QA Automation interview Real Time Question

1] Can You Brief Me About Yourself ?

Pls Answer Intro that we have shared with You in Session.

==

2] Tell me your Day to Day activities as QA?

First thing I do after login in my system. I check the active sprint in Jira for our project code. There I can see my assigned open tasks. After that I will check my mail if there is any important mail I need to take action on.

Then we have our daily scrum meeting where we used to tell our previous day actions what we did,

what we are planning for today and if we have any blocker to discuss. Product owner and scrum master help us to resolve that blocker.

After that I need to take the pending task and do needed action whether creating test case, Execution, Defect retesting if any.

==

3] What is Data types?

Datatype are used to Represent the type of data or information which we are going to used in JAVA program.

Types Of Data Type

1.Primitive Data Type:-

There are eight types of primitive data type. All the primitive data types are keywords, memory size of primitive data type are fixed. The first alphabet is always small(Lowercase).

2.Non-Primitive Data Types:- There are two types of non-primitive data types. Non-primitive data types are identifier. Memory size of non-primitive data types are not fixed. The first alphabet is always capital (Upper Case).

1.Primitive

byte – 1byte
short – 2 byte
int- 4 byte
long-8byte
float-4 byte
double-8byte
char-2byte
boolean-1bit

2.Non-Primitive

String(not fix size)
Class(Not fix size)

Could you differentiate an Interface and an Abstract class?

An abstract class may have instance methods, which can implement a default behavior.

On the other hand, an interface can’t implement any default behavior. However, it can declare different constants and instance methods. While an interface has all the public members, an abstract class contains only class members like private, protected and so on.

What are the different ways you can use “Static”?

Static can be used in four ways: static variables, static methods, static classes and it can be used across a block of code in any class in order to indicate code that runs when a virtual machine starts and before the instances are created.

===

Do you think all property of Immutable Object needs to be final?

Not compulsory as we can easily achieve the same by making member as nonfinal but private and not changing them except in the constructor. Also, avoid providing setter methods for them. If it is a mutable object,

then prevent leaking any reference for that member.

===

In Java, what is the default value of Float and Double?

Default value of Float is 0.0f while 0.0d for Double.

Access specifier/modifier

1. What are the access modifiers in java?

>> The access modifiers in Java specifies the accessibility or scope of a field, method, constructor, or class.

There are four types of Java access modifiers:

1. Private
2. Default
3. Protected
4. Public

===

2. Can we have a private constructor in Java?

>> **Yes**, we can declare a constructor as private. If we declare a constructor as private we are not able to create an object of a class. We can use this private constructor in the Singleton Design Pattern.

(A singleton class is one which limits the number of objects creation to one.)

===

3. Which access specifier can be used with a class?
>> **Private**, **Default**, **Protected**, **Public.** All access specifier used with a class.

===

4. Can we declare a top-level class as private?
>> **No**, we cannot declare a top-level class as private or protected. It can be either public or default (no modifier). If it does not have a modifier, it is supposed to have a default access.

===

5. Can we declare an abstract method as private?
>> No, An abstract class is permitted to have both concrete and abstract methods.

===

6. Can a method or a class be final and abstract at the same time?
>> **No**, An abstract method can't declare as **final.** you cannot override final methods in Java. But, in-case of abstract, you must override an abstract method to use it.
abstract and final both terms are opposite to each other. An abstract method says you must redefine it in a derived class. A final method says you can't redefine this method in a derived class.

===

7. Why are access modifiers used?
>> In Java, access modifiers are used to set the accessibility (visibility) of classes, interfaces, variables, methods, constructors, data members, and the setter methods

8. Which is the default access modifier?
>> Default access modifier means we do not explicitly declare an access modifier for a class, field, method, etc. A variable or method declared without any access control modifier is available to any other class in the same package.

===

9. What is the default access modifier for Interface?
>> All abstract, default, and static methods in an interface are implicitly **public**. In addition, an interface can contain constant declarations. All constant values defined in an interface are implicitly public , static , and final. (Methods -> public Variables -> public, static, final.)

===

10. What is the role of private constructor in Java?
>> The private constructor is useful in case we want to restrict the object creation. For example, Singleton pattern can be implemented using a private constructor..
(A singleton class is one which limits the number of objects creation to one.)

===

11. Which is the least restrictive access modifier in Java?
>> public.
The public is the most common and least restrictive access modifier in Java. You can apply public modifier into variables, methods and both top-level and inner classes in Java.

===

12. Which is the most restrictive access modifier in Java?

>> **Private access modifier** is the most restrictive access level.
Methods, variables, and constructors that are declared private can only be accessed within the declared class itself.

==

13. Which access modifier is also known as Universal access modifier?.

>> **public** access modifier is also known as **universal access modifier**.
(**private** access modifier is also known as **native access modifier**,
default access modifier is also known as **package access modifier**,
protected access modifier is also known as an **inherited access modifier**.)

