

1. What is JVM and explain me the Java memory allocation

When a source code is compiled, .class files are created containing byte codes these .class files are interpreted by JVM's to convert them into machine language or assembly language.

The various memories areas of JVM are:

- a. Method area - Class level data is present
- b. Heap area – Objects data is present
- c. Stack area – data stored in stack area is thread safe
- d. PC registers – Has PC registers for threads 1 to n
- e. Native method stacks

2. What is Polymorphism and encapsulation?

Polymorphism – Polymorphism in java is a concept by which we can perform a single action by different ways. Polymorphism means many forms.

There are two types of polymorphism in java: compile time polymorphism and runtime polymorphism. We can perform polymorphism in java by method overloading and method overriding.

Encapsulation - **Encapsulation in Java** is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as single unit. The whole idea behind encapsulation is to hide the implementation details from users. We can create a fully encapsulated class in java by making all the data members of the class private. Now we can use setter and getter methods to set and get the data in it.. This way data can only be accessed by public methods thus making the private fields and their implementation hidden for outside classes. That is why encapsulation is known as **data hiding**.

3. What is method overloading and Method overriding?

Method Overloading – When a class had more than one method with same method name but with different type of arguments or different number of arguments then the method is treated as method overloaded.

Method Overriding - When a inherited super class method is modified in the sub class, then we call it as method is overridden. Through method overriding, we can modify super class method according to requirements of sub class.

The rules of method over ridding is

- a. Return type should be compatible
- b. We can keep the same visibility or increase the visibility but cannot reduce the visibility
- c. The arguments should not be changed. If it is changed then it is called as method overloaded.

4. Why string is Immutable?

When a string object is created, we cannot perform any changes in the object. When an existing object is modified, a new object is created but existing object is not modified. This is called Immutable

5. What is the difference between String and String buffer?

String is immutable as explain above. String bugger is mutable it mean when an object is created and if any changes has to be done we can append with new changes but it will not create a new object hence it is called mutable.

6. What is the difference between array and array list?

Array	Arraylist
Array are of fixed length	Arraylist are of variable length. It changes dynamically
Array support primitive like int, float, Boolean data types and Objects	It does not support primitive data types(Can use autoboxing to change primitive to object) but supports only objects and generics
Array can be multidimensional	It is always single dimensional

7. What is the difference between hash map and Hash table?

Hash Map	Hash table
HashMap is non synchronized and not thread safe	HashTable is thread safe and synchronized
Hashmap allows one null key and any number of null values	Hashtable do not allow null keys and null values in the HashTable object.
Hashmap object values are iterated by using iterator	HashTable is the only class other than vector which uses enumerator to iterate the values of HashTable object
Hash map has initial value of 16 when l reaches some level it changes it to 32	

8. What is a vector in Java?

Vector are implementation class of collection framework.It imeplements List interface.

- The underlying data structure of vector is resizeable array.
- Duplicates are allowed in vector
- Insertion order is preserved
- Null insertion is possible
- Most of the methods present in Vector are synchronized hence vector object is thread safe.
- For adding objects we use addElement(object)
- To remove we use removeElementAt(object)

9. What is set in java?

Set is a child interface of collection. If we want to represent group of individual objects as single entity where duplicates are not allowed and insertion order not preserved then we should go for set.

10. What is an abstract class?

- A class that is declared with abstract keyword, is known as abstract class in java. It can have abstract and non-abstract methods (method with body). We cannot create objects to those classes, which are declared as abstract. But, we can create objects to sub classes of abstract class, provided they must implement abstract methods.
- It is not compulsory that abstract class must have abstract methods. It may or may not have abstract methods. But the class which has at least one abstract method must be declared as abstract.
- You can't create objects to abstract class even though it does not contain any abstract methods.

11. What is an interface?

An interface can have methods and variables just like the class but the methods declared in interface are by default abstract (only method signature, no body). Also, the variables declared in an interface are public, static & final by default.

The class that implements interface must implement all the methods of that interface. Also, java programming language does not support multiple inheritance, using interfaces we can achieve this as a class can implement more than one interfaces, however it cannot extend more than one classes.

12. Why Java is Platform independent?

when you compile java program on one operating system then java compiler generates byte code. you can run that byte code on any operating system which has jvm. jvm is understanding byte code and generating native code for the corresponding operating system. because of byte code and jvm, java is platform independent u can use write once run anywhere.....

