**JAVA Interview Questions Part1**

**1. What would you rate your Java experience out of 10?**

6→ is normal

7→ is good

8→ is well

I can rate my Java knowledge in testing perspective 7.

**2. What is your Java level? Do you use it in terms of testing only or in terms of development as well?**

I use in terms of testing

**3. OOP concepts and how you used them in your last project?**

I used OOP concepts in my last project for example encapsulation for hiding my data. Getter is helping to read the hiding data, setter for set the data. I use encapsulation in my POJO class. I declared variables as private. Because I need to use getter and setter methods in the API framework.

Another concept is inheritance. It is for use to improve the reusability. Object, methods, variables. Inherit the one class to another class. In the testNG framework the test base page must be readable for every single class. Because the before method and after method must be reused. In that case I use inheritance in my test base.

Abstraction- achieve the abstraction two ways, abstract class or interface. I can use the abstract method in my base page. Interface provided by selenium library which I have used in my framework for example WebDriver, take screenshot, list etc.

Polymorphism, like a parent-child relation. Parent to be reference to child objects. Driver class is helping to achieve multi-browser testing. I can test my test cases in different browsers. So I should use polymorphism.

**4. What do you know about Interface?**

Interfaces are from the abstraction concept. Interface is pure abstraction. It uses to achieve abstraction because the abstract method you can only create either in abstract class or either interface. Abstract class one class inherits only from one abstract class. However one class can inherit from multiple interfaces. Main benefits interface it allow to have multiple inheritance. One class can inherit 100 or more interfaces at the same time.

**5. What is the benefit of Interface?**

The main benefit is it allows you to have multiple inheritances. So one class can inheritance a hundreds or many more interface the same time

**6. What kind of collections do you use in Java?**

Collection is a one of the three data structure in Java:

Array, Collection and Map.

In Selenium I use **List** for List of elements. I also use **set** collection types in my framework. **get ()** window handles method. To handle multiple tabs. The return type of this get() windows handles method is going to return me set(). Because they can not be duplicated, the window can return that method.

In selenium if i have to test multiple checkboxes and if i want to get elements from dropdown box whenever I want to avoid the duplicate element, i can apply the set collections type. if I want to use all duplicate elements from dropdown instead of using list i can use set. so that way only gives me unice element from the dropdown.

3td date structure Map. But Map does have inheritance relation with collection. Map use for key and value. In API testing data store json file key and value format. In the Configuration Properties file data store also key and value format. Map, we can also use database testing. If we take data from a table we can use it as well.

**7. How to call the last stored variable in an Array?**

We can use length() to give us a total index number and we need to find the last element with length()-1.

**8. Where do you use Set, HashMap, List in your framework?**

Set can us it whenever I need to get unic option from dropdown or whenever I need to handle multiple tabs such as get window handle methods

HashMap if I wanna get the data from two columns from a table. if I wanna get the data from properties file / json fail i can use the hashMap

List I use it when I want to get all of the elements from dropdown or checkbox. if i need to get duplicate elements i do use the List.

**9. What is the difference between iterator and for/for each loop?**

We can use an iterator in ArrayList but not array. Iterator will help us iterate through the elements. Difference is with an iterator you can make changes/remove items to the list while iterating but within for each loop we cannot make changes to our list.

**10. What iterator methods do you use?**

ArrayList<Integer>list=newArrayList<>(Arrays.asList(100,100,200,200,300,300,40));

Iterator<Integer> it= list.iterator();

while(it.hasnext()){

if(it.next()<300){

it.remove();

}

}

System.out.println.(list);

-.-> iterator(), hasnext(),next(),remove() methods are iterative methods. we can use them for iterator interfaces.

