**Java Part-3 Assignment**

Q1.) Take 20 integer inputs from user and print the following:

number of positive numbers

number of negative numbers

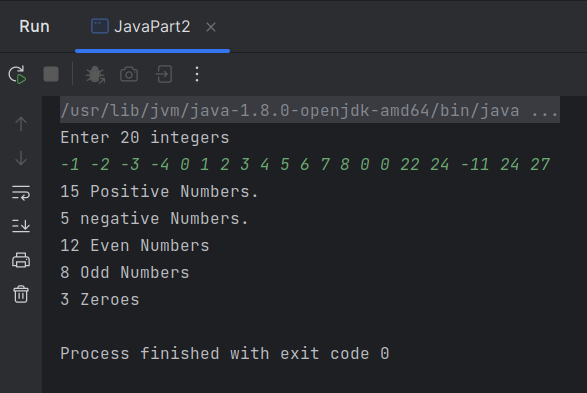
number of odd numbers

number of even numbers

number of 0s.

**Sol.**

import java.util.\*;  
class PositiveOrNegative{  
 int positiveCount=0;  
 int negativeCount=0;  
 public void count(List<Integer> a) {  
 for (Integer i : a) {  
 if (i >= 0) positiveCount++;  
 else negativeCount++;  
 }  
 System.*out*.println(positiveCount+" Positive Numbers.");  
 System.*out*.println(negativeCount+" negative Numbers.");  
 }  
}  
  
class OddOrEven{  
 int oddCount=0;  
 int evenCount=0;  
 public void count(List<Integer> a){  
 for (Integer i : a) {  
 if (i%2==0) evenCount++;  
 else oddCount++;  
 }  
 System.*out*.println(evenCount+" Even Numbers");  
 System.*out*.println(oddCount+" Odd Numbers");  
 }  
}  
  
class ZeroOrNot{  
 int zeroCount=0;  
 public void count(List<Integer> a){  
 for (Integer i : a) {  
 if (i == 0) zeroCount++;  
 }  
 System.*out*.println(zeroCount+" Zeroes");  
 }  
}  
  
public class JavaPart2 {  
 public static void main(String[] args){  
 List<Integer> inputArray=new ArrayList(20);  
 Scanner inp=new Scanner(System.*in*);  
 System.*out*.println("Enter 20 integers");  
  
 for(int i=0;i<20;i++){  
 inputArray.add(inp.nextInt());  
 }  
 PositiveOrNegative pn=new PositiveOrNegative();  
 pn.count(inputArray);  
  
 OddOrEven oe=new OddOrEven();  
 oe.count(inputArray);  
  
 ZeroOrNot zn=new ZeroOrNot();  
 zn.count(inputArray);  
  
  
 }  
}



Q2.) Take an array of 10 elements. Split it into middle and store the elements in two dfferent arrays. E.g.-

INITIAL array :

58 24 13 15 63 9 8 81 1 78

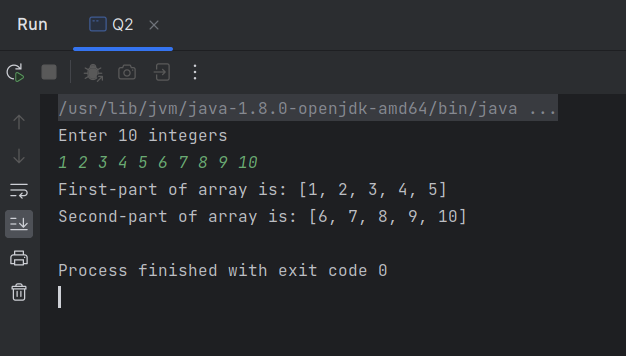
After spliting :

58 24 13 15 63

9 8 81 1 78

**sol.**

import java.util.ArrayList;  
import java.util.List;  
import java.util.Scanner;  
  
  
public class Q2 {  
 public static void main(String[] args){  
 List<Integer> inputArray = new ArrayList(20);  
 List<Integer> splitArray1 = new ArrayList(5);  
 List<Integer> splitArray2 = new ArrayList(5);  
 Scanner inp = new Scanner(System.*in*);  
 System.*out*.println("Enter 10 integers");  
 int inputValue;  
 for (int i = 0; i < 10; i++) {  
 inputValue=inp.nextInt();  
 inputArray.add(inputValue);  
 if(i<5) splitArray1.add(inputValue);  
 else splitArray2.add(inputValue);  
 }  
 System.*out*.println("First-part of array is: "+splitArray1);  
 System.*out*.println("Second-part of array is: "+splitArray2);  
 }  
}



