Assessment 1

1. What is metaspace and heap memory?

Meta space is a new memory space – starting from the Java 8 version; it has replaced the older PermGen memory space. The most significant difference is how it handles memory allocation. Specifically, this native memory region grows automatically by default. The OpenJDK uses Meta space to store its class metadata. It can contribute a large part to the non-Java-heap memory footprint of a Java VM process. Meta space is memory the VM uses to store class Metadata. Metadata is defined as data about the data. It is the documentation about the information which is required by the users. It is used to analyze data usage and performance of particular data.

Heap Memory is created when JVM starts up and used by the application as long as the application is running. Whenever an object is created it occupies the space in the heap memory while reference of that object creates in the stack. It dynamically handles the memory blocks.

1. Generate multiples of 2 until 20 using recursive function

**package** assessment;

**public** **class** Recursive {

**static** **void** multiplies(**int** a,**int** b) {

**if**(b>10)

**return** ;

System.***out***.println(a\*b);

*multiplies*(a,b+1);

}

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

// **TODO** Auto-generated method stub

**int** a=2;

*multiplies*(a, 1);

}

}

Sol:2

4

6

8

10

12

14

16

18

20

1. Check if two strings are equal or not

**package** com.pack;

**import** java.util.Scanner;

**public** **class** Employee {

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

String s1;

String s2;

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter strings to compare:");

s1=sc.nextLine();

s2=sc.nextLine();

**if**(s1.equals(s2)) {

System.***out***.println("Strings are equals.");

}

**else** {

System.***out***.println("Strings are not equal.");

}

}

}

Sol:-

Enter strings to compare:

hello

hello

Strings are equals.

1. Print the character count in a string say, string s ="helloworld" print h-1, e-1, l-3,o-2

**package** com.pack;

**import** java.util.Scanner;

**import** java.util.HashMap;

**import** java.util.Map;

**public** **class** Employee {

**static** **void** characterCount(String s) {

HashMap<Character, Integer> count = **new** HashMap<Character, Integer>();

**char**[] str = s.toCharArray();

**for**(**char** c : str) {

**if**(count.containsKey(c)) {

count.put(c, count.get(c)+1);

}

**else** {

count.put(c, 1);

}

}

**for**(@SuppressWarnings("rawtypes") Map.Entry entry: count.entrySet()) {

System.***out***.println(entry.getKey() + " " + entry.getValue());

}

}

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

Scanner sc = **new** Scanner(System.***in***);

String s;

System.***out***.println("Enter string:");

s=sc.nextLine();

*characterCount*(s);

sc.close();

}

}

Sol:

Enter string:

helloworld

r 1

d 1

e 1

w 1

h 1

l 3

o 2

1. Why java is platform independent?

Java platform independent means 'write once and run anywhere' or WORA.

It implies that it doesn't matter on what operating system(lets say,Windows OS) the code was written, it could be run on the other operating system(lets say Linux) conveniently and without any issue.

Platform independent means that the compiled code can run on all operating systems. A machine understands code which is in machine level language.So the compiler converts high level language that is human readable language into format of machine level language that machine will understood. So compiler translates the source code from a programming language to a executable code. This executable code is executed by CPU directly or it may be a byte code that is interpreted by a virtual machine.

1. Can we create class as final.

Yes, we can create a class as final. But a class which is declared as final cannot be subclassed or it cannot be override.

1. Consider we have employee class with empid, empname and salary and list of employees get the the highest salary paid employee data
2. Consider a list of duplicate values remove duplicate value and get unique values from the list

**package** com.pack;

**import** java.util.Scanner;

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.List;

**import** java.util.stream.Collectors;

**public** **class** Employee {

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

Scanner sc = **new** Scanner(System.***in***);

ArrayList<Integer> numlist = **new** ArrayList<Integer>();

**int** num;

System.***out***.println("Enter numbers in list:");

**for**(**int** i=0;i<10;i++) {

num=sc.nextInt();

numlist.add(num);

}

System.***out***.println("Before");

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

List<Integer> listwithoutDuplicates = numlist.stream().distinct().collect(Collectors.*toList*());

System.***out***.println("After");

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

sc.close();

}

}

Sol:

Enter numbers in list:

5

5

5

8

5

7

6

3

Before

[5,5,5,8,5,7,6,3]

After

[5,8,7,6,3]

1. Can we write try and finally without catch block what is the use.

Yes, it is possible to use try and finally block without catch block. Because whether the exception occurred in try block, even then finally block will execute.