# Java Lab Code Files

## Java-Lab-main/1 - Total, avg with functions and this keyword/1 - Total,Avg with Functions/Sum.java

```
public class Sum {
    public static void tot(int a, int b) {
        int total = a + b;
        int avg = total / 2;
        System.out.println("THE SUM OF TWO NUMBER IS " + total);
        System.out.println("THE AVERAGE OF TWO NUMBER IS " + avg);
    }

    public static void main(String args[]) {
        tot(56, 90);
    }
}
```

# Java Lab Code Files

## Java-Lab-main/1 - Total, avg with functions and this keyword/2 - Total,Avg with This Keyword/ThisEx.java

```
class ThisEx {
  int a, b;

  public void ThisEx1(int a, int b) {
    this.a = a;
    this.b = b;
    int c = a + b;
    int avg = c / 2;
    System.out.println("THE SUM OF TWO NUMBER IS " + c);
    System.out.println("THE AVERAGE OF TWO NUMBER IS " + avg);
  }

  public static void main(String args[]) {
    int x, y;
    x = Integer.parseInt(args[0]);
    y = Integer.parseInt(args[1]);

    ThisEx S = new ThisEx();
    S.ThisEx1(x, y);
  }
}
```

# Java Lab Code Files

## Java-Lab-main/10 - Packages/Calc.java

```
import calculator.DoubleCalculator;

public class Calc
{
 public static void main(String args[])
 {
  DoubleCalculator c1=new DoubleCalculator();
  double r;
  r= c1.add(10.0,20.0);
  System.out.println("sum = "+r);
  r= c1.sub(10.0,20.0);
  System.out.println("diff = "+r);
  r= c1.mul(10.0,20.0);
  System.out.println("product = "+r);
  r= c1.div(10.0,20.0);
  System.out.println("div = " +r);
 }
}
```

# Java Lab Code Files

## Java-Lab-main/10 - Packages/calculator/DoubleCalculator.java

```
package calculator;

public class DoubleCalculator
{
 public double add(double a, double b)
 {
  return a+b;
 }
 public double sub(double a, double b)
 {
  return a-b;
 }
 public double mul(double a, double b)
 {
  return a*b;
 }
 public double div(double a, double b)
 {
  return a/b;
 }
}
```

# Java Lab Code Files

## Java-Lab-main/11 - Exception Handling using Multiple Catch Blocks/ExceptionDemo.java

```java
import java.util.*;

public class ExceptionDemo {
    public static void main(String args[]) {
        int a, b, c;
        Scanner in = new Scanner(System.in);

        System.out.print("\nPLEASE ENTER TWO NUMBER FOR DIVISION : ");
        a = in.nextInt();
        b = in.nextInt();
        try {
            System.out.println("I AM IN TRY BLOCK 1");
            c = a / b;
            System.out.println("\nDIVISION OF TWO NUMBER IS : " + c);
        } catch (InputMismatchException e) {
            System.out.println("\nI AM IN CATCH BLOCK 1");
            System.out.println("\n CHECK INPUT \n");
            System.out.println(e);
        } catch (ArithmeticException e) {
            System.out.println("\nI AM IN CATCH BLOCK 1");
            System.out.println("\nDIVISION BY ZERO IS NOT POSSIBLE\n");
            System.out.println(e);
        }
        System.out.println("\nEND OF ARITHMETIC EXCEPTION CONCEPT");
        in.close();
    }
}
```

# Java Lab Code Files

## Java-Lab-main/12 - User Defined Exceptions/CustomException.java

```
class AgeException extends Exception
{
 AgeException(String s)
 {
  super(s);
 }
}
public class CustomException
{
 static void validate(int age) throws AgeException
 {
  if(age<18)
   throw new AgeException("cant vote");
  else
   System.out.println("can be voted");
 }
 public static void main(String args[]) throws AgeException
 {
  validate(5);

 }
}
```