13. What are access modifiers? Give me an example?

Access modifiers in java are used to control the visibility of a field, method, class and constructor. There are 4 access modifiers in java. They are : **1). Private 2). Default or Package 3). Protected 4). Public**

Private - Private members of a class whether it is a field or method or constructor cannot be accessed outside the class. Can be inherited to sub class within package

Default or Package - Default members or members with No-Access modifiers are accessed or visible within the package only. It applies to outer classes also.

Protected - Protected member can be used within the package only. Protected Member can be inherited to any sub classes.

14. What are java exceptions? Give me an example

Exceptions are events that occur during the execution of programs that disrupt the normal flow of instructions (e.g. divide by zero, array access out of bound, etc.).

15. What is the difference between throws and throwable?

Throws is a keyword in java which is used in the method signature to indicate that this method may throw mentioned exceptions.

The java.lang.Throwable class is the superclass of all errors and exceptions in the Java language. Only objects that are instances of this class (or one of its subclasses) are thrown by the Java Virtual Machine or can be thrown by the Java throw statement.

16. What is the difference between Error and exception?

Errors	Exceptions
Errors are mostly caused by the environment in which application is running.	Exceptions are mainly caused by the application itself.
All errors in java are unchecked type.	Exceptions include both checked as well as unchecked type.
Errors happen at run time. They will not be known to compiler.	Checked exceptions are known to compiler where as unchecked exceptions are not known to compiler because they occur at run time.
It is impossible to recover from errors.	You can recover from exceptions by handling them through try-catch blocks
Examples : java.lang.StackOverflowError, java.lang.OutOfMemoryError	Examples : Checked Exceptions : SQLException, IOException Unchecked Exceptions :

	ArrayIndexOutOfBoundsException, ClassCastException, NullPointerException
--	--

17. What is the difference between Error, throwable and exception?

Refer Answer 15 and 16

18. What are collection APIs, give me an example

The [Java Collections Framework](#) is a collection of interfaces and classes which helps in storing and processing the data efficiently. Collections are growable in nature ie base on requirement we can increase or decrease the size. Collections can hold both homogeneous and heterogeneous elements.

Every collection class are implemented based on some standard data structure.

Example - List, Set, Maps

19. What is the difference between final and finally?

final is a keyword which is used to make a variable or a method or a class as “**unchangeable**”. In simple terms, A variable which is declared as final, it's value can not be changed once it is initialized. A method declared as final can not be overridden or modified in the sub class. A class declared as final cannot be extended.

Finally

finally is a block which is used for exception handling along with try and catch blocks. finally block is always executed whether exception is raised or not and raised exception is handled or not. Most of time, this block is used to close the resources like database connection, I/O resources etc.

20. Will java supports multiple inheritance?

No java does not support multiple inheritance but it supports multiple interfaces which can we accomplished as multiple inheritance.

21. What are the different types of interface? (Ans List, set, Queue)

22. What are wrapper class? Give me an example

As the name says, a wrapper class wraps (encloses) around a data type and gives it an object appearance. Wherever, the data type is required as an object, this object can be used. Wrapper classes include methods to unwrap the object and give back the data type.

```
int k = 100;  
Integer it1 = new Integer(k);
```

The **int** data type **k** is converted into an object, **it1** using **Integer** class. The **it1** object can be used in Java programming wherever **k** is required an object.

23. What is boxing and unboxing in Java? Explain with an example

Autoboxing: Automatic conversion of primitive types to the object of their corresponding wrapper classes is known as autoboxing. For example - conversion of int to Integer, long to Long, double to Double etc.

Unboxing: It is just the reverse process of autoboxing. Automatically converting an object of a wrapper class to its corresponding primitive type is known as unboxing. For example - conversion of Integer to int, Long to long, Double to double etc

24. Explain for each loop

- The foreach Java loop was introduced in *JDK 1.5*.
- The foreach loop enables to traverse through [the arrays](#) sequentially, without using the index variable.
- The Foreach is often used in arrays.

25. What are iterators, explain with an example

Iterators are one of the cursors in java which are used to retrieve object from the collects one by one.

> Iterators are universal i.e it is applicable for any collections

> Read and remove operation can be done using iterator

> We can create Iterator object by using `iterator()` method of Collection interface. `Iterator itr = c.iterator();` where `c` is the collection object.

Iterator interface defines the following three method

- a. `Public Boolean hasNext()`
- b. `Public object next()`
- c. `Public void remove();`

Limitations are Iterators can move single direction and replace cannot be performed

26. How do you access Private variables in different class?

We can access private variable by using getter and setter methods. Also we can pass the value to a constructor and using this keyword we can assign value to a private variable.

