobj.add(50);

uobj1.iobj.add(60);

uobj1.searchElement( );

Utility uobj2 = new Utility( );

uobj2.storeAllElements(uobj1);

System.out.println("Minimum element is "+

uobj1.findMin( ));

}

}

**Q.11. Modularize design to package level and develop the code. A vehicle registration portal accepts the following data from Vehicle owners: a) Vehicle Number b) Wheeler (either 2 or 4) c) Owner name. d) Mobile. Vehicle class contains parameterized constructor, toString( ) methods. The Vehicle Demo class has a main ( ) method which reads and stores data of 100 vehicles into an array and displays the menu with following operations: a) Add data b) Display data based on vehicle number.**

**package** PACK1;

**public** **class** Vehicle

{

**private** String vno;

**private** **int** wheeler;

**private** String oname;

**private** **long** mobile;

**public** Vehicle(String vno,**int** wheeler,String oname,**long** mobile)

{

**this**.vno=vno;

**if**(wheeler==2||wheeler==4)

{

**this**.wheeler=wheeler;

}

**else**

{

wheeler=0;

}

**this**.oname=oname;

**this**.mobile=mobile;

}

**public** String getvno()

{

**return** vno;

}

**public** **int** getwheeler()

{

**return** wheeler;

}

**public** String getoname()

{

**return** oname;

}

**public** **long** getmobile()

{

**return** mobile;

}

**public** String toString()

{

String str;

str=String.*format*("vehicle no: %s %n wheeler:%d %n owner name:%s %n mobile :%d",getvno(),getwheeler(),getoname(),getmobile());

**return** str;

}

}

**package** PACK2;

**import** java.util.Scanner;

**import** PACK1.Vehicle;

**public** **class** VehicleDemo

{

// we are defining an array

**private** **static** Vehicle *v*[];

**private** **static** **int** *i*=0;

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

**public** **static** **void** initialize(**int** n)

{

*v*=**new** Vehicle[n];

}

**public** **static** **int** menuselection()

{

System.***out***.println("1.Add vehicle data");

System.***out***.println("2.Display vehicle details based on vehicle no");

System.***out***.println("3.exit");

System.***out***.println("Enter the choice");

// read the choice based on the input in the console

**return** *sc*.nextInt();

}

**public** **static** **void** Addvehicle()

{

System.***out***.println("Enter the details: vehicle no, wheeler , owner and mobile ");

Vehicle temp=**new** Vehicle(*sc*.next(),*sc*.nextInt(),*sc*.next(),*sc*.nextLong());

//

*v*[*i*++]=temp;

}

**public** **static** **void** searchbasedonvehicleno()

{

System.***out***.println("Enter the vehicle no");

String search=*sc*.next();

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

{

//locate the specific object get the vehicle no ,

//compare the vehicle no equal to the string.

**if**(*v*[*i*].getvno().equals(search))

{

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

**return**;

}

}

System.***out***.println("Vehicle number not found");

}

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

{

**int** ch;

*initialize*(*sc*.nextInt());

**while**(**true**)

{

ch=*menuselection*();

**switch**(ch)

{

**case** 1:

*Addvehicle*();

**break**;

**case** 2:

*searchbasedonvehicleno*();

**break**;

**case** 3:

System.*exit*(0);

}

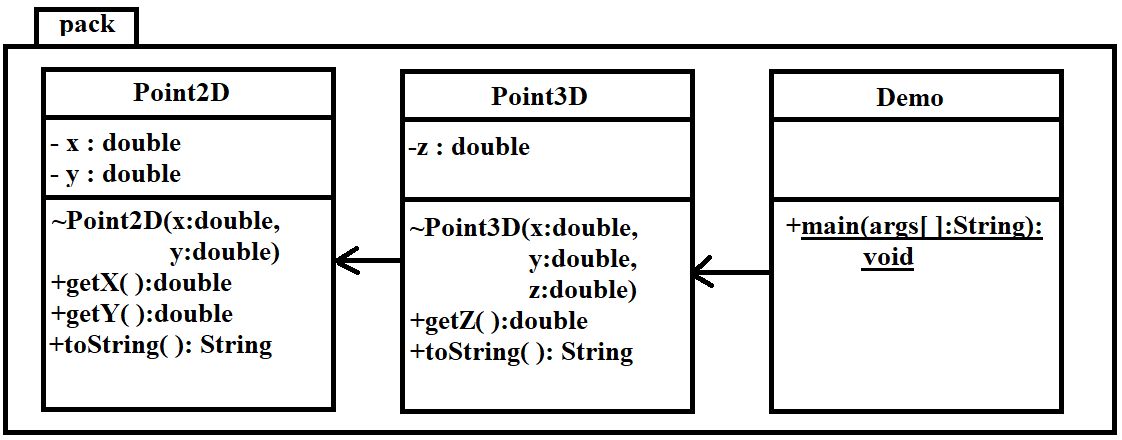
}

}

}

**Q.12. Draw the class diagram and develop a class Point2D with private attributes x and y or type double and a class Point3D which inherits Point2D and has a private attribute Z. Write constructors, setters, getters and toString( ) methods in both classes and a main( ) method of class Demo to create object of Point3D to print all values of x, y and z.**

**Class Diagram**



**Program**

package pack;

class Point2D

{

private double x, y;

Point2D(double x, double y)

{

this.x = x;

this.y = y;

}

public double getX( ) { return x; }

public double getY( ) { return y; }

public String toString( )

{

return "X :"+getX( )+"\nY :"+getY( );

}

}

class Point3D extends Point2D

{

private double z;

Point3D(double x, double y, double z)

{

super(x,y);

this.z = z;

}

public double getZ( ) { return z; }

public String toString( )

{

System.out.println(super.toString( ));

return "Z :"+getZ( );

}

}

public class Demo {

public static void main(String[] args)

{

Point3D pobj = new Point3D(10,20,30);

System.out.println(pobj);

}

}