1. What is metaspace and heap memory ?

Metaspace 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.**

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.

1. Why java is platform independent ?

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.

1. 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.

1. Generate multiples of 2 until 20 using recursive function

class multiplication {

    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);

    }

}

1. Check if two strings are equal or not

class EqualString {

   public static void main(String[] args) {

      String str1 = “Hello Java”;

      String str2 = “Hello Java”;

      String str3 = “Bye”;

      boolean returnval1 = str2.equals(str1);

      boolean returnval2 = str2.equals(str3);

      System.out.println("str2 is equal to str1 = " + returnval1);

      System.out.println("str2 is equal to str3 = " + returnval2);

   }

}

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

public class Count {

public static void main(String[] args) {

String str;

int i, length, counter[] = new int[256];

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

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

str = scanner.nextLine();

length = str.length();

for (i = 0; i < length; i++) {

counter[(int) str.charAt(i)]++;

}

for (i = 0; i < 256; i++) {

if (counter[i] != 0) {

System.***out***.println((char) i + " --> " + counter[i]);

}

}

}

}

1. 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. You can use try with finally. As you know finally block always executes even if you have exception or return statement in try block except in case of System.

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

public class EmployeeData {

int emp\_id;

String emp\_name;

double salary;

public EmployeeData(int emp\_id, String emp\_name, double salary)

{

this.emp\_id = emp\_id;

this.emp\_name = emp\_name;

this.salary = salary;

}

public int getId()

{

return emp\_id;

}

public String getName()

{

return emp\_name;

}

public double getSalary()

{

return salary;

}

*@Override*

public String toString()

{

return "Id : "+emp\_id

+", Name : "+emp\_name

+", Salary : "+salary;

}

public static void main(String[] args) {

List<EmployeeData> emp\_list = new ArrayList<EmployeeData>();

emp\_list.add(new EmployeeData(100, "ABC", 100000));

emp\_list.add(new EmployeeData(200, "XYZ", 150000));

emp\_list.add(new EmployeeData(300, "EFG", 200000));

emp\_list.add(new EmployeeData(400, "IJK", 300000));

emp\_list.add(new EmployeeData(500, "LMN", 400000));

Optional<EmployeeData> highestsalaryWrapper = emp\_list.stream().collect(Collectors.*maxBy*(Comparator.*comparingDouble*(EmployeeData::getSalary)));

EmployeeData highestsalary = highestsalaryWrapper.get();

System.***out***.println("ID: "+ highestsalary.getId());

System.***out***.println("Name: "+ highestsalary.getName());

System.***out***.println("Salary: "+ highestsalary.getSalary());

}

}