27. Prepare for one java program to write on the board

28. What is Constructor Over loading?

Constructor overloading is a technique in Java in which a class can have any number of constructors that differ in parameter lists. The compiler differentiates these constructors by taking into account the number of parameters in the list and their type.

29. With out using sync key word how do you perform synchronization?

30. What is Super keyword ? when and where do you use it ?

Super keyword refers to immediate parent of a class

- a. super.<variable_name> refers to the variable of variable of parent class
- b. super() invokes the constructor of immediate parent class.
- c. super.<method_name> refers to the method of parent class.

1. Write a program to print Fibonacci series.

```
2. public class Fiboseries {
3.     static int n1=0,n2=1,n3=0;
4.     static void printFibonacci(int count){
5.         if(count>0){
6.             n3 = n1 + n2;
7.             n1 = n2;
8.             n2 = n3;
9.             System.out.print(" "+n3);
10.            printFibonacci(count-1);
11.        }
12.    }
13.    public static void main(String args[]){
14.        int count=10;
15.        System.out.print("The fibonocci series of "+count+ " is
16.        "+n1+" "+n2); //printing 0 and 1
17.        printFibonacci(count-2);
18.    }
```

```
Problems @ Javadoc Declaration Console
<terminated> Fiboseries [Java Application] C:\Program Files\Java\jre1.8.0_91\bin\javaw.exe (Jul 31, 201
The fibonocci series of 10 is 0 1 1 2 3 5 8 13 21 34
```


2. Write a program to find out duplicate characters in a string

```
public class Duplicatechar {

    static void duplicateCharCount(String inputString)
    {
        //Creating a HashMap containing char as key and it's occurrences as
value
        HashMap<Character, Integer> charCountMap = new HashMap<Character,
Integer>();

        //Converting given string to char array

        char[] strArray = inputString.toCharArray();

        //checking each char of strArray

        for (char c : strArray)
        {
            if(charCountMap.containsKey(c))
            {
                //If char is present in charCountMap, incrementing it's count
by 1

                charCountMap.put(c, charCountMap.get(c)+1);
            }
            else
            {
                //If char is not present in charCountMap,
                //putting this char to charCountMap with 1 as it's value

                charCountMap.put(c, 1);
            }
        }

        //Getting a Set containing all keys of charCountMap

        Set<Character> charsInString = charCountMap.keySet();
```



```

        System.out.println("Duplicate Characters In "+inputString);

        //Iterating through Set 'charsInString'

        for (Character ch : charsInString)
        {
            if(charCountMap.get(ch) >1)
            {
                //If any char has a count of more than 1, printing it's count

                System.out.println(ch + " : "+ charCountMap.get(ch));
            }
        }
    }

    public static void main(String[] args)
    {
        duplicateCharCount("JavaJ2EE");

        duplicateCharCount("Fresh Fish");

        duplicateCharCount("Better Butter");
    }
}

```

```

<terminated> Duplicatechar [Java Application] C:\Program Files\Java\jre1.8.0_91
Duplicate Characters In JavaJ2EE
a : 2
E : 2
J : 2
|

```

Write a program for Insertion Sort in java.

```

public class inserionsort
{

    public static void main(String a[]){
        int[] arr1 = {58,34,2,56,7,67,88,42};
        int[] arr2 = doInsertionSort(arr1);
        for(int i:arr2){
            System.out.print(i);
            System.out.print(", ");
        }
    }
}

```

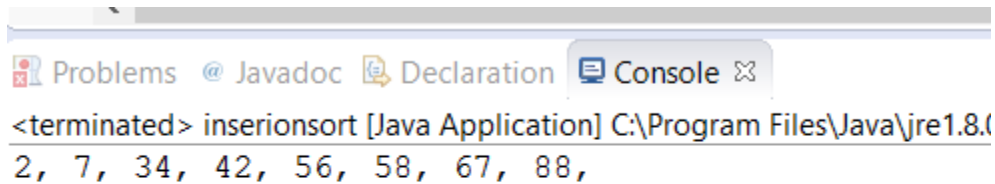
```

    }

    public static int[] doInsertionSort(int[] input){

        int temp;
        for (int i = 1; i < input.length; i++) {
            for(int j = i ; j > 0 ; j--){
                if(input[j] < input[j-1]){
                    temp = input[j];
                    input[j] = input[j-1];
                    input[j-1] = temp;
                }
            }
        }
        return input;
    }
}

```



The screenshot shows an IDE window with tabs for Problems, Javadoc, Declaration, and Console. The Console tab is active, displaying the output of a Java application named 'inserionsort'. The output indicates the application has terminated and shows the array [2, 7, 34, 42, 56, 58, 67, 88].

```

<terminated> inserionsort [Java Application] C:\Program Files\Java\jre1.8.0
2, 7, 34, 42, 56, 58, 67, 88,

```

How to swap two variables, by using pass by reference method ?

