**Programs for practice**

1. **Static method**

**// Online Java Compiler**

**// Use this editor to write, compile and run your Java code online**

import java.util.\*;

public class Main

{

static void add(int a,int b){

System.out.println(a + b);

}

void add1(int a,int b){

System.out.println(a + b);

}

public static void main(String[] args) {

int a,b;

Scanner jo=new Scanner(System.in);

Main obj=new Main();

a=jo.nextInt();

b=jo.nextInt();

//System.out.println(a);

add(a,b);//static method call

obj.add1(a,b);// instance method call

}

}

import java.util.\*;

public class Main

{

static void sub(int a, int b){

System.out.println(a-b);

}

void sub1(int a, int b){

System.out.println(a-b);

}

public Integer mul(Integer a, Integer b){

return(a\*b);

}

public static void main(String[] args) {

Scanner r=new Scanner(System.in); // Scanner--> class name r --> object reference new--> keyword

Main m=new Main(); // object refering Main class

int a=r.nextInt();

int b=r.nextInt();

Integer x= Integer.valueOf(a);

Integer y= Integer.valueOf(b);

sub(a,b); // static method call

m.sub1(a,b); // Instance method call

System.out.println(m.mul(x,y));

System.out.println(a+b);

}

}

**2. 2D arrays**

// Online Java Compiler

// Use this editor to write, compile and run your Java code online

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc= new Scanner(System.in);

int[][] arr1 = new int[4][4];

int i,j;

for(i=0;i<4;i++)

{

for(j=0;j<4;j++)

{

arr1[i][j]=sc.nextInt();

}

}

for(i=0;i<4;i++)

{

for(j=0;j<4;j++)

{

System.out.print(arr1[i][j]+" ");

}

System.out.print("\n");

}

}

}

**3.**  **Encapsulation, Abstract class and interface program**

interface A{

void a(); //by default, public and abstract

void c();

}

//Creating abstract class that provides the implementation of one method of A interface

abstract class B implements A{

public void c(){System.out.println("I am C ");}

}

//Creating subclass of abstract class, now we need to provide the implementation of rest of the methods

class M extends B{

public void a(){System.out.println("I am a\n");}

}

//Creating a test class that calls the methods of A interface

class Test5{

public static void main(String args[]){

A a=new M();

a.a();

a.c();

}}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Interface :**

interface Program1

{

public void method1();

}

class MyProgram1 implements Program1

{

public void method1()

{ System.out.println("hello this is overridden method.."); }

}

**Abstract Class :**

abstract class ClassOne

{

abstract public void method();

public void method1()

{ System.out.println("Method"); }

}

class ClassTwo extends ClassOne

{

public void method()

{ System.out.println("Method 2"); }

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Encapsulation**

class Area {

private int length=10;

private int breadth=20;

// method to calculate area

public void getArea() {

int area = length \* breadth;

System.out.println("\_\_Area\_\_: " + area);

}

}

class Main {

public static void main(String[] args)

{

Area rectangle = new Area();

//System.out.println(rectangle.length);

rectangle.getArea();

}

}

**4. INHERITANCE**

**SINGLE INHERITANCE:**

class Bike

{

void printBike()

{

System.out.println("Bike class");

}

}

class Car extends Bike

{

void printCar()

{

System.out.println("Car class");

}

}

class Main

{

public static void main(String args[]){

Bike b = new Bike();

b.printBike();

Car c = new Car();

c.printCar();

c.printBike();

Bike b1 = new Car();

b1.printBike();

}

}

**MULTILEVEL INHERITANCE:**

class Bike

{

private int x = 1;

protected int y = 2;

public int z = 3;

void printBike()

{

System.out.println("Bike class");

}

}

class Car extends Bike

{

void printCar()

{

System.out.println("Car class");

}

}

class Plane extends Car

{

void printPlane()

{

System.out.println("Plane class");

}

}

class Main

{

public static void main(String args[]){

Bike b = new Bike();

b.printBike();

System.out.println(b.z);

Car c = new Car();

c.printCar();

c.printBike();

Plane p = new Plane();

p.printPlane();

p.printCar();

p.printBike();

System.out.println(p.y);

Bike b1 = new Car();

b1.printBike();

}

}

**HIERARCHICAL INHERITANCE:**

class A {

public void print\_A() { System.out.println("Class A"); }

}

class B extends A {

public void print\_B() { System.out.println("Class B"); }

}

class C extends A {

public void print\_C() { System.out.println("Class C"); }

}

class Main {

public static void main(String[] args)

{

B b = new B();

b.print\_A();

b.print\_B();

C c = new C();

c.print\_A();

c.print\_C();

}

}

**5. POLYMORPHISM**

**COMPILE TIME POLYMORPHISM- METHOD OVERLOADING**

class Shapes {

void shape(int r) {

System.out.println("Area is: "+ 3.14\*r\*r+ " sq.cm");

}

void shape(int l, int b) {

System.out.println("Area is: "+ l\*b+ " sq.cm");

}

void shape(float s) {

System.out.println("Area is: "+ s\*s+ " sq.cm");

}

}

class Main {

public static void main(String[] args) {

Shapes sp = new Shapes(); // Create a Animal object

sp.shape(2.5f);

sp.shape(4);

sp.shape(10,15);

}

}

**RUNTIME POLYMORPHISM- METHOD OVERRIDING**

class Animal {

public void animalSound() {

System.out.println("The animal makes a sound");

}

}

class Pig extends Animal {

public void animalSound() {

System.out.println("The pig says: wee wee");

}

}

class Dog extends Pig {

public void animalSound() {

System.out.println("The dog says: bow wow");

}

}

class Main {

public static void main(String[] args) {

Animal myAnimal = new Animal(); // Create an Animal object

Pig myPig = new Pig(); // Create a Pig object

Dog myDog = new Dog(); // Create a Dog object

myAnimal.animalSound();

myPig.animalSound();

myDog.animalSound();

} }

**6. 1D ARRAYS**

**/\* This is an example of 1D Array getting input of 4 numbers and outputting the same\*/**

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc= new Scanner(System.in);

int[] arr1 = new int[4];

int i,j;

for(i=0;i<4;i++)

{

arr1[i]=sc.nextInt();

}

for(i=0;i<4;i++)

{

System.out.print(arr1[i]+" ");

}

}

}

**7. Strings**

**i)Assign string value to variable and print the string**

//Basic string program

import java.util.\*;

public class Main {

public static void main(String[] args) {

String first =”Joy with Java”;

System.out.println(first);

}

}

**ii) Getting string input from user**

**Type 1:**

**Using nextLine():**

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc= new Scanner(System.in); // System.in is a standard input stream

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

String str= sc.nextLine(); //reads string

System.out.print("You have entered: "+str);

}

}

**Type 2:**

**Using next():**

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc= new Scanner(System.in); // System.in is a standard input stream

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

String str= sc.next(); //reads string upto first space

System.out.print("You have entered: "+str);

}

}