# Java Lab Code Files

## Java-Lab-main/13 - Multithreading and Multitasking/SleepTest.java

```
public class SleepTest extends Thread {
    public void run() {
        for (int i = 1; i < 5; i++) {
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
            System.out.println(i);
        }
    }

    public static void main(String args[]) {
        SleepTest t1 = new SleepTest();
        SleepTest2 t2 = new SleepTest2();
        t1.start();
        t2.start();
    }
}

class SleepTest2 extends Thread {
    public void run() {
        for (int i = 5; i < 10; i++) {
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
            System.out.println(i);
        }
    }
}
```

# Java Lab Code Files

## Java-Lab-main/14 - Thread Priorities (!Internal)/TestPriority1.java

```
class TestPriority1 extends Thread {
    public void run() {
        System.out.println("running thread name is:" + Thread.currentThread().getName());
        System.out.println("running thread priority is:" + Thread.currentThread().getPriority());
    }

    public static void main(String args[]) {
        TestPriority1 m1 = new TestPriority1();
        TestPriority1 m2 = new TestPriority1();
        m1.setPriority(Thread.MIN_PRIORITY);
        m2.setPriority(Thread.MAX_PRIORITY);
        m1.start();
        m2.start();
    }
}
```

## Java-Lab-main/15 - Synchronization/SyncThread.java

```
class Disp
{
 synchronized void disp1(int n)
 {
  for(int i=1;i<=n;i++)
  {
  System.out.println(i);
  try
  {
 Thread.sleep(500);

  } catch (Exception e) {
  }
  }
 }
}
public class SyncThread extends Thread
{
 Disp d;
 SyncThread(Disp d)
 {
  this.d=d;
 }
 public void run()
 {
  d.disp1(5);
 }
 public static void main(String args[])
 {
  Disp d1=new Disp();
  SyncThread t1=new SyncThread(d1);
  SyncThread t2=new SyncThread(d1);

  t1.start();
  t2.start();
 }
}
```

## Java-Lab-main/2 - Classes and Objects/TestStudent.java

```
class Student {
    int rollno;
    String name;

    void insertRecord(int r, String n) {
        rollno = r;
        name = n;
    }

    void displayInformation() {
        System.out.println(rollno + " " + name);
    }
}

class TestStudent {
    public static void main(String args[]) {
        Student s1 = new Student();
        Student s2 = new Student();
        s1.insertRecord(111, "Karan");
        s2.insertRecord(222, "Aryan");
        s1.displayInformation();
        s2.displayInformation();
    }
}
```

## Java-Lab-main/3 - Method and Constructor Overloading/Student.java

```
public class Student {
    int id;
    String name;

    Student() {
        System.out.println("this a default constructor");
    }

    Student(int i, String n) {
        id = i;
        name = n;
    }

    void Teacher() {
        System.out.println("this a method overloading");

    }

    void Teacher(int a, int b) {
        int i, k;
        i = a;
        k = b;
        System.out.println(i + k);
    }

    public static void main(String[] args) {
        Student s = new Student();
        System.out.println("\nDefault Constructor values: \n");
        System.out.println("Student Id : " + s.id + "\nStudent Name : " + s.name);
        System.out.println("\nParameterized Constructor values: \n");
        Student student = new Student(10, "David");
        System.out.println("Student Id : " + student.id + "\nStudent Name : " + student.name);
        Student T = new Student();
        T.Teacher();
        T.Teacher(5, 6);
    }
}
```

# Java Lab Code Files

## Java-Lab-main/4 - Parameter Passing Objects (!Internal)/Javaapp.java

```java
class Data {
    int data1;
    int data2;
}

class SetData {
    void setData(Data da, int d1, int d2) {
        da.data1 = d1;
        da.data2 = d2;
    }

    void getData(Data da) {
        System.out.println("data1 : " + da.data1);
        System.out.println("data2 : " + da.data2);
    }
}

public class Javaapp {
    public static void main(String[] args) {
        Data da = new Data();
        SetData sd = new SetData();
        sd.setData(da, 50, 100);
        sd.getData(da);
    }
}
```

# Java Lab Code Files

## Java-Lab-main/5 - Array Manipulation (!Internal)/ArrayInputExample1.java

```java
import java.util.Scanner;

public class ArrayInputExample1 {
    public static void main(String[] args) {
        int n;
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements you want to store: ");
        n = sc.nextInt();
        int[] array = new int[n];
        System.out.println("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            array[i] = sc.nextInt();
        }
        System.out.println("Array elements are: ");
        for (int i = 0; i < n; i++) {
            System.out.println(array[i]);
        }
        sc.close();
    }
}
```

