**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) Which of these classes is used to define exceptions?

**a) Exception**

b) Throwable

c) Abstract

d) System

2) Which of these keywords is used by a class to use an interface defined previously?

a) import

b) Import

**c) implements**

d) Implements

3) Which of the following is the correct way of implementing an interface salary by class manager?

a) class manager extends salary {}

**b) class manager implements salary {}**

c) class manager imports salary {}

d) none of the mentioned

4) A java interface can contain ————

a) public static Final Variables only

b) public Abstract methods

c) Abstract methods(unimplemented) and implemented methods both

**d) public static Final Variables and abstract methods both**

5) Which of the following operators is used to generate instance of an exception which can be thrown using throw?

a) thrown

b) alloc

c) malloc

**d) new**

6) Which of the following handles the exception when a catch is not used?

a) finally

b) throw handler

**c) default handler**

d) java run time system

7) Which of these are types of multitasking?

a) Process based

b) Thread based

**c) Process and Thread based**

d) None of the mentioned

8) Which of these statements is incorrect?

a) By multithreading CPU idle time is minimized, and we can take maximum use of it

b) By multitasking CPU idle time is minimized, and we can take maximum use of it

c) Two thread in Java can have the same priority

**d) A thread can exist only in two states, running and blocked**

9) Which among the following best describes the protected specifier?

a) Members are most secure and can’t be used outside class

b) Members are secure but can be used outside the class

**c) Members are secure as private, but can be inherited**

d) Members are secure like private, but can’t be inherited

10) If the protected members are to be made accessible only to the nearest subclass and no further subclasses, which access specifier should be used in inheritance?

**a) The sub class should inherit the parent class privately**

b) The sub class should inherit the parent class as protected

c) The sub class should inherit the parent class as public

d) The sub class can use any access modifier

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

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

11) class A{

public void doA(){

B b = new B();

b.dobB();

System.out.print("doA");

}

}

class B{

public void dobB(){

C c = new C();

c.doC();

System.out.print("doB");

}

}

class C{

public void doC(){

if(true)

throw new NullPointerException();

System.out.print("doC");

}

}

public class Test{

public static void main(String args[]){

try{

A a = new A();

a.doA();

}catch(Exception ex){

System.out.print("error");

}

}

}

a) "doCdoBdoA" is printed

b) "doAdoBdoC" is printed

c) "doBdoAerror" is printed

**d) "error" is printed**

12) Determine output of the following code.

interface A { }

class C { }

class D extends C { }

class B extends D implements A { }

public class Test extends Thread{

public static void main(String[] args){

B b = new B();

if (b instanceof A)

System.out.println("b is an instance of A");

if (b instanceof C)

System.out.println("b is an instance of C");

}

}

a) Nothing.

b) b is an instance of A.

c) b is an instance of C.

**d) b is an instance of A followed by b is an instance of C.**

13) The output of the following fraction of code is

public class Test{

public static void main(String args[]){

String s1 = new String("Hello");

String s2 = new String("Hellow");

System.out.println(s1 = s2);

}

}

a) Hello

**b) Hellow**

c) Compilation error

d) Throws an exception

14) Given the following piece of code:

public class School{

public abstract double numberOfStudent();

}

which of the following statements is true?

a) The keywords public and abstract cannot be used together.

b) The method numberOfStudent() in class School must have a body.

c) You must add a return statement in method numberOfStudent().

**d) Class School must be defined abstract.**

15) What is the output of the following program code?

public class Test{

public static void main(String args[]){

try{

int i;

return;

}

catch(Exception e){

System.out.print("inCatchBlock");

}

finally{

System.out.println("inFinallyBlock");

}

}

}

a) inCatchBlock

b) inCatchBlock inFinallyBlock

**c) inFinallyBlock**

d) The program will return without printing anything

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

Q16) Given two strings 'str1' and 'str2', check if these two strings are isomorphic to each other.Two strings str1 and str2 are called isomorphic if there is a one to one mapping possible for every character of str1 to every character of str2 while preserving the order.