Q3.) 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

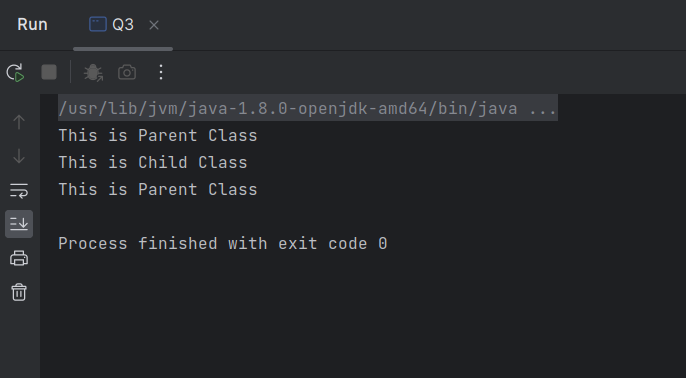
1 - method of parent class by object of parent class

2 - method of child class by object of child class

3 - method of parent class by object of child class

**sol.**

class ParentClass{  
 public void print(){  
 System.*out*.println("This is Parent Class");  
 }  
}  
  
class ChildClass extends ParentClass{  
 ChildClass(){};  
 ChildClass(int a){  
 super.print();  
 }  
 public void print(){  
 System.*out*.println("This is Child Class");  
 }  
}  
public class Q3 {  
 public static void main(String[] args){  
 ParentClass pc=new ParentClass();  
 pc.print();  
 ChildClass cc=new ChildClass();  
 cc.print();  
 ChildClass cc1=new ChildClass(1);  
  
  
 }  
}

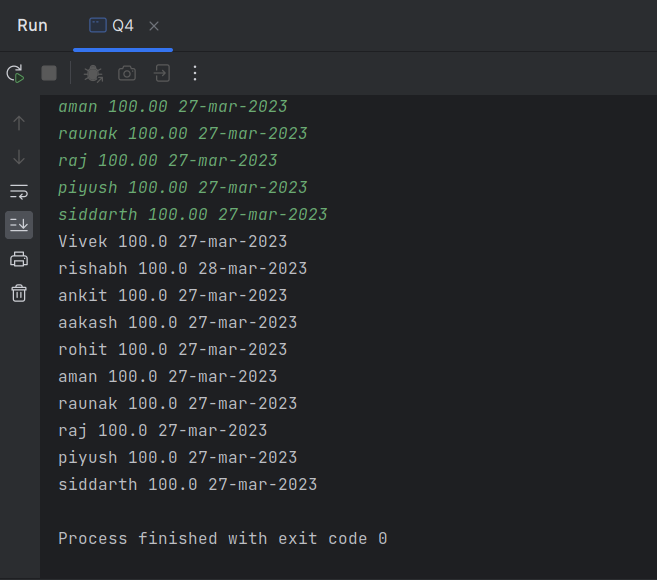


Q4.) Write a program to print the name, salary and date of joining of 10 employees in a company.

Use array of objects.

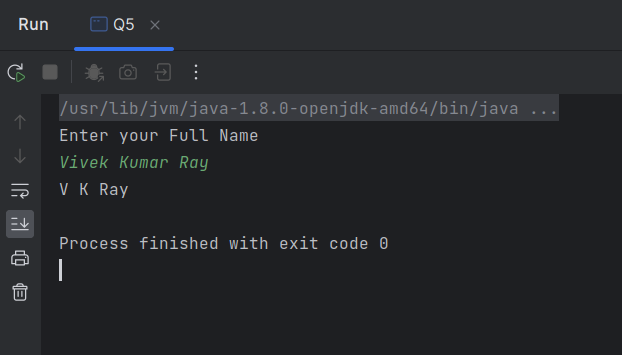
**sol.**

import java.util.ArrayList;  
import java.util.Date;  
import java.util.List;  
import java.util.Scanner;  
class Employee{  
 String name;  
 double salary;  
 String date;  
}  
public class Q4 {  
 public static void main(String[] args) {  
 List<Employee> inputArray = new ArrayList(10);  
 Scanner inp = new Scanner(System.*in*);  
 System.*out*.println("Enter Name, Salary and Date of joining of 10 Employee ");  
  
