class WrapperClassTester {

public static void main(String[] args) {

int i = 45;//primitive data int

Integer integer = new Integer(i);// Integer wrapper class instantiation

int i2 = integer.intValue();// unwrapping primitive data int from wrapper object

Integer integer2 = new Integer("23");

// all wrapper class except Character can take String in argument

System.out.println(integer2);

Integer intObj1 = new Integer(25);

Integer intObj2 = new Integer("25");

Integer intObj3 = new Integer(35);

//compareTo demo

System.out.println("Comparing using compareTo Obj1 and Obj2: " + intObj1.compareTo(intObj2));

System.out.println("Comparing using compareTo Obj1 and Obj3: " + intObj1.compareTo(intObj3));

// Equals demo

System.out.println("Comparing using equals Obj1 and Obj2: " + intObj1.equals(intObj2));

System.out.println("Comparing using equals Obj1 and Obj3: " + intObj1.equals(intObj3));

Float f1 = new Float("2.25f");

Float f2 = new Float("20.43f");

Float f3 = new Float(2.25f);

System.out.println("Comparing using compare f1 and f2: " + Float.compare(f1,f2));

System.out.println("Comparing using compare f1 and f3: " + Float.compare(f1,f3));

// Addition of Integer with Float

Float f = intObj1.floatValue() + f1;

System.out.println("Addition of intObj1 and f1: "+ intObj1 + "+" + f1 + "=" + f);

int x = Integer.parseInt("34");

System.out.println(x);

double y = Double.parseDouble("34.7");

System.out.println(y);

}

}



///////////////////////////////////////

class MyDivException extends Exception

{

public MyDivException(String message) {

super(message);// refer to it parent

}

}

class Tester

{

public static void divide(int x, int y) throws MyDivException {

if(y == 0)

throw new MyDivException("The divisor should not be zero");

int z = x/y;

System.out.println(z);

}

public static void main(String[] args)

{

try

{

divide(6,0);

}catch(MyDivException e) {

System.out.println(e.getMessage());

}

}

}



//////////////////////////////////////////////////////

class Record<E> {

private E record;

public void display(E item) {

System.out.println(item);

}

}

class Student {

private int studentId;

private String studentName;

public Student(int studentId,String studentName)

{

this.studentId=studentId;

this.studentName=studentName;

}

public String toString()

{

return "Student: Id = " + studentId + " Name = " + studentName;

}

}

class GenericsDemo {

public static void main(String[] args)

{

Student s1 = new Student(101,"Robert");

Record<Integer> integerRecord = new Record<Integer>(); //integerRecord can be used to display only integers

integerRecord.display(12);

//integerRecord.display(s1); will give an error as we are trying to add a student class object

Record<Student> studentRecord = new Record<>(); //studentRecord can be used to display only Students

studentRecord.display(s1);

//studentRecord.display(15); will give an error as we are trying to add an integer

}

}