==

Can we create an Immutable object, which contains a mutable object?

Yes, its possible to create an Immutable object which may contain a mutable object, you just need to be a little bit careful not to share the reference of the mutable component, instead, you should return a copy of it if you have to. Most common example is an Object which contain the reference of java.util.Date object.

==

How do you convert bytes to String? (answer)

you can convert bytes to the string using string constructor which acceptsbyte[], just make sure that right character encoding otherwise platform's default character encoding will be used which may or may not be same.

==

Can we cast an int value into byte variable? what will happen if the value of int is larger than byte?

Yes, we can cast but int is 32 bit long in java while byte is 8 bit long in java so when you cast an int to byte higher 24 bits are lost and a byte can only hold a value from -128 to 128.

==

Which class contains clone method? Cloneable or Object? (answer)

java.lang.Cloneable is marker interface and doesn't contain any method clone method is defined in the object class. It is also knowing that clone is a native method means it's implemented in C or C++ or any other native language.

==

Can I store a double value in a long variable without casting? (answer)

No, you cannot store a double value into a long variable without casting because the range of double is more that long and you we need to type cast. It's not dificult to answer this question but many develoepr get it wrong due to confusion on which one is bigger between double and long in Java.

==

Why main method is static? Can we execute a program without main() method? If yes, how?

Java program's main method has to be declared static because keyword static allows main to be called without creating an object of the class in which the main method is defined. If we omit static keyword before main Java program will successfully compile but it won't execute. The answer to this question depends on the version of java you are using. Prior to JDK 7, the main method was not mandatory in a java program.

You could write your full code under static block and it ran normally.

The static block is first executed as soon as the class is loaded before the main(); the method is invoked and therefore before the main() is called. main is usually declared as static meth…

explain the various access modifiers used in Java.

Default – No keyword required

Private

Protected

Public

access-modifiers-in-java

1.Default:

When no access modifier is specified for a class , method or data member – It is said to be having the default access modifier by default.

The data members, class or methods which are not declared using any access modifiers i.e. having default access modifier are accessible only within the same package.

2.Private:

The private access modifier is specified using the keyword private.

- The methods or data members declared as private are accessible only within the class in which they are declared.
- Any other class of same package will not be able to access these members.
- Top level Classes or interface can not be declared as private because
- private means "only visible within the enclosing class".
- protected means "only visible within the enclosing class and any subclasses"
- Hence these modifiers in terms of application to classes, they apply only to nested classes and not on top level classes

3.protected:

The protected access modifier is specified using the keyword protected.

The methods or data members declared as protected are accessible within same package or sub classes in different package.

4.public:

The public access modifier is specified using the keyword public.

- The public access modifier has the widest scope among all other access modifiers.
- Classes, methods or data members which are declared as public are accessible from every where in the program. There is no restriction on the scope of a public data members.

Why Is String Immutable in Java?

An immutable object is an object whose internal state remains constant after it has been entirely created. This means that once the object has been assigned to a variable, we can neither update the reference nor mutate the internal state by any means.

The key benefits of keeping this class as immutable are caching, security, synchronization, and performance.

==

What are the different class loaders used by JVM??

The Java ClassLoader is a part of the Java Runtime Environment that dynamically loads Java classes into the Java Virtual Machine. The Java run time system does not need to know about files and file systems because of classloaders.

Java classes aren't loaded into memory all at once, but when required by an application. At this point, the Java ClassLoader is called by the JRE and these ClassLoaders load classes into memory dynamically.

==

Types of ClassLoaders in Java??

Not all classes are loaded by a single ClassLoader. Depending on the type of class and the path of class, the ClassLoader that loads that particular class is decided. To know the ClassLoader that loads a class the getClassLoader() method is used. All classes are loaded based on their names and if any of these classes are not found then it returns a NoClassDefFoundError or ClassNotFoundException.

A Java Classloader is of three types:

- **BootStrap ClassLoader:**

A Bootstrap Classloader is a Machine code **which kickstarts the operation when the JVM calls it**. It is not a java class. Its job is to load the first pure Java ClassLoader. Bootstrap ClassLoader loads classes from the location rt.jar. Bootstrap ClassLoader doesn't have any parent ClassLoaders. It is also called as the Primodial ClassLoader.

- **Extension ClassLoader:**

The Extension ClassLoader is a child of Bootstrap ClassLoader and loads the extensions of core java classes from the respective JDK Extension library. It loads files from jre/lib/ext directory or any other directory pointed by the system property java.ext.dirs.

• System ClassLoader:

An Application ClassLoader is also known as a System ClassLoader. It loads the Application type classes found in the environment variable CLASSPATH, -classpath or -cp command line option. The Application ClassLoader is a child class of Extension ClassLoader.

==

What is a join?

This is a keyword used to query data from more tables based on the relationship between the fields of the tables. Keys play a major role when JOINs are used.