 for (int i = 0; i < 10; i++) {  
 Employee e=new Employee();  
 e.name=inp.next();  
 e.salary=inp.nextDouble();  
 e.date=inp.next();  
 inputArray.add(e);  
 }  
 for(Employee i:inputArray) {  
 System.*out*.println(i.name+" "+i.salary+" "+i.date);  
 }  
 }  
}



Q5.) Write a program that takes your full name as input and displays the abbreviations of the first and middle names except the last name which is displayed as it is. For example, if your name is Robert Brett Roser, then the output should be R.B.Roser.

**Sol.**

import java.util.Scanner;  
class Name{  
 String firstName;  
 String middleName;  
 String lastName;  
}  
public class Q5 {  
 public static void main(String[] args) {  
 System.*out*.println("Enter your Full Name");  
 Scanner inp = new Scanner(System.*in*);  
 for (int i = 0; i < 1; i++) {  
 Name e = new Name();  
 e.firstName = inp.next();  
 e.middleName = inp.next();  
 e.lastName = inp.next();  
 System.*out*.println(e.firstName.charAt(0)+ " " + e.middleName.charAt(0)+ " " + e.lastName);  
 }  
  
 }  
}

Q6) What is the difference between equals() method and equality operator (==) in Java?

**sol.:-** Both equals() method and the == operator are used to compare two objects in Java.

== is an operator and equals() is method. But == operator compares reference or memory

location of objects in the heap, whether they point to the same location or not.

Q7) What is the difference between StringBuilder and StringBuffer?

* StringBuffer is synchronized i.e. thread safe. It means two threads can't call the methods of StringBuffer simultaneously.

StringBuilder is non-synchronized i.e. not thread safe. It means two threads

can call the methods of StringBuilder simultaneously.

* StringBuffer is less efficient than StringBuilder.

StringBuilder is more efficient than StringBuffer.

Q8) Explain the use of final keyword in variable, method and class.

The final keyword is a non-access modifier used for classes, attributes and methods, which makes them non-changeable (impossible to inherit or override).

Q9) Is it possible that the ‘finally’ block will not be executed? If yes then list the case.

The finally block may not execute if the JVM exits while the try or catch code is being executed. The try block of the writeList method opens a PrintWriter . The program should close that stream before exiting the writeList method.

Q10) What are shallow copy and deep copy in java?

In shallow copy, only fields of the primitive data type are copied while the objects' references are not copied. Deep copy involves the copy of primitive data types as well as to object references.

Q11) What will be the output of below program?

public class TestClass

{

public static void main(String[] args)

{

int a = 30;

int b = 40;

int c = 10;

int expression = (a \* b)/(a - b + c);

System.out.println("Result: " +expression);

}

}

**sol.-** ArithematicException will be shown.

Q12) Why it is always recommended to keep the clean-up activities like closing the I/O resources or DB connections inside a finally block?

**Sol-** Clean up typically means closing db connections, closing IO streams etc. finally{} block is always invoked even if exception has occurred.

Q13) What happens if the below code is executed?

public class Test

{

public static void main(String[] args)

{

int[] list = new int[4];

System.out.println(list[4]);

}

}

**sol.-** ArrayIndexOutOfBoundsException will be there

Q14) How many objects will be created for the following codes:

A.

String str1 = "abc"; //Line1

String str2 = new String("abc"); //Line2

B.

String str1 = "abc"; //Line1

String str2 = "abc"; //Line2

C.