**11. How do you write or read data from or to file with Java?**

**12. Difference between Heap and Stack?**

Both are used for memory but have different purposes

**Stack:** Stores primitives, method calls, and references

**Heap:** Stores the objects themselves. The String pool is also in the heap.

**13. You have a list of fruits which some of them repeats. How would you write a method which will return a fruit that repeats the most?**

**14. Java version you are using?**

8 and 11

**15. Abstraction vs Interface?**

**16. Java generics?**

**17. What is HashMap explain, how do you use in your framework?**

**18. What is thread safe? How do you make your code thread safe?**

**19. String builder/buffer differences?**

**20. How can you achieve abstraction?**

**21. What is the difference between List, Queue and Set?**

**22. What is a Map?**

**23. How do you iterate over HashMap?**

**24. What is garbage?**

**25. What is class in Java?**

Class is the template of the object. What we have a class we will be able to create objects from it. We can create multiple objects from the class. object cannot exist without a class.

**26. What kind of exception did you face and how did you handle it?**

***Java*** has two type exceptions: checked and unchecked exceptions. Ex.(*NoSuchMethodException, NullPointerException, RuntimeException )*

**Selenium** has just one type exception; it is an unchecked exception. Ex(*NoSuchElementException, NoSuchFrameException, NoSuchWindowException).*

***Handling;***

**Checked Exceptions→**These are the exceptions that are checked at compile time. If some code within a method throws a checked exception, then the method must either handle the exception or it must specify the exception using the [throws keyword](https://www.geeksforgeeks.org/throw-throws-java/) OR The **try-catch** is the simplest method of handling exceptions. Put the code you want to run in the try block, and any Java exceptions that the code throws are caught by one or more catch blocks. This method will catch any type of Java exceptions that get thrown. This is the simplest mechanism for handling exceptions.

## **Unchecked Exceptions →**In Java exceptions under Error and RuntimeException classes are unchecked exceptions, everything else under throwable is checked. Consider the following Java program. It compiles fine, but it throws *ArithmeticException* when run. The compiler allows it to compile because *ArithmeticException* is an unchecked exception.

**27. What is the difference between a HashMap and HashTable?**

**HashMap:** Works by entries, which are key/value pairs.

HashMap does not synchronize**.**

HasMap **takes the null keyword.**

**HashTable:** Same as HashMap BUT HashTable methods are **synchronized.**

(P.S!!Only Methods are SYNCHRONIZED, not CLASSES or VARIABLES) .

Hashtable **does not take Null keyword**

**28. What is Polymorphism? What is runtime polymorphism?**

​​a. Polymorphism is a very important concept in OOP because:

1. *it enables to change the behavior of the applications in the run time based on the object on which the invocation happens.*

*2. an object can have different forms.*

b. Two types: **Compile Time** which is Static and **Run Time** Polymorphism which is related with child and parent class.

c. Polymorphism is implemented using the concept of Method overloading and method overriding. This can only happen when the classes are under the parent and child relationship using inheritance.

**29. Give me an example from your framework where you used OOP concepts?**

**Encapsulation** →I use it for access modifying in my project. All the classes in a framework are an example of Encapsulation. In POM classes, we declare the data members using **@FindBy** and initialization of data members will be done using [Constructor](https://www.softwaretestingmaterial.com/java-tutorial/#constructor) to utilize those in methods.

**Interface**→ I use it in my DriverClass. WebDriver itself is an Interface. Basic statement we all know in Selenium is WebDriver driver = new FirefoxDriver(); I am initializing the Firefox browser using Selenium WebDriver. It means I am creating a *reference variable (driver)* of the *interface (WebDriver)* and creating an *Object*. Here *WebDriver* is an *Interface* as mentioned earlier and *FirefoxDriver* is a *class*.

**​​Inheritance** → I use it in my BasePage and TestBase classes. I extend the Base Class in other classes such as Tests and Utility Class. I can extend one class (Base Class like WebDriver Interface) into another class (like Tests, Utility Class) with Inheritance.

**Polymorphism** → allows me to perform a task in multiple ways. I can also combine overloading and overriding with polymorphism.