What are the types of join and explain each?

There are various types of join which can be used to retrieve data and it depends on the relationship between tables.

Inner join.

Inner join return rows when there is at least one match of rows between the tables.

Right Join.

Right join return rows which are common between the tables and all rows of Right hand side table. Simply, it returns all the rows from the right hand side table even though there are no matches in the left hand side table.

Left Join.

Left join return rows which are common between the tables and all rows of Left hand side table. Simply, it returns all the rows from Left hand side table even though there are no matches in the Right hand side table.

Full Join.

Full join return rows when there are matching rows in any one of the tables. This means, it returns all the rows from the left hand side table and all the rows from the right hand side table.

What is a query?

A DB query is a code written in order to get the information back from the database. Query can be designed in such a way that it matched with our expectation of the result set. Simply, a question to the Database.

How many types of memory areas are allocated by JVM?

JVM (Java Virtual Machine) is an abstract machine, In other words, it is a program/software which takes Java bytecode and converts the byte code (line by line) into machine understandable code.

JVM(Java Virtual Machine) acts as a run-time engine to run Java applications. JVM is the one that actually calls the main method present in Java code. JVM is a part of the JRE(Java Runtime Environment).

JVM perform some particular types of operations:

- Loading of code
- Verification of code
- Executing the code
- It provide run-time environment to the users

==

Types of Memory areas allocated by the JVM??

All these functions take different forms of the memory structure. The memory in the JVM divided into 5 different parts:
Class(Method) Area
Heap
Stack
Program Counter Register
Native Method Stack
Let's see about them in brief:

• Class Loader:

It is a subsystem of JVM which is used to load class files.It is mainly responsible for three activities.

- Loading ,Linking, Initialization

• Class(Method) Area:

It stores class level data of every class such as the runtime constant pool, field and method data, the code for methods.

• Heap:

It is used to allocate memory to objects at run time

• Stack:

Each thread has a private JVM stack, created at the same time as thread. It is used to store data and partial results which will be needed while returning value for method and performing dynamic linking.
Java Stack stores frames and a new frame is created each time at every invocation of the method.
A frame is destroyed when its method invocation completes

Polymoprphism

1. What is polymorphism in java?

>> **Polymorphism :** we can perform a *single action in different ways.* (One object showing different behaviour at different stages of life cycle.)

==

2. Give real-life examples of polymorphism?

>> A person at the same time can have different characteristic. Like a man at the same time is a father, a husband, an employee.

==

3. How many types are of polymorphism in java?

>> 1. **Compile time** polymorphism (static binding)
2. **Runtime** polymorphism (dynamic binding).

==

4. What are compile-time and run-time polymorphism?

>> **1. Compile time polymorphism:-**
In compile time polymorphism method declaration is going to get binded to it's definition at **compile time**, based on **argument** is known as compile time polymorphism.
2. Runtime polymorphism:-
In run time polymorphism method declaration is going to get binded to it's definition at **run time**, based on **object creation** is known as run time polymorphism.

==

5. What is method overloading?

>> Declaring multiple methods with same name but with different argument in a same class is called method overloading.

==

6. What is method overriding?

>> when two methods have the same method name and parameters. One of the methods is in the parent class, and the other is in the child class is called method overriding.

==

7. How to achieve Compile-time polymorphism?

>> Compile-time polymorphism is achieved through **method overloading**.

==

8. How to achieve Runtime-time polymorphism?

>> run-time polymorphism is achieved through **method overriding**.

==

9. Can we achieve method overloading by changing the return type?

>> No, We cannot achieve method overloading through return type in java.

10. Difference between method overloading and overriding?

>> **Overloading**	**Overriding**
Method overloading is a compile time polymorphism.	Method overriding is a run time polymorphism.
It help to rise the readability of the program.	Used to provide specific implementation of the method that is already provided by its parent class or super class
It is perform within the class	It is performed in two classes with inheritance relationship.
In this, methods must have same name and different parameter.	In this, methods must have same name and same parameter.
return type can or can not be same, but we must have to change the parameter.	return type must be same or co-variant.
Example of Compile-time polymorphism.	Example of run-time polymorphism.

INTERFACE

1. Can we re-assign a value to a field of interfaces?

>> **No**, we can't change the value of an interface field because interface fields are final and static by default.

==

2. Can we declare an Interface with "abstract" keyword?

>> There can be only abstract methods in the Java interface, not method body. Interfaces can have abstract methods and variables. So **No** need to use it explicitly.

==

3. For every Interface in java, .class file will be generated after compilation. True or false?

>> **True**. For every interface written in a java file, class file will be generated after compilation.

==

4. Can we override an interface method with visibility other than public?

>> **No**. We cannot override an interface method if it's visibility is not public.
And if it has its visibility as public then you can override it with the same method signature (i.e., with the same access specifier public whenever you implement the interface to any class.

