Java Theory Test

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

JVM (Java Virtual Machine) is an abstract machine, a specification that provides runtime environment in which java bytecode can be executed.

The JVM performs following operation:

- Loads code
- Verifies code
- Executes code
- Provides runtime environment

VM provides definitions for the:

- Memory area
- Class file format
- Register set
- Garbage-collected heap
- Fatal error reporting etc.

It is:

- **A specification** where working of Java Virtual Machine is specified. But implementation provider is independent to choose the algorithm. Its implementation has been provided by Sun and other companies.
- **An implementation** Its implementation is known as JRE (Java Runtime Environment).
- **Runtime Instance** Whenever you write java command on the command prompt to run the java class, an instance of JVM is created.

Memory Allocation:

- In Java, memory is allocated only to objects. There is no explicit allocation of memory, there is only the creation of new objects.
- The Java runtime employs a garbage collector that reclaims the memory occupied by an object once it determines that object is no longer accessible. This automatic process makes it safe to throw away unneeded object references because the garbage collector does not collect the object if it is still needed elsewhere.
- In Java, it's easy to let go of an entire "tree" of objects by setting the reference to the tree's root to `null`; the garbage collector will then reclaim all the objects

2. What is Polymorphism and encapsulation?

Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as as single unit. In encapsulation the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class, therefore it is also known as data hiding.

To achieve encapsulation in Java

- Declare the variables of a class as private.
- Provide public setter and getter methods to modify and view the variables values.

Polymorphism: it describes a language's ability to process objects of various types and classes through a single, uniform interface.

Polymorphism in Java has two types: Compile time polymorphism (static binding) and Runtime polymorphism (dynamic binding). Method overloading is an example of static polymorphism, while method overriding is an example of dynamic polymorphism.

3. What is method overloading and Method over riding?

Method overloading in Java occurs when two or more methods in the same class have the exact same name but different parameters. Overloading happens at compile time.

Method overriding If a derived class requires a different definition for an inherited method, then that method can be redefined in the derived class. This would be considered overriding. An overridden method would have the exact same method name, return type, number of parameters, and types of parameters as the method in the parent class, and the only difference would be the definition of the method. Happens at run time.

4. Why string is Immutable?

Java uses the concept of string literal.Suppose there are 5 reference variables,all referes to one object "sachin".If one reference variable changes the value of the object, it will be affected to all the reference variables. That is why string objects are immutable in java.

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

String	StringBuffer
String class is immutable.	StringBuffer class is mutable.
String is slow and consumes more memory when you concat too many strings because every time it creates new instance.	StringBuffer is fast and consumes less memory when you cancat strings.
String class overrides the equals() method of Object class. So you can compare the contents of two strings by equals() method.	StringBuffer class doesn't override the equals() method of Object class.

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

Array	**ArrayList**
static	dynamic
Doesn’t support generics and is not compile time type safe	Supports generics and is compile time type safe
Takes less memory	Takes more memory compared to arrays
Holds primitives and objects	Holds only objects

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

Hash Map	**Hash table**
HashMap is **non synchronized**. It is not-thread safe and can't be shared between many threads without proper synchronization code.	Hashtable is **synchronized**. It is thread-safe and can be shared with many threads.
Allows null key values and multiple null key values	Doesn’t allow any null or null key value
Fast	slow
Traversed by iterator which is fail fast	Traversed by enumerator which is not fail fast
Inherits AbstractMap class	Inherits Dictionary class

8. What is a vector in Java?

Vector implements a dynamic array. It is similar to ArrayList, but with two differences:

- Vector is synchronized.
- Vector contains many legacy methods that are not part of the collections framework.

9. What is set in java?

A Set is a Collection that cannot contain duplicate elements. It models the mathematical set abstraction.

The Set interface contains only methods inherited from Collection and adds the restriction that duplicate elements are prohibited.

10. What is an abstract class?

Abstract classes are classes that contain one or more abstract methods. Abstract method is a method that is declared, but contains no implementation. Abstract classes may not be instantiated, and require subclasses to provide implementations for the abstract methods.

- Abstract classes may or may not contain *abstract methods* ie., methods with out body (public void get();)
- But, if a class have at least one abstract method, then the class **must** be declared abstract.
- If a class is declared abstract it cannot be instantiated.
- To use an abstract class you have to inherit it from another class, provide implementations to the abstract methods in it.
- If you inherit an abstract class you have to provide implementations to all the abstract methods in it.

11. What is an interface?

An interface is a reference type in Java, it is similar to class, it is a collection of abstract methods. A class implements an interface, thereby inheriting the abstract methods of the interface.

An interface is similar to a class in the following ways:

- An interface can contain any number of methods.
- An interface is written in a file with a **.java** extension, with the name of the interface matching the name of the file.
- The byte code of an interface appears in a **.class** file.
- Interfaces appear in packages, and their corresponding bytecode file must be in a directory structure that matches the package name.

However, an interface is different from a class in several ways, including:

- You cannot instantiate an interface.