**30. What is the global variable?**

Global variable is → static variable. Static/ global variables are always accessible. Static variables can be accessed by calling with the class name ClassName.VariableName. we can even use it inside of the instance method. However we cannot use instance variables inside static methods or blocks.

**31. How do you handle exceptions?**

I can handle the exception, by using the throws keyword OR The try-catch

**32. Difference between protected and default?**

If the same package exists, they are always visible. But if they are outside of the package one of them is visible or may be visible. protected can be visible out of the package if it has inheritance.

**33. Can you give "public: access modifier where you are overriding a protected method?**

**Yes**, the protected method of a superclass can be **overridden** by a subclass. If the superclass method is protected, the subclass overridden method can have **protected** or **public** (but not **default** or **private**) which means the **subclass** overridden method can not have a **weaker access specifier**.

**34. How can you find max value in an int array?**

To get the minimum or maximum value from the array we can use the **Collections.min()** and **Collections.max()** methods. But as this method requires a list type of data we need to convert the array to list first using the above explained “**aslist()**” function.

Integer[] num = { 2, 4, 7, 5, 9 };

int min = Collections.min(Arrays.asList(num));

int max = Collections.max(Arrays.asList(num));

**35. Can class be final?**

1- if the class is going to be super class, it can not be final.

2- Abstract class cannot be final, other classes can be final.

3- We make sure that we are not creating any subclasses because if we create a final keyword it can never be inherited to any other classes.

*P:S!!!*

1. *We can apply the Final keyword within the variable, but the final variable can not resign.*
2. *We can apply the Final keyword within the final method, but the final method can not be overridden.*

**36. What is the difference between Static methods and other methods ?**

In Java, a static method is a method that belongs to a class. Static method we can call it through in the classes. I use it in my framework. The get() method in the driver class is the static. from the configuration reader class getproperty() method is also static. And all the methods from the browser utilities class are static as well.

**37. Static, non-static methods?**

Any method of a class which is not static is called a non-static method or an instance method. Method does not contain a static keyword. it means it belongs to the object. if it belongs to the object we can not call it through in the classes.

1. Static Methods can access static variables without any objects, however non-static methods and non-static variables can only be accessed using objects.
2. Static methods can be accessed directly in static and non-static methods. For example the static public static void main() method can access the other static methods directly. Also, a non-static regular method can access static methods directly.

**38. Up casting, down casting?**

**39. What is the difference between final, finally and finalize?**

**final** is the **keyword** → we can apply to the class, method and variable.

**finally** is **block**→ we use it in **try-catch** block. we can not use it without a **try-catch** block.

**finalize** is **method→** we don't need to use it. Garbage collectors use it implicitly. Garbage collectors always call it just **before** the deletion/destroying the object which is eligible for Garbage Collection, so as to perform **clean-up activity**.

final is the keyword finally is block finalize is method

**40. Tell me how do you understand immutability in java and what is the difference between mutable?**

Immutability is unchangeable. for variable methods and classes. if we use the final keyword it means those issues are immutability and string variables are also immutability. Immutability objects can not change. String pool is also immutable. we can not duplicate it, it is only one object.

**Mutable:** It supports get() and set() methods to deal with the object.

It provides methods to change the object.

we can duplicate it,

**Immutable:** It only supports the get() method to pass the value of the object.

It does not provide any method to change the object value.

we can not duplicate it,

**41. Can we have multiple inheritance in java? If yes, how do we achieve that?**

For the classes no we can not do multiple inheritance in java. We can not extend more than one class. But in interfaces it is possible. We can achieve it by using the **"implemented"** keyword.

**42. How did you implement polymorphism in your framework?**

I made a WebDriver as a reference and the object is a chrome, firefox or etc.

WebDriver driver= new ChromeDriver();

**43. What could be the reason if you are getting NullPointerException?**

Null keyword does not reference any object or any instances. So if there is no object, no instances, if we still call the instances this time we will get a nullPointerException.