==

5. Can an interface extend a class?

>> An interface is **Not** extended by a class, it is implemented by a class. An interface can extend multiple interfaces.

==

6. Like classes, does interfaces also extend Object class by default?

>> interface itself **doesn't extend Object**, Interfaces in java don't inherit from Object class. They don't have default parent like classes in java.

7. Can interfaces have static methods?

>> Static Methods in Interface are those methods, which are defined in the interface with the **keyword static**. Similar to Default Method in Interface, Java interface static method helps us in providing security by not allowing implementation classes to override them.

==

8. What are marker interfaces? What is the use of marker interfaces?

>> A marker interface is an interface that has no methods or constants inside it. Marker interface is used as a tag that inform the Java compiler by a message so that it can add some special behaviour to the class implementing it.

==

9. Can we achieve multiple inheritance by using interface?

>> Multiple inheritance by interface occurs if a class implements multiple interfaces or also if an interface itself extends multiple interfaces.

==

10. Can we create an object of an interface?

>> **No**, we **can’t** create an object to interface but we can create classes where we can implement the abstract methods of the interface. (The classes which implement the abstract methods of interface are known as
implementation classes.)

==

11. Can we declare the interface as final?

>> **No**. We can not instantiate interfaces, so in order to make interfaces useful we must create subclasses. If you make a method final you cannot override it and, if you make a variable final you cannot modify it.

==

12. Can an interface extend another interface?

>> An **interface can extend another interface** in the same way that a class can extend another class.

==

13. Can we declare a constructor in the interface?

>> **No**, you cannot have a constructor within an interface in Java. You can have only public, static, final variables and, public, abstract, methods.
(From Java9 onwards interfaces allow private and private static methods.)

ABSTRACT CLASS

1. Abstract class must have only abstract methods. True or false?

>> **False**, An abstract class can have both abstract and concrete methods.

==

2. Is it compulsory for a class which is declared as abstract to have at least one abstract method?

>> **Not necessary,** abstract class may or may not contains abstract methods. but if we declared a class as abstract using abstract keyword then it is called abstract class.
But doesn't mean that it contains at least one abstract method.

==

3. Can we use "abstract" keyword with constructor and variable?

>> An abstract keyword **cannot be used** with variables and constructors. You cannot override a constructor you cannot provide body to it if it is made abstract.

==

4. Why final and abstract can not be used at a time?

>> **No**, We can't use abstract and final together because you cannot override final methods in Java. But, in-case of abstract, you must override an abstract method to use it.

==

5. Can we instantiate a class which does not have even a single abstract methods but declared as abstract?

>> **No**, We cannot instantiate an abstract class in Java because it is abstract, it is not complete, hence it cannot be used.

==

6. Can we declare abstract methods as private?

>> abstract method **cannot be private,** incase of an abstract method, you cannot use it from the same class, you need to override it from subclass and use. Therefore, the abstract method cannot be private.

==

7. We can't instantiate an abstract class. Then why constructors are allowed in abstract class?

>> It is because, we can't create objects to abstract classes but we can create objects to their sub classes. From sub class constructor,
there will be an implicit call to super class constructor. That's why abstract classes should have constructors. Even if you don't write constructor for your abstract class, compiler will keep default constructor.

==

8. Can we declare abstract methods as static?

>> **No**, **abstract method cannot be static** as static methods should have a body and abstract methods should not.

==

9. Abstract classes can be nested. True or false?

>> **True**, Abstract classes can be nested i.e an abstract class can have another abstract class as it's member.

===

10. Can we declare abstract methods as synchronized?

>> No, abstract methods can not be declared as synchronized. But methods which override abstract methods can be declared as synchronized.

===

11. Can we declare local inner class as abstract?

>> Yes, Local inner class can be abstract.

===

12. Can abstract class have constructors in Java?

>> Yes, abstract class have constructor.

===

13. Can abstract class implements interface in Java? do they require to implement all methods?

>> Yes, an abstract class can implement an interface by using the implements keyword. Since they are abstract, they don't need to implement all methods. It's good practice to provide an abstract base class, along with an interface to declare Type.

===

14. Can abstract class have static methods in Java?

>> Yes, an abstract class can declare and define static methods, nothing prevents from doing that. But, you must follow guidelines for making a method static in Java, as it's not welcomed in a object oriented design, because static methods can not be overridden in Java. It's very rare, you see static methods inside abstract class, but as I said, if you have very good reason of doing it, then nothing stops you.

===

INHERITANCE

1. What is Inheritance in Java?

>> one class acquires the property of another class is called inheritance.

==

2. What are different types of Inheritance supported by Java?

>> Single Inheritance, Multilevel Inheritance, Multiple Inheritance, Multiple Inheritance.