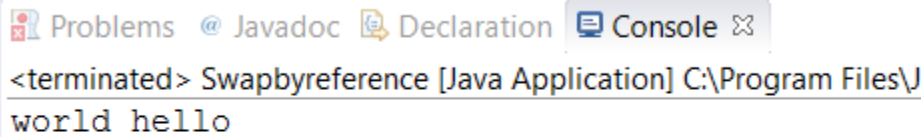
```

public class Swapbyreference {

    public static void main(String[] args) {
        String a[] = {"hello", "world"};
        swap(a);
        System.out.println(a[0] + " " + a[1]);
    }

    static void swap(String[] a) {
        String t = a[0];
        a[0] = a[1];
        a[1] = t;
    }
}

```



The screenshot shows an IDE console window with tabs for Problems, Javadoc, Declaration, and Console. The Console tab is active, displaying the output of a Java application named 'Swapbyreference'. The output consists of two lines: '<terminated> Swapbyreference [Java Application] C:\Program Files\J' and 'world hello'.

```
<terminated> Swapbyreference [Java Application] C:\Program Files\J
world hello
```

How to make a list immutable?

```
public class unmodifiablelist {

    public static void main(String[] args) {

        List<Character> list = new ArrayList<Character>();

        list.add('x');
        list.add('y');

        System.out.println("Initial list"+ list);

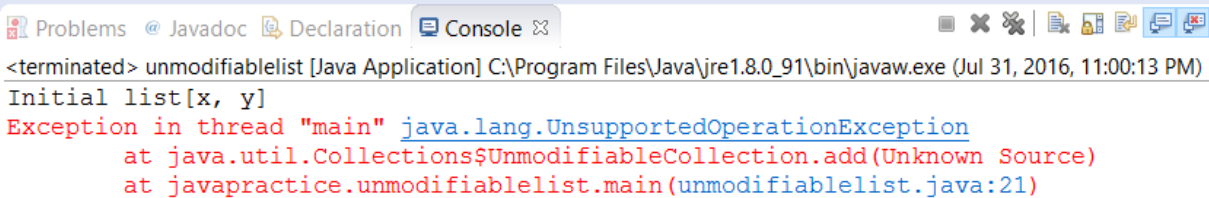
        List<Character> immutablelist =
Collections.unmodifiableList(list);

        immutablelist.add('a');

        System.out.println(immutablelist);

    }

}
```



The screenshot shows an IDE console window with tabs for Problems, Javadoc, Declaration, and Console. The Console tab is active, displaying the output of a Java application named 'unmodifiablelist'. The output shows the initial list '[x, y]' and then an exception: 'Exception in thread "main" java.lang.UnsupportedOperationException at java.util.Collections\$UnmodifiableCollection.add(Unknown Source) at javapractice.unmodifiablelist.main(unmodifiablelist.java:21)'.

```
<terminated> unmodifiablelist [Java Application] C:\Program Files\Java\jre1.8.0_91\bin\javaw.exe (Jul 31, 2016, 11:00:13 PM)
Initial list[x, y]
Exception in thread "main" java.lang.UnsupportedOperationException
    at java.util.Collections$UnmodifiableCollection.add(Unknown Source)
    at javapractice.unmodifiablelist.main(unmodifiablelist.java:21)
```

Write a program to remove duplicates from sorted array

```
public class removeduplicatearray {  
  
    public static int[] removeDuplicates(int[] input){  
  
        int j = 0;  
        int i = 1;  
        //return if the array length is less than 2  
        if(input.length < 2){  
            return input;  
        }  
        while(i < input.length){  
            if(input[i] == input[j]){  
                i++;  
            }  
            else  
            {  
                input[++j] = input[i++];  
            }  
        }  
        int[] output = new int[j+1];  
        for(int k=0; k<output.length; k++){  
            output[k] = input[k];  
        }  
  
        return output;  
    }  
  
    public static void main(String a[]){  
        int[] input1 = {2,2,6,6,8,9,10,10,10,12,12};  
        int[] output = removeDuplicates(input1);  
        for(int i:output){  
            System.out.print(i+" ");  
        }  
    }  
}
```