**44. Array or ArrayList to use? Why?**

If I am only dealing with the objects that I can use ArrayList. And if some Primitives are also involved then I can use Array.

ArrayList size is adjustable. We can add or remove items. we can create many methods with arraylist. Array we can add or remove items. we can not change the size. Array’s advantage is that it is faster than arrayList.

**45. Method overloading VS Method Overriding?**

* In case of o**verloading**, method names must be the same, but the parameters must be different.
* In case of **overriding**, method name and parameters must be the same.
* In method **overloading**, the return type can be the same or different.
* In method **overriding**, return type must be the same or covariant type.
* We can **overload** methods in the same class but method **overriding** occurs in two classes that have inheritance relationships.
* We cannot **override** static, final, or private methods in Java,
* But we can **overload** static, final, and private methods in Java.

**46. What are the differences between Throw and Throws keyword?**

**47. Tell me the differences between HashMap and LinkedHashMap?**

Order is the difference between hashMap and LinkedHasMap.

**HashMap:** There is no such order. Ordet can be roundem.

**LinkedHashMap:** it keeps the insertion order. 1th element is the 1th, the 2nd element is the 2nd.

**48. Tell me the differences between Static and Instance Variable/Block?**

**Static and Instance Variable**

**Static Variable:** all the objects have the same value.

**Instance Variable:** one class we can copy multiple objects with different values.

**Static and Instance block**

A **static block** is a block that is executed only once.

**Instance blocks** are different. Unlike static blocks, they execute every time you create an object for a class.

A **static block** similarly contains code that executes only one time no matter how many times you create an object for that class.

The **instance block** contains code that needs to execute every time an object is created no matter through which constructor.

**49. What's the use of super keyword? What's the use for this keyword?**

super is used to refer to super-class's // this is used to refer to current-class's

**this.** → instance variable & methods

**this()** → constructor refers to the current class object.

→It is used for invoking the current class method.

→ It can be used anywhere in the parameterized constructor.

**super.** → we need to use it in subclass.

**super()** → constructor refers to the immediate parent class object.

→ It is used for invoking parent class methods.

→ It is always the first line in the child class constructor.

**50. Why OOP concept is important for producing a software?**

**JAVA Interview Questions and Answers PART 2**

**1. Explain public static void main.**

**public →** is the access modifier. It means we can use it anywhere, like a package or outside of a package.

**static→** is the keyword present; it has only one specific copy of a variable or methods. we can access it with no objects because it can be called class name right away.

**void**→it means there is no return type of the method. We use the void to the method if we don't want the method to return any value at the end.

**main→** it is responsible for running java applications.

**String[] args**→ Java main method accepts a single argument of type String array. This is also called as java command line arguments.

**2. Can we execute a java program without using the main method?**

***No, we do need it***. In the earliest versions of java, like before version 8 this was possible, but after java version 8 it is not possible. Java 7 one of the ways to execute the program without the main method is using a static block.

P:S!! In testing we use other approaches such as the Test annotations to run our code. It can work because codes are running with junit or TestNg not java.

**3. Difference between local variable and instance variable?**

The first difference is the declaration. Local variable has to be in the block. Instance variables can be declared out of method.

The second difference is that the instance variable is default, local variable can not default.

instance variable we can use without initializing it can be given default result. Local variables should be initialized other way it will give an error.

**4. Main uses of this keyword? Can it be used to refer statics?**

**5. What is Java String Pool?**

***String pool is*** the ***memory*** location in the heap. this memory location specifically designed ***for string literal objects.*** we can create string in a two difference way;

1.String str1= “Cybertek”; // *String pool*

2. String str2= new String (“Cybertek”); //*heap*

*// what is the difference between these two String objects? → store the memory location where they are located is different.*

**6. What kind of exceptions have you faced and how you handled them?**

*In Selenium;* NoSuchElementException() ,NoSuchWindowException(), NoSuchFrameException(),WebdriverException() etc.