==

3. Why multiple Inheritance is not supported by Java?

>> To reduce the complexity and simplify the language, multiple inheritance is not supported in java.
Consider a scenario where A, B, and C are three classes. The C class inherits A and B classes. If A and B classes have the same method and you call it from child class object, there will be ambiguity to call the method of A or B class.

==

4. Why Inheritance is used by Java Programmers?

>> Inheritance is used for code reuse and leveraging Polymorphism by creating a type hierarchy.
It's better to use Inheritance for type declaration but for code reuse composition is a better option because it's more flexible.

==

5. What is the difference between Inheritance and Encapsulation?

>> Inheritance is an object oriented concept which creates a parent-child relationship.
On the other hand, Encapsulation is an object oriented concept which is used to hide the internal details of a class

==

6. What is the difference between Inheritance and Abstraction?

>> Abstraction is an object oriented concept which is used to simply things by abstracting details. It helps in the designing system. On the other hand, Inheritance allows code reuse. You can reuse the functionality
you have already coded by using Inheritance.

7. What is the difference between Polymorphism and Inheritance?

>> Inheritance	Polymorphism
Inheritance is one in which a new class is created (derived class) that inherits the features from the already existing class(Base class).	Whereas polymorphism is that which can be defined in multiple forms.
It is basically applied to classes.	Whereas it is basically applied to functions or methods.
Inheritance supports the concept of reusability and reduces code length in object-oriented programming.	Polymorphism allows the object to decide which form of the function to implement at compile-time (overloading) as well as run-time (overriding).
Inheritance can be single, hybrid, multiple, hierarchical and multilevel inheritance.	Whereas it can be compiled-time polymorphism (overload) as well as run-time polymorphism (overriding).
It is used in pattern designing.	While it is also used in pattern designing.

4] What is Constructor?

Constructor are special member of a class. Constructor are used to initialize data member(variable)of class & to load non-static members into object. At the point of constructor declaration below points need to be follows:-

Constructor Name should as class name.
Any Number of constructor declared inside the class but the constructor name must be as similar to class name with different arguments.
Constructor are classified into two types

Default constructor :- If constructor is not declared in JAVA class then at the time of compilation, compiler provides constructor for the class which is called as default constructor .If Programmer has declared the compiler will not provide default. The constructor which is provided by compiler at the time of compilation is known as default constructor.
User Defined Constructor :- If Programmer is declaring constructor in class then it is consider to be as user defined constructor.
It classified into two types:- a)Zero Arguments b) With arguments

5]What do you mean by User Defined Exception?

User Defined Exception or custom exception is **creating your own exception class and throws** that exception using 'throw' keyword. This can be **done by extending the class Exception**. The keyword
“throw” is used to create a new Exception and throw it to the catch block.

6] Difference between static and non-static methods

Static method uses complie time binding or early binding. **Non-static method** uses run time binding ordynamic binding. A **static method** cannot be overridden being compile time binding. A **non-static method** can be overridden being dynamic binding.

===

7] What is a super keyword in java?

The super keyword refers to **superclass (parent) objects**. It is **used to call superclass methods**, and toaccess the superclass constructor. The most common use of the super keyword is to eliminate the confusion between superclasses and subclasses that have methods with the same name.

===

8] What is casting & its types ?

Converting one type of information in to another type of information is known as casting.

The casting having mainly two types as below :-

A} Primitive   B} Non-primitive

A} Primitive :-

Converting one type (datatype) information into another datatype information is known as primitive casting.

Primitive casting have three types :-

Implicit Casting
Explicit Casting
Boolean Casting

Implicit Casting:- Converting Lower datatype information into higher datatype information is known as implicit Casting.

B] Non-Primitive Casting:-

Converting One type of class into another type of class is known as non-primitive casting.

It has two types: a) UP Casting   b) DOWN Casting

UP Casting : -

Father S = new son();

Assigning subclass property into superclass is known as up casting. Before preforming upcasting operation inheritance operation take place first. After performing inheritance the property which are present in super class comes into subclass. In the subclass comes into subclass.At the time of upcasting the property which are inheritance from super class are only eligible for operation. The new property which where declared inside subclass are not eligible for upcasting operation.

===

9] Can we declare many interfaces object class inside the interface class.

Yes, **you can** define a **class inside** an **interface**. **In** general, if the methods of the **interface** use this **class** and if **we** are not using it anywhere else **we will declare** a **class** within an **interface**.

==

10] What do you mean by User Defined Exception?

User Defined Exception or custom exception is **creating your own exception class and throws** that exception using 'throw' keyword. This can be **done by extending the class Ex**“throw” is used to create a new Exception and throw it to the catch block.

==

11] How can you use interface and how it is different from Abstract class?

Abstract class may have Abstract and concrete methods, and there is not any compulsion in adding
abstract method in abstract class. But **in Interface, we do have only abstract methods** and we don’t need
to write abstract keyword in Interface this is by default public and abstract.