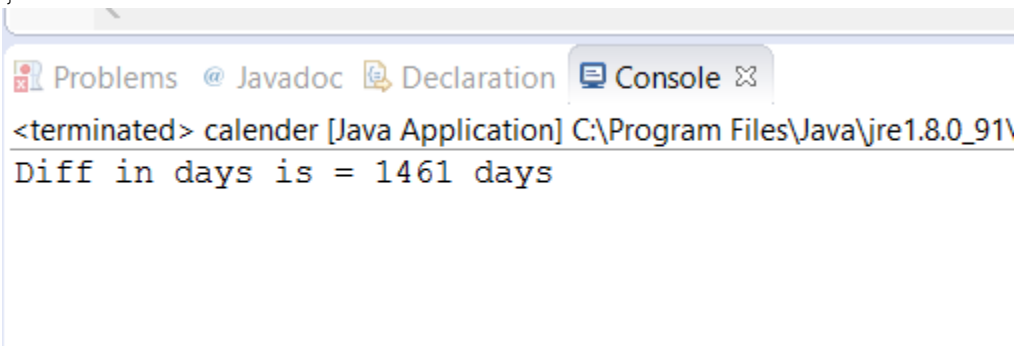
Problems @ Javadoc Declaration Console

<terminated> removeduplicatearray [Java Application] C:\Program I
2 6 8 9 10 12

.....

Find out the number of days in between two given dates ?

```
public class calender {  
  
    public static void main(String[] args) {  
  
        Calendar calendar1 = Calendar.getInstance();  
        Calendar calendar2 = Calendar.getInstance();  
  
        calendar1.set(2012, 5, 23);  
        calendar2.set(2016, 5, 23);  
  
        long milisecdate1 = calendar1.getTimeInMillis();  
        long milisecdate2 = calendar2.getTimeInMillis();  
  
        long diffinmilisec = milisecdate2 - milisecdate1;  
  
        long diffindays = diffinmilisec / (24*60*60*1000);  
  
        System.out.println("Diff in days is = " + diffindays + " days");  
    }  
}
```



Write a program to create deadlock between two threads

```
public class deadlock  
{  
  
    String str1 = "Java";  
    String str2 = "UNIX";  
  
    Thread trd1 = new Thread("My Thread 1") {  
        public void run() {  
            while(true) {  
                synchronized(str1) {  
                    synchronized(str2) {  
                        System.out.println(str1 + str2);  
                    }  
                }  
            }  
        }  
    }  
}
```

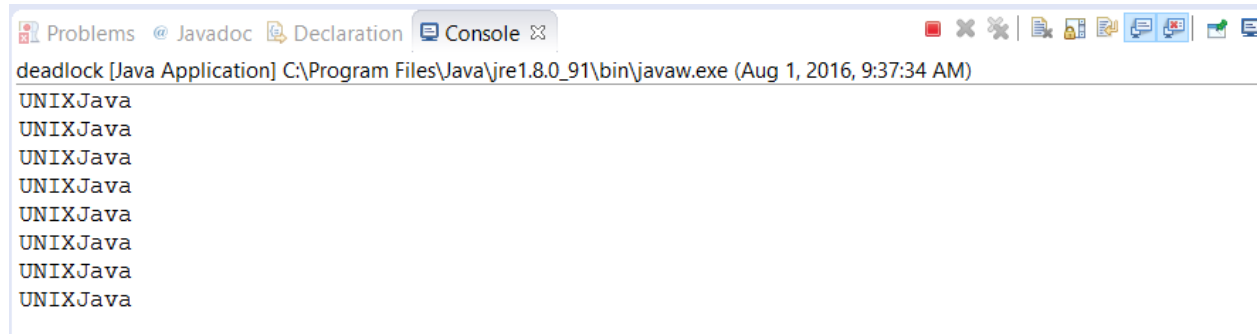
```

};

Thread trd2 = new Thread("My Thread 2"){
    public void run(){
        while(true){
            synchronized(str2){
                synchronized(str1){
                    System.out.println(str2 + str1);
                }
            }
        }
    }
};

public static void main(String a[]){
    deadlock mdl = new deadlock();
    mdl.trd1.start();
    mdl.trd2.start();
}
}

```



```

deadlock [Java Application] C:\Program Files\Java\jre1.8.0_91\bin\javaw.exe (Aug 1, 2016, 9:37:34 AM)
UNIXJava
UNIXJava
UNIXJava
UNIXJava
UNIXJava
UNIXJava
UNIXJava
UNIXJava

```