**Time allowed: 90 Minutes Max. Marks: 40**

**General Instructions:**

* **Follow the instructions given in each section.**
* **Make sure that you attempt the questions in order.**

**SECTION-A (10\*1 mark=10 marks)**

***(All questions are compulsory)***

1) The accuracy and efficiency of a HashMap can be guaranteed with:

a) override equals method

b) override hashCode method

c) None of these

**d) All of these**

2) Which of these interface handle sequences?

a) Set

**b) List**

c) Comparator

d) Collection

3) Which of these interface declares core method that all collections will have?

a) Set

b) Comparator

c) EventListner

**d) Collection**

4) Elements of which of the collection can be traversed using Enumeration?

a) ArrayList

**b) Vector**

c) HashSet

d) TreeMap

5) Which of the following is incorrect statement regarding the use of generics and parameterized types in Java?

a) Generics provide type safety by shifting more type checking responsibilities to the compiler

b) Generics and parameterized types eliminate the need for down casts when using Java Collections

**c) When designing your own collections class (say, a linked list), generics and parameterized types allow you to achieve type safety with just a single class definition as opposed to defining multiple classes**

d) All of the mentioned

6) Why are generics used?

a) Generics make code more fast

b) Generics make code more optimised and readable

**c) Generics add stability to your code by making more of your bugs detectable at compile time**

d) Generics add stability to your code by making more of your bugs detectable at a runtime

7) Which of these method of InputStream is used to read integer representation of next available byte input?

**a) read()**

b) scanf()

c) get()

d) getInteger()

8) Which of these data type is returned by every method of OutputStream?

a) int

b) float

c) byte

**d) none of the mentioned**

9) Parameterized queries can be executed by?

a) ParameterizedStatement

**b) PreparedStatement**

c) CallableStatement and Parameterized Statement

d) All kinds of Statements

10) Which of the following is not a valid statement in JDBC?

a) Statement

b) PreparedStatement

**c) QueryStatement**

d) CallableStatement

**SECTION-B (5\*2 mark=10 marks)**

***(All questions are compulsory)***

11) In ListIterator, after a call to next() or previous(), if a call is made to set(E element) followed by add() or remove() what happens?

a) compilation error

**b) UnsupportedOperationException**

c) ConcurrentModificationException

d) none

12) What will be the output of the following program?

public class GenericsWithObjectsDemo {

public static void main(String args[]) {

GenericsWithObjects<Double> doubleObject = new GenericsWithObjects<Double>(12.0);

doubleObject.print();

GenericsWithObjects<String> stringObject = new GenericsWithObjects<String>("MC");

stringObject.print();

}

}

class GenericsWithObjects<T> {

Object obj;

GenericsWithObjects(Object obj) {

this.obj = obj;

}

void print() {

System.out.println(obj);

}

}

**a)12.0**

**MC**

b)12.0

12.0

c) Compilation Error

d) Runtime Error

13) What will be output for the following code? Note: file is made in c drive

import java.io.\*;

class files

{

public static void main(String args[])

{

File obj = new File(""/java/system"");

System.out.print(obj.canWrite());

System.out.print("" "" + obj.canRead());

}

}

a) true false

b) false true

c) true true

**d) false false**

14) What will be the output of the following Java program?

import java.io.\*;

class filesinputoutput

{

public static void main(String args[])

{

InputStream obj = new FileInputStream("inputoutput.java");

System.out.print(obj.available());

}

}

Note: inputoutput.java is stored in the disk.

a) true

b) false

**c) prints number of bytes in file**

d) prints number of characters in the file

15) For the below Test class which of the following way of object creation is invalid?

class Test<T extends Runnable> { }

class MyThread1 extends Thread { }

class MyThread2 implements Runnable {

public void run(){}

}

a) Test<Runnable> t1 = new Test<Runnable>();

b) Test<MyThread1> t2 = new Test<MyThread1>();

c) Test<MyThread2> t3 = new Test<MyThread2>();

**d) None of these**

**SECTION-C(Coding Question) (2x5 marks=5 marks)**

Q16) Write Java program to find minimum sum of two numbers formed from all digits in a given array.