==

12] What is Run time polymorphism

Run-Time Polymorphism: Whenever an object is bound with the functionality at **run time**, this is known as **runtime polymorphism**. The **runtime polymorphism** can be achieved by method overriding. Java virtual machine determines the proper method to call at the **runtime**, not at the **compile time**.

==

13] Why the main method is static?

Java **main**() **method** is always **static**, so that compiler can call it without the creation of an object or before the creation of an object of the class
Static method of a class can be called by using the class name only
without creating an object of a class.

==

14] What do you mean by Static keyword in Java?

Static means it is at class level not at instance level, we have static method, static variable & static inner class. When we have any variable as static so it **will remain same for all the instance of our classes**, and static/Private/Final methods can’t be over-ridden like **if we have initialized any method as Static so we cannot override it in any child class**

==

15] Can you tell me Oops concepts and relate it with your Framework?

We have Polymorphism, Inheritance, Encapsulation and Abstraction in Oops. So, we will start with

1] DATA ABSTRACTION

Data Abstraction means to handle complexity by hiding unnecessary details from the user. In java, abstraction is achieved by interfaces and abstract classes**. We can achieve 100% abstraction using interfaces.**

In Selenium, WebDriver itself acts as an interface. Consider the below statement: WebDriver driver = new ChromeDriver();

We initialize the Chrome Browser using Selenium Webdriver. It means we are creating a reference variable (driver) of the interface (WebDriver) and creating an Object.

We can apply Data Abstraction in a Selenium framework by using the Page Object Model design pattern. We define all our locators and their methods in the page class.

We can use these locators in our tests but we cannot see the implementation of their underlying methods. So we only show the locators in the tests but hide the implementation. This is a simple example of how we can use Data Abstraction in our Automation Framework.

2] ENCAPSULATION

Encapsulation is defined **as the wrapping up of data under a single unit**. It binds together code and the data it manipulates.

Encapsulation can be achieved by: Declaring all the variables in the class as private and writing public methods in the class to set and get the values of variables.

All the classes in an Automation Framework are an example of Encapsulation.

In Page Object Model classes, we declare the data members using @FindBy and initialization of data members will be done using Constructor to utilize those in methods.

3] INHERITANCE

It is one of the oops principle where one class Aquire the property of another class with the help of extend keyword.

The class from where property are inhering or aquiring is known as Supper class.

We can apply Inheritance in our Automation Framework by creating a Base Class to initialize the WebDriver interface, browsers, waits, reports, logging, etc. and then we can extend this Base Class and its methods in other classes like Tests or Utilities. This is a simple example of how we can apply Inheritance in our framework.

4] POLYMORPHISM

Polymorphism allows us **to perform a single action in different ways**. In Java polymorphism can beachieved by two ways:

Method Overloading:

When there are multiple methods with **same name but different parameters** thenthese methods are said to be overloaded.
Methods can be overloaded by change in number of arguments or/and change in type of arguments.
In Selenium Automation**, Implicit wait is an example of Method Overloading**. In Implicit wait we usedifferent time stamps such as **SECONDS, MINUTES, HOURS etc.**

Method Overriding:

If Subclass has the **same method as declared in the parent class it is known as method overriding**
In Selenium Automation, Method Overriding can be achieved by overriding any WebDriver method. Forexample, we can override the findElement method
In assertion we have used overload because in assertion we used to like asset.true(actual, expected) andsecond time we can use same assert.true(actual, expected, message.

==

16] How to call static method and variable in java?

Direct calling, Calling by class name.

==

17] Can I access Static method by using object reference?

Yes we can, but we got one warning that you need to access it via Direct or By class name.

==

18] How to call non-static method and variable in java?

For calling non static method we need to create object first

==

19] Can we overload & override main method?

Overload-Yes
Override-No

20] What do you mean by wrapper class and how will you do data conversion?

Wrapper class in java are used for data conversion. In data conversion if user wants to convert Int tostring, String to int, Boolean, double then we use Wrapper class.
integer.parseInt(); - To convert string to Integer Double.parseDouble(); - To convert string to Double Boolean.parse Boolean(); - To convert string to BooleanString.valueof(); - To convert Integer to String.

==

21] Can you convert string a =”110a” in integer?

No we got NumberFormatException while converting the above string.

==

22] What do you mean by Call by Value & Call by Reference in Java?

Call by value means suppose we have created one sum method with input parameter int a, int b. So while calling the creating the object and running we provide values that is know as call by value.

==

23] What is a static keyword in Java?

In the **Java** programming language, the **keyword static** indicates that the particular **member belongs to a type itself, rather than to an instance of that type**. This means that **only one** instance of
that **static** member is created which is shared across all instances of the class

==

24] What do you mean by Exceptions in Java?

Exception is like **any interruption in our normal flow**. Like if we are running anything and we got issues in our script this is we called exception, we have 2 types of exception Run Time & Compile Time. (checked & Unchecked exceptions)

==

25] Diff between Abstract class & interface?