- An interface does not contain any constructors.
- All of the methods in an interface are abstract.
- An interface cannot contain instance fields. The only fields that can appear in an interface must be declared both static and final.
- An interface is not extended by a class; it is implemented by a class.
- An interface can extend multiple interfaces.

12. Why Java is Platform independent?

With Java, you can compile source code on Windows and the compiled code (bytecode to be precise) can be executed (interpreted) on any platform running a JVM. So yes you need a JVM but the JVM can run any compiled code, the compiled code is platform independent.

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

The access modifiers in java specifies accessibility (scope) of a data member, method, constructor or class.

There are 4 types of java access modifiers:

1. private
2. default
3. protected
4. public

14. What are java exceptions? Give me an example

In java, exception is an event that disrupts the normal flow of the program. It is an object which is thrown at runtime.

1. Checked Exception
2. Unchecked Exception
3. Error

15. What is the difference between throws and throwable?

throws is also a keyword in java which is used in the method signature to indicate that this method may throw mentioned exceptions. The caller to such methods must handle the mentioned exceptions either using try-catch blocks or using throws keyword. Below is the syntax for using throws keyword.

Throwable is a super class for all types of errors and exceptions in java. This class is a member of java.langpackage. Only instances of this class or it's sub classes are thrown by the java virtual machine or by the throw statement. The only argument of catch block must be of this type or it's sub classes. If you want to create your own customized exceptions, then your class must extend this class.

16. What is the difference between Error and exception?

An error is an irrecoverable condition occurring at runtime like out of memory error. These kind of jvm errors cannot be handled at runtime.

Exceptions are because of condition failures, which can be handled easily at runtime.

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

Throwable is a super class for all types of errors and exceptions in java. This class is a member of java.langpackage. Only instances of this class or it's sub classes are thrown by the java virtual machine or by the throw statement. The only argument of catch block must be of this type or it's sub classes. If you want to create your own customized exceptions, then your class must extend this class.

Both **java.lang.Error** and **java.lang.Exception** classes are sub classes of **java.lang.Throwable** class.

java.lang.Error class represents the errors which are mainly caused by the environment in which application is running. For example, **OutOfMemoryError** occurs when JVM runs out of memory or **StackOverflowError** occurs when stack overflows.

java.lang.Exception class represents the exceptions which are mainly caused by the application itself. For example, **NullPointerException** occurs when an application tries to access null object or**ClassCastException** occurs when an application tries to cast incompatible class types. In this article, we will discuss the differences between Error and Exception in java

18. What are collection APIs, give me an example

The Collection API is a set of classes and interfaces that support operation on collections of objects. These classes and interfaces are more flexible, more powerful, and more regular than the vectors, arrays, and hashtables if effectively replaces.

Example of classes: HashSet, HashMap, ArrayList, LinkedList, TreeSet and TreeMap. Example of interfaces: Collection, Set, List and Map.

19. What is the difference between final and finally?

Final	Finally
Final is used to apply restrictions on class, method and variable. Final class can't be inherited, final method can't be	Finally is used to place important code, it will be executed whether exception is handled or not.

overridden and final variable value can't be changed.	

20. Will java supports multiple inheritance? No Java does not support multiple inheritance.

In multiple inheritance, child classes inherit both the methods *and the data* of the parent classes. This can cause problems, especially when the same named entity (whether method or data) can appear in multiple parents, or if two or more parents actually contain the exact same entity they themselves inherited from a common ancestor.

21. What are the different types of interface?

List, Set and Queue

22. What are wrapper class? Give me an example?

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.

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

The automatic conversion of primitive data types into its equivalent Wrapper type is known as boxing and opposite operation is known as unboxing.

```
  class BoxingExample1{
  public static void main(String args[]){
    int a=50;
             Integer a2=new Integer(a);//Boxing
Integer a3=5;//Boxing
System.out.println(a2+" "+a3);  }
    }


class UnboxingExample1{


  public static void main(String args[]){


    Integer i=new Integer(50);


        int a=i;




        System.out.println(a);
```

```
 }
```

24. Explain for each loop

It is mainly used to traverse array or collection elements. The advantage of for-each loop is that it eliminates the possibility of bugs and makes the code more readable.

```
for(data_type variable : array | collection){}
```

25. What are iterators, explain with an example

Iterator enables you to cycle through a collection, obtaining or removing elements.

```
public class IteratorDemo {



  public static void main(String args[]) {

    // Create an array list

    ArrayList al = new ArrayList();

    // add elements to the array list

    al.add("C");

    al.add("A");

    al.add("E");

    // Use iterator to display contents of al

    System.out.print("Original contents of al: ");

    Iterator itr = al.iterator();

    while(itr.hasNext()) {

      Object element = itr.next();

     System.out.print(element + " "); }}}
```

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

Use getter and setter methods.

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

ThreadSafe class - any class that extends this will be able to check if some thread has acquired a lock on it and let its instance be accessed by one thread at a time.

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

The **super** keyword in java is a reference variable that is used to refer immediate parent class object.

1. super is used to refer immediate parent class instance variable.
2. super() is used to invoke immediate parent class constructor.
3. super is used to invoke immediate parent class method.