**Input**:

arr[] = {6, 8, 4, 5, 2, 3}

**Output**:

The required sum is 604

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | arr[] = { 8, 5, 4, 3, 22} | arr[] = {2,3,1} | arr[] = {1,0} |
| **Output** | The required sum is 3570 | The required sum is 15 | The required sum is 1 |

Solution :

**import java.util.PriorityQueue;**

**class JavaProgram**

**{**

**// Returns sum of two numbers formed**

**// from all digits in a[]**

**public static long solve(int[] a)**

**{**

**// min Heap**

**PriorityQueue<Integer> pq = new PriorityQueue<Integer>();**

**// to store the 2 numbers formed by array elements to**

**// minimize the required sum**

**StringBuilder num1 = new StringBuilder();**

**StringBuilder num2 = new StringBuilder();**

**// Adding elements in Priority Queue**

**for (int x : a)**

**pq.add(x);**

**// checking if the priority queue is non empty**

**while (!pq.isEmpty())**

**{**

**num1.append(pq.poll()+ "");**

**if (!pq.isEmpty())**

**num2.append(pq.poll()+ "");**

**}**

**// the required sum calculated**

**long sum = Long.parseLong(num1.toString()) +**

**Long.parseLong(num2.toString());**

**return sum;**

**}**

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

**{**

**int arr[] = {6, 8, 4, 5, 2, 3};**

**System.out.println("The required sum is "+ solve(arr));**

**}**

**}**

Q17) Write a Java program to merge two files into a third file

**Input**:

F1 = "hi", F2 = "hello"

**Output**: F3=”hihello”

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | F1 = "satara", F2 = "—kolhapur--" | F1 = "java", F2 = "is best" | F1 = "apple ", F2 = "a day keeps doctor away" |
| **Output** | F3= satara--kolhapur-- | F3=javais best | apple a day keeps doctor away |

Solution :

**import java.io.\*;**

**public class JavaProgram**

**{**

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

**{**

**// PrintWriter object for file3.txt**

**PrintWriter pw = new PrintWriter("file3.txt");**

**// BufferedReader object for file1.txt**

**BufferedReader br = new BufferedReader(new FileReader("file1.txt"));**

**String line = br.readLine();**

**// loop to copy each line of**

**// file1.txt to file3.txt**

**while (line != null)**

**{**

**pw.println(line);**

**line = br.readLine();**

**}**

**br = new BufferedReader(new FileReader("file2.txt"));**

**line = br.readLine();**

**// loop to copy each line of**

**// file2.txt to file3.txt**

**while(line != null)**

**{**

**pw.println(line);**

**line = br.readLine();**

**}**

**pw.flush();**

**// closing resources**

**br.close();**

**pw.close();**

**System.out.println("Merged file1.txt and file2.txt into file3.txt");**

**}**

**}**

**SECTION-D (Coding Question)(1x10 mark=10 mark)**

Q18) Write Java Program to find min and max using Generic interfaces.

**Input**: arr[] = { 3, 6, 2, 8, 6 }

**Output**:

Minimum value: 2

Maximum value: 8

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | arr[] = { 73, 6, 82, 8, 16 } | arr[] = { 1.4, 9.4, 8.2 } | arr[] = {‘a’,’b’,’k’} |
| **Output** | Minimum value: 6  Maximum value: 82 | Minimum value: 1.4  Maximum value: 9.4 | Minimum value: a  Maximum value: k |

Solution :

**import java.io.\*;**

**// An interface that extends Comparable**

**interface MinMax<T extends Comparable<T> > {**

**T min();**

**T max();**

**}**

**class MyClass<T extends Comparable<T> >**

**implements MinMax<T> {**

**T[] values;**

**MyClass(T[] obj) { values = obj; }**

**public T min()**

**{**

**// 'T' is typename and 'o1' is object\_name**

**T o1 = values[0];**

**for (int i = 1; i < values.length; i++)**

**if (values[i].compareTo(o1) < 0)**

**o1 = values[i];**

**return o1;**

**}**

**// Defining abstract max() method**

**public T max()**

**{**

**// 'T' is typename and 'o1' is object\_name**

**T o1 = values[0];**

**for (int i = 1; i < values.length; i++)**

**if (values[i].compareTo(o1) > 0)**

**o1 = values[i];**

**// Return the maximum element in an array**

**return o1;**

**}**

**}**

**class JavaProgram {**

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

**{**

**Integer arr[] = { 3, 6, 2, 8, 6 };**

**MyClass<Integer> obj1 = new MyClass<Integer>(arr);**

**System.out.println("Minimum value: " + obj1.min());**

**System.out.println("Maximum value: " + obj1.max());**

**}**

**}**