A class declared with abstract keyword is known as abstract class.it is nothing but an incomplete method , where a programmer can declared complete as well as incomplete method.
An incomplete method where method declration is present but method definition will be absent.

Abstract class can **inherit another class using extends keyword** and implement an interface. Interface can inherit only an inteface. Abstract class can be inherited **using extends keyword**. **Interface can only be implemented using implements keyword.**

==

26] What is static and non-static?

In non-static method, the method can access static data members and static methods as well as non- static members and method of another class or same class. Binding process.
Static method uses compile time or early binding. Non-static method uses runtime or dynamic binding. Overriding.

==

27] Can you tell me about StaleElementReferenceException?
Stale means old or decayed, here it sounds like element which was present on that page is no longer there or decayed. To handle this, we can refresh the webpage before pointing to that element. We canwrite script for waiting via explicit wait by writing expected condition.refresh. Or we can go with page object model in that we can over come this stale element exception.

==

28] What is the difference between scenario & scenario outline?
When we have single scenario and we need to run it one time at that place we use Scenario.
If you want some parametrization or Data Driven testing at that time, we can use scenario outline where we have to use Example keyword like if we are running this scenario for 3 different data set like username& pass. so, it will run 3 times.

==

GIT AND JENKINS QUESTIONS

1] You have worked in Jenkins can you tell me how you have created jobs in Jenkins?

We have separate Dev-Ops Team to create Jenkins jobs at broad level but we also have access to jenkins, so we have created jobs for our internal purpose.
For creating any job we **have click on create new job**->inside that give **name of your job->select freestyle project->then add.**
Beside that we can provide description of our project and in source code management we can choose Git-> provide repo url ->after that provide some schedule if you want to run the job on any specific schedule time.-> select window batch command-file location-save-click on build now for running. After triggering we can check log in console.

==

2] Explain Git workflow.

Step 1: Set up a Github Organization. ...
Step 2: Fork Organization Repository to Your Personal GitHub. ...
Step 3: Clone the Repository to Your Local Machine. ...
Step 4: Create a Branch for your Working Files. ...
Step 5: Set Remote Repository to the GitHub Organization. ...
Step 6: Get Coding!
Step 7: Pull the Most Recent Files From the Organization Repo **Step 8**: Merge the Master Branch Into the Feature Branch **Step 9:** Push Your Code to your GitHub Repo
Step 10: Make a Pull Request to the Organization Repo
==

3] Does Jenkins require a local system for CI?

It is a server-based application and requires a web server like Apache Tomcat
==

4] How to solve Merge conflict in GIT?

As we are only 2 tester working on this project, if we have any merge conflict I **used to pull all the latest file/scripts to my local system.** Then I will analyze the difference between that particular file and merge file. After that I will check with my team member whether all his imp things are covered then I will add my steps and push the script to the central repo.
==

5] How to set up Jenkins ?

Step 1 − Go to the Jenkins dashboard and Click on New Item

Step 2 − In the next screen, enter the Item name, in this case we have named it Helloworld. Choose the 'Freestyle project option'

Step 3 − The following screen will come up in which you can specify the details of the job.

Step 4 − We need to specify the location of files which need to be built. In this example, we will assume that a local git repository(E:\Program) has been setup which contains a 'HelloWorld.java' file. Hence scroll down and click on the Git option and enter the URL of the local git repository.
Note − If you repository if hosted on Github, you can also enter the url of that repository here. In addition to this, you would need to click on the Add button for the credentials to add a user name and password to the github repository so that the code can be picked up from the remote repository.

Step 5 − Now go to the Build section and click on Add build step → Execute Windows batch command

Step 6 − In the command window, enter the following commands and then click on the Save button.

Javac HelloWorld.java Java HelloWorld

Step 7 − Once saved, you can click on the Build Now option to see if you have successfully defined the job.

Step 8 − Once the build is scheduled, it will run. The following Build history section shows that a build is in progress.

Step 9 − Once the build is completed, a status of the build will show if the build was successful or not. In our case, the following build has been executed successfully. Click on the #1 in the Build history to bring up the details of the build.

Step 10 − Click on the Console Output link to see the details of the build

==

6] Have you used GIT in your project can you explain about it?
Yes I have used GIT, It is a **version control tool**. Where we can maintain our central repo. we used to manage our code via GIT only. We use **Git to maintain our project in our local system.** So, if someone liketo work on that project I need to send complete update copy to him and after that he can work on that. There are chances that single project is handled by multiple teams across the globe. So, it will be difficultif we won't use GIT.