It has two ways to handle the exceptions. [throws keyword](https://www.geeksforgeeks.org/throw-throws-java/) and the **try-catch.**

**try-catch→ s**hould be inside the method.

throws keyword→ we declare it in the method signature. throws keyword look like an easy way but it has a disadvantage. whoever calls the method they will still have to handle this exception, that is the method thrown right down.

if we have a custom method in our utility class it is better to handle exceptions with try-catch. it means next time when we call it we will not get any exception.

**7. What is null pointer exception? What is class cast exception?**

**8. What is the most recent exception that you got?**

*In Selenium;* NoSuchElementException(), NoSuchWindowException(), NoSuchFrameException(),WebdriverException() etc.

**9. Static and non-static methods?**

**10. Difference between final method and abstract method?**

**Final method**; it can not be overridden. Static and non-static methods can be final. Constructor can not be final. We can not declare a constructor without a body.

**Abstract method;** it can be overridden in subclasses. Static method cannot be abstract. Private method it cannot be abstract.

**11. Can a class be static?**

If a question is in ***any class***, the answer is no!!! if it is an inner class. **YES** it can be static.

P:S!! inner class meaning is create a class in said the another class

*classA{ // class can not be static*

*static classB{* // *inner class can be static*

*}*

*} //*How do we extend the inner class?

*public class C* extend A.B{}

**12. What is constructor chaining in Java?**

**13. Has a VA Is A relation in Java?**

**is relation →** In Java, an Is-A relationship depends on inheritance. Further inheritance is of two types, class inheritance and interface inheritance. It is used for code reusability in Java. there is an **extends or implement** keyword in the class declaration in Java, then the specific class is said to be following the Is-A relationship.

**has relation →**In Java, a Has-A relationship simply means that an instance of one class has a reference to an instance of another class or another instance of the same class. For example, a car has an engine, a dog has a tail and so on. In Java, there is no such keyword that implements a Has-A relationship. But we mostly use **new** keywords to implement a Has-A relationship in Java.

**14. Tell us the logic behind all OPP concepts with examples.**

**15. When does the finally block not execute?**

if the system is running it never stops but if the system is terminated then none of the code gets executed that also includes the finally block. If we would like to stop the finally block we have only one legal way to make finally not execute is to call **System.exit** in try or catch

try {

throw new (RuntimeException);

} catch (RuntimeException e){

system.exit()

} finally {

}

**16. What is the length of EMPTY and null Strings?**

An empty string is a string instance of zero length, whereas a null string has no value at all. An empty string is represented as "". It is a character sequence of zero characters. A null string is represented by null. It can be described as the absence of a string instance.

String s1 = " "; // *this is represent to object*

String s2= null; // *it does not represent to object*

system.out.println (s1.length()); // output will be 0

system.out.println (s2.length()); // output will be *NullPointerexception.*

**17. What are the differences between Java and type script languages?**

By the way, I only use Java. Type script is also being used in Angular applications by the developer. I am not using a type script language.

**18. What is Singleton and where do you use it in your framework?**

Singleton is a design pattern. One instance at everywhere. In my framework in Driver class I have one instance. Driver class in my framework returns me to a driver. One instance being used in the inthaire framework is the singleton design pattern. When we develop a class in such a way that it can have only an instance at any time, it is called the Singleton design pattern. In a class we can create multiple instances, by object. When we create objects we are using a constructor.

*Driver class is the example of a singleton design pattern.*

**19. How do you compare two values from different data structures?**

**20. You have two string values; how do you compare them?**

**21. Difference between abstract class and non-abstract class?**

**22. Difference between extends keyword and implements keyword?**

**23. What are the limitations of Array? How it is different than List?**

**24. How to compare two Arrays? Two Lists?**

**25. Adding the elements of one Array to another?**

**26. How do you print any statement without using?**

**27. What are the limitations of an Interface?**

**28. Difference between class and interface?**

**29. What are the castings in java?**

**30. Give me an example of up casting and down casting?**