1. **What is Metaspace and heap memory?**

Metaspace is introduced in Java 8. It by default auto increases its size depending on the underlying OS. It has Native Memory(provided by underlying OS) and has efficient garbage collection.

Heap memory is a part of memory allocated to JVM, which is shared by all executing threads in the application. It is the part of JVM in which all class instances and are allocated. It is created on the Start-up process of JVM. It does not need to be contiguous, and its size can be static or dynamic.

**2.Generate multiples of 2 until 20 using recursive function.**

**package** com.assignment;

**class** GFG {

**static** **void** mul\_table(**int** N, **int** i)

{

**if** (i > 10)

**return** ;

System.***out***.println(N + " \* " + i + " = " + N \* i);

*mul\_table*(N, i + 1);

}

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

{

**int** N = 2;

*mul\_table*(N, 1);

}

}

**3.Check if two strings are equal or not.**

package com.assignment;

**import** java.util.Scanner;

**public** **class** GFG {

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

String str1, str2;

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

System.***out***.println("Enter first String");

str1 = scanner.nextLine();

System.***out***.println("Enter second String");

str2 = scanner.nextLine();

// Comparing two input string

**if** (str1.equals(str2))

System.***out***.print("Equal Strings");

**else**

System.***out***.print("UnEqual Strings");

}

}

**4.Print the character count in a string say**

**string s ="helloworld" print h-1, e-1, l-3,o-2…**

**public** **class** GFG {

**static** **final** **int** ***MAX\_CHAR*** = 256;

**static** **void** getOccuringChar(String str)

{

**int** count[] = **new** **int**[***MAX\_CHAR***];

**int** len = str.length();

**for** (**int** i = 0; i < len; i++)

count[str.charAt(i)]++;

**char** ch[] = **new** **char**[str.length()];

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

ch[i] = str.charAt(i);

**int** find = 0;

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

**if** (str.charAt(i) == ch[j])

find++;

}

**if** (find == 1)

System.***out***.println(

"Number of Occurrence of "

+ str.charAt(i)

+ " is:" + count[str.charAt(i)]);

}

}

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

{

String str = "helloworld";

*getOccuringChar*(str);

}

}

**5. Why is java platform independent?**

Java compiler produces a unique type of code called bytecode unlike c compiler where compiler produces only natively executable code for a particular machine.

When the Java program runs in a particular machine it is sent to java compiler, which converts this code into intermediate code called bytecode. This bytecode is sent to Java virtual machine (JVM) which resides in the RAM of any operating system. JVM recognizes the platform it is on and converts the bytecodes into native machine code. Hence java is called platform independent language.

**6.Can we create class as final?**

A class can be made final by using the final keyword. The final class cannot be inherited and so the final keyword is commonly used with a class to prevent inheritance.

**7.** **considder we have employee class with empid, empname and salary and list of employees get the the highest salary paid employee data.**

package com.pack;

import java.util.\*;

class Emp

{

int salary;

String name;

int id;

Emp(int i,String g,int d)

{

this.salary=i;

this.name=g;

this.id= d;

}

}

class Manhattan

{

public static void main(String []args)

{

ArrayList<Emp> a = new ArrayList<Emp>();

a.add(new Emp(100,"javed",10));

a.add(new Emp(500,"apporva",11));

a.add(new Emp(250,"sumit",12));

a.add(new Emp(100,"itika",13));

a.add(new Emp(90,"latika",14));

a.add(new Emp(67,"jatin",15));

a.add(new Emp(340,"nitin",16));

a.add(new Emp(2300,"Shivani",17));

Iterator<Emp> i = a.iterator();

int maxsalary=0;

String name = null;

int id=0;

if(i.hasNext())

{

Emp e=i.next();

maxsalary=e.salary;

}

Iterator<Emp> i1 = a.iterator();

while(i1.hasNext())

{

Emp e1 = i1.next();

if(maxsalary<=e1.salary)

{

maxsalary=e1.salary;

name=e1.name;

id=e1.id;

}

//System.out.println(maxsalary);

}

System.out.println(" name = " + name + " max salary " + maxsalary+ "id " +id);

}

}

**8.** **consider a list of duplicate values remove duplicate value and get unique values from the list.**

import java.util.ArrayList;

import java.util.LinkedHashSet;

import java.util.Set;

public class GFG {

public static void main(String[] args) {

ArrayList<String> pets = new ArrayList<>();

pets.add("cat");

pets.add("dog");

pets.add("cat");

pets.add("hamster");

System.out.println(pets);

Set<String> hashSet = new LinkedHashSet(pets);

ArrayList<String> removedDuplicates = new ArrayList(hashSet);

System.out.println(removedDuplicates);

}

}

**9. can we write try and finally without catch block what is the use.**

Yes, we can have try without catch block by using finally block. We can use try with finally. As we know finally block always executes even if you have exception or return statement in try block except in case of System.

The finally block always executes when the try block exits. So we can use finally without catch but we must use **try**. The finally block always executes when the try block exits. So you can use finally without catch but you must use try.

​**10.what is java memory model?**

The Java memory model specifies how the Java virtual machine works with the computer's memory (RAM). The Java virtual machine is a model of a whole computer so this model naturally includes a memory model - AKA the Java memory model.

It is very important to understand the Java memory model if we want to design correctly behaving concurrent programs. The Java memory model specifies how and when different threads can see values written to shared variables by other threads, and how to synchronize access to shared variables when necessary.

**11.what is young and old generations?**

**​**The Young Generation is where all new objects are allocated and aged. When the young generation fills up, this causes a minor garbage collection. A young generation full of dead objects is collected very quickly. Some survived objects are aged and eventually move to the old generation.

The Old Generation is used to store long surviving objects. Typically, a threshold is set for young generation object and when that age is met, the object gets moved to the old generation. Eventually the old generation needs to be collected. This event is called a major garbage collection.

**12.what is eden and survivor space?**

* **Eden Space**: The pool from which memory is initially allocated for most objects.
* **Survivor Space**: The pool containing objects that have survived the garbage collection of the Eden space.

**13.What is garbage collector and how it works?**

As long as an object is being referenced, the JVM considers it alive. Once an object is no longer referenced and therefore is not reachable by the application code, the garbage collector removes it and reclaims the unused memory.

**Garbage collection** is the process of managing memory, automatically. It finds the unused objects (that are no longer used by the program) and delete or remove them to free up the memory. The **garbage collection** mechanism uses several GC algorithms. The most popular algorithm that is used is **Mark and Sweep.**