Note: All occurrences of every character in str1 should map to the same character in str2

**Input**:

str1 = aab

str2 = xxy

**Output**: 1

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | str1 = aab  str2 = xxy | str1 = bbc  str2 = mno | str1 = aaa  str2 = ccc |
| **Output** | 1 | 0 | 1 |

Solution :

**import java.util.Arrays;**

**import java.util.\*;**

**public class MyClass {**

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

**String s1 = "abc";**

**String s2 = "xyz";**

**Solution obj = new Solution();**

**boolean a = obj.areIsomorphic(s1,s2);**

**if(a){**

**System.out.println(1);**

**}**

**else{**

**System.out.println(0);**

**}**

**}**

**}**

**class Solution**

**{**

**public static boolean areIsomorphic(String str1,String str2)**

**{**

**char[] map1 = new char[26], map2 = new char[26];**

**for(int idx=0; idx < 26; map1[idx] = '#', map2[idx] = '#', idx++);**

**char[] chArray1 = str1.toCharArray(), chArray2 = str2.toCharArray();**

**if(chArray1.length != chArray2.length) return false;**

**int idxFromChar1, idxFromChar2;**

**for(int idx=0; idx<chArray1.length; idx++)**

**{**

**idxFromChar1 = chArray1[idx]-'a';**

**idxFromChar2 = chArray2[idx]-'a';**

**if(map1[idxFromChar1] == '#' && map2[idxFromChar2] == '#')**

**{**

**map1[idxFromChar1] = chArray2[idx];**

**map2[idxFromChar2] = chArray1[idx];**

**}**

**else if(map1[idxFromChar1] != chArray2[idx] ||**

**map2[idxFromChar2] != chArray1[idx]**

**)**

**return false;**

**}**

**return true;**

**}**

**}**

Q17) Write a java program for reversing the string by swapping string characters.

**Input**: America

**Output**: aciremA

Solution :

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | S = "forever" | S = "samsung" | S = "unicorn" |
| **Output** | reverof | gnusmas | nrocinu |

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

**class Solution {**

**public String reverseString(String s) {**

**byte[] bytes = s.getBytes();**

**int i = 0;**

**int j = s.length() - 1;**

**while (i < j) {**

**byte temp = bytes[i];**

**bytes[i] = bytes[j];**

**bytes[j] = temp;**

**i++;**

**j--;**

**}**

**return new String(bytes);**

**}**

**}**

**public class MyClass**

**{**

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

**{**

**Solution s=new Solution();**

**String r=s.reverseString("America");**

**System.out.println(r);**

**}**

**}**

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

Q18) You have given a non-empty string. This string can consist of lowercase and uppercase english alphabets. Convert the string into an alternating sequence of lowercase and uppercase characters without changing the character at the 0th index.

**Input :** Astrology

**Output :** AsTrOlOgY

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | ninja | Network | MPSC |
| **Output** | nInJa | NeTwOrK | MpSc |

Solution :

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

**import java.util.\*;**

**public class MyClass{**

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

**String S ="Astrology";**

**Solution ob = new Solution();**

**System.out.println(ob.getCrazy(S));**

**}**

**}**

**class Solution{**

**static String getCrazy(String s)**

**{**

**StringBuilder b=new StringBuilder();**

**if(Character.isLowerCase(s.charAt(0)))**

**{**

**for(int i=0;i<s.length();i++)**

**{**

**if(i%2==0)**

**{**

**b.append((Character.toLowerCase(s.charAt(i))));**

**}**

**else**

**{**

**b.append((Character.toUpperCase(s.charAt(i))));**

**}**

**}**

**}**

**else if(Character.isUpperCase(s.charAt(0)))**

**{**

**for(int i=0;i<s.length();i++)**

**{**

**if(i%2==0)**

**{**

**b.append(Character.toUpperCase(s.charAt(i)));**

**}**

**else**

**{**

**b.append((Character.toLowerCase(s.charAt(i))));**

**}**

**}**

**}**

**return b.toString();**

**}**

**}**