String str1 = new String("abc"); //Line1

String str2 = new String("abc"); //Line2

**sol-** A-3 object

B-1 object

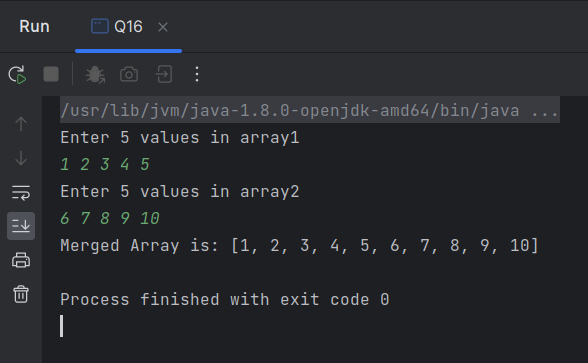
C-4 object

Q15) How do you check whether a String is empty in Java?

**Sol-** isEmpty() Method : The isEmpty() method checks whether a string is empty or not. This method returns true if the string is empty (length() is 0), and false if not.

Q16) Write a program in java to join two arraylists into one arraylist.

**Sol-**

import java.util.ArrayList;  
import java.util.List;  
import java.util.Scanner;  
  
public class Q16 {  
 public static void main(String[] args) {  
 List<Integer> inputArray1 = new ArrayList(5);  
 List<Integer> inputArray2 = new ArrayList(5);  
 List<Integer> mergedArray = new ArrayList(10);  
 Scanner inp = new Scanner(System.*in*);  
  
 System.*out*.println("Enter 5 values in array1");  
 for(int i=0;i<5;i++) {  
 inputArray1.add(inp.nextInt());  
 }  
  
 System.*out*.println("Enter 5 values in array2");  
 for(int i=0;i<5;i++) {  
 inputArray2.add(inp.nextInt());  
 }  
  
 mergedArray.addAll(inputArray1);  
 mergedArray.addAll(inputArray2);  
  
 System.*out*.println("Merged Array is: " + mergedArray);  
  
  
 }  
}

Q17) Which of the following methods can be used to set every element of the List to a specified value?

set()

add()

complete()

fill()

**sol-** fill()

Q18) Which of the following guarantees type-safety in a collection?

Abstract Classes

Interface

Collection

Generics

**sol-** Generics

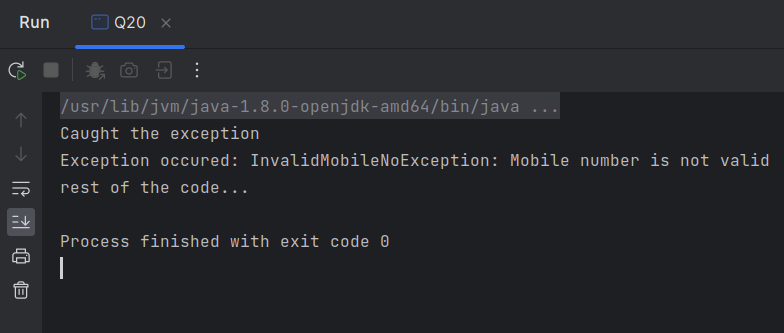
Q19) Differentiate between Comparable and Comparator in the context of Java.

**Sol-** A comparable interface is implemented by all the String class and wrapper classes. Moreover, custom objects use a comparable interface for sorting. On the other hand, the comparator interface is chiefly used to sort the custom objects. You can use it to compare the objects of different classes too.

Q20) Write a Java program to create and throw custom exceptions.

**Sol-**

class InvalidMobileNoException extends Exception{  
 public InvalidMobileNoException(String str){  
 super(str);  
 }  
}  
  
public class Q20 {  
 static void validate (String mobNo) throws InvalidMobileNoException{  
 if(mobNo.length()!=10){  
 throw new InvalidMobileNoException("Mobile number is not valid ");  
  
 }  
 else {  
  
 System.*out*.println("Mobile Number Registered");  
 }  
 }  
  
 public static void main(String[] args) {  
 try  
 {  
 // calling the method  
 *validate*("98875563456");  
 }  
 catch (InvalidMobileNoException ex)  
 {  
 System.*out*.println("Caught the exception");  
  
 // printing the message from InvalidMobileNoException object  
 System.*out*.println("Exception occured: " + ex);  
 }  
  
 System.*out*.println("rest of the code...");  
 }  
 }



Q21) What is the output of the below code?

class IABC{

public static void main(String args[])

{

String obj = "Hello";

String obj1 = "ABC";

String obj2 = "Hello";

System.out.println(obj.equals(obj1) + " " + obj.equals(obj2));

}

}

**Sol-** false true