## Java-Lab-main/6 - Inheritances/Hierarchical Inheritance/Main.java

```
class BaseClass {
    int parentNum = 10;
}

class SubClass1 extends BaseClass {
    int childNum1 = 1;
}

class SubClass2 extends BaseClass {
    int childNum2 = 2;
}

class SubClass3 extends BaseClass {
    int childNum3 = 3;
}

public class Main {
    public static void main(String args[]) {
        SubClass1 childObj1 = new SubClass1();
        SubClass2 childObj2 = new SubClass2();
        SubClass3 childObj3 = new SubClass3();
        System.out.println("parentNum * childNum1 = " + childObj1.parentNum *
childObj1.childNum1);
        System.out.println("parentNum * childNum2 = " + childObj2.parentNum *
childObj2.childNum2);
        System.out.println("parentNum * childNum3 = " + childObj3.parentNum *
childObj3.childNum3);
    }
}
```

# Java Lab Code Files

## Java-Lab-main/6 - Inheritances/Multilevel Inheritance/Z.java

```java
class X {
    public void methodX() {
        System.out.println("Class X method");
    }
}

class Y extends X {
    public void methodY() {
        System.out.println("class Y method");
    }
}

class Z extends Y {
    public void methodZ() {
        System.out.println("class Z method");
    }

    public static void main(String args[]) {
        Z obj = new Z();
        obj.methodX(); // calling grand parent class method
        obj.methodY(); // calling parent class method
        obj.methodZ(); // calling local method
    }
}
```

## Java-Lab-main/6 - Inheritances/Single Inheritance/B.java

```
class A
{
   public void methodA()
   {
     System.out.println("Base class method");
   }
}
class B extends A
{
   public void methodB()
   {
     System.out.println("Child class method");
   }
   public static void main(String args[])
   {
     B obj = new B();
     obj.methodA();
     obj.methodB();
  }
}
```

# Java Lab Code Files

## Java-Lab-main/7 - Method Overriding/Bike2.java

```
class Vehicle {
    void run() {
        System.out.println("Vehicle is running");
    }
}

class Bike2 extends Vehicle {
    void run() {
        System.out.println("Bike is running safely");
    }

    public static void main(String args[]) {
        Bike2 obj = new Bike2();
        obj.run();
    }
}
```

## Java-Lab-main/8 - Polymorphism/St1.java

```
class student1
{
int rollno;
String name, branch;
void display()
{
System.out.println("SUPER CLASS DISPLAY");
 System.out.println("HELLO");
}
}
class studentdemo extends student1
{
void display()
{
System.out.println("SUB CLASS DISPLAY");
System.out.println("WELCOME");
}
}
class St1
{
public static void main (String a[])
{
student1 s;
student1 s1= new student1();
studentdemo s2= new studentdemo();
s=s1;
s.display();
s=s2;
s.display();
}
}
```

## Java-Lab-main/9 - Final Keyword/Final Class/Honda1.java

```java
final class Bike {
}

class Honda1 extends Bike// error can not be inherited
{
    void run() {
        System.out.println("running safely with 100kmph");
    }

    public static void main(String args[]) {
        Honda1 honda = new Honda1();
        honda.run();
    }
}
```

## Java-Lab-main/9 - Final Keyword/Final Method/Honda.java

```java
class Bike {
    final void run() {
        System.out.println("running");
    }
}

class Honda extends Bike {
    void run() // error
    {
        System.out.println("running safely with 100kmph");
    }

    public static void main(String args[]) {
        Honda honda = new Honda();
        honda.run();
    }
}
```

# Java Lab Code Files

## Java-Lab-main/9 - Final Keyword/Final with variable/FinalVariable.java

```
class Bike9 {
    final int speedlimit = 90;// final variable

    void run() {
        speedlimit = 400; // error
    }
}

class FinalVariable {
    public static void main(String a[]) {
        Bike9 s1 = new Bike9();
        s1.run();
    }
}
```