Pls also mention GIT command.
Git status- which shows status of all the files,if we have some files which is not yet added to our repo so itwill give us untracked file.
After that we can use GIT add command after adding it will added to particular index and we can committhis file using Git Commit-(Message) we can commit this untracked file. Also we have Git Merge, Git Post,Git Pull, Git It in etc.

```
git add [file]
git status
git branch
git merge [branch name]
git push [variable name] master
git pull [Repository Link]
git reset [file]
git commit -m "[ Type in the commit message]"
```

STRING ONE LINE QUE.

java.lang.String class is used to create a string object.
Different String methods:

compareTo - The Java String compareTo() method is used for comparing two strings lexicographically.
boolean equals() - The java string equals() method compares the two given stringsbased on the content of the string (case sensitive)
String concat() – concat two strings
boolean equalsIgnoreCase() - The java string equals() method compares the two givenstrings based on the content of the string (not casesensitive)
char charAt() – index position - The java string charAt() method returns a char value atthe given index number.
toUpperCase() – convert to upper case
toLowerCase() – convert to lower case
trim() – remove spaces from both sides of string
substring() -- returns part of string
boolean startWith() – ends with specified suffix or not

==

java file mai ek hi public class hoti hai, uske alava aur classes v ho sakti hai par public ek hi.
If we need to create one variable for multiple values, we need to use Array concept.
Int marks[] = new int[5]
Array can store only homogenous data, int for int array, string for string,
If we need to add heterogeneous data in array, we need to create object array.
Object a[] = new Object[5]; now we can add different data type objects.
Array is fixed in size, which we define while creating.
If we try to access index value >= given index value, we got arrayOutOfBundException.
We can add new elements in run time under collections while in array we cannot.
Collection is a group of objects. To represent this we need certain interfaces and classes.
Common operations we generally do on collections are adding objects, removing objects & finding object.
Collection (I) is called collection interface having methods which are common throughout all collections.
Collections is basically a class from java.util package which contains some methods whichwe can use for collection objects.

List (I) is child of collection(I). In list Insertion order is preserved and duplicates are allowed.
ArrayList, LinkedList, Vector these are different classes which implements **List Interface**.
Set(I) is child of collection(I). Insertion order is not preserved & duplicates not allowed.
HashSet, Linked Hashset these are different classes which implements **Set Interface**.
Queue(I) is child of collection(I). We used it when we need prior to processing means first infirst out concept.
priorityQueue is class which implements **Set Interface.**
There is one independent interface known as **Map(I).** In Map(I) objects are created with **keyand value** pair. Key cannot be duplicate, but Value can be.
Hashmap, Linked Hashmap, Hash Table these are different classes which implements **Map Interface.**
Whatever methods present in Collection(I) are also present in their child interface i.e List,Set, Queue.
add(object o), addAll(Collection c), remove(Object o), removeAll(Collection c), retainAll(Collection c) these are some methods of Collection Interface.
clear(), isEmpty(), size(), contains(), conatinsAll(), toArray() are also some methods.
In List **index** play an important role because with the help of index only we can find**duplicates elements**.
add(index , object), get(index), set(index, object) are methods of List Interface.
ArrayList al= new ArrayList(), it allows heterogenous objects also.
ArrayList<Str> al= new <str>(), now it can store objects of string only.
Collections.Sort(al) , Collections.Shuffle(al) This will sort & shuffle the objects of arraylist.
We can read the data with for loop, for each loop, iterator () method.
JVM have 2 types of memories **Static pool -static data, heaps-Non static data**

OOPS ONE LINE CONCEPTS

We can create multiple classes inside one main class.

Class name cannot be a keyword.

In **Polymorphism**, we can create many methods with same name, differentiating with input parameter. Void Walk(){}, void walk(int steps){} . We cannot create duplicate methods or one method into another.

Compile time polymorphism, tell which method is called before running it.

By applying **static** it means now it became class property not object one and it applicable to all objects throughout the class.

Static methods are accessed by class, **Non Static methods** are accessed by objects.

Constructor in java is to make objects, we can create constructor but there is one default constructor in java, like **students a = new students();** highlighted is the default constructor.

Constructor are non static methods. Constructor can be **parameterized**

We can create our own constructor like **students b= new students(int rollNo ,String name){}**

this() keyword is used to call one constructor by another.

this() keyword also used in agar 2 alag alag methods mai same parameter name hai , to conflict na ho isliye **this.name**, **this.rollNo** use krte h. this is known as method overloading bcs you are using one method parameter in second one.

Inheritance ka matlab hai ki Parent class ki property ko hum child class mai bhi use kar sake.

super() keyword parent class ke constructor ko child mai le ata hai , means ab agar hum Sci class ka object baneyenge to students class ka v ek object banega.

Java ke andar jitne v by default objects hote hai unke ek parents class hoti hai **object.**

Run time polymoíphism tell which method is called while running it.

Public, private, protected inko access modifier bolte hai.

Agar hamne kisi method ko Private bnaya hai to vo bas usi class mai hi use ho skta hai aur kahi nahi.

Abstraction iska matlba hai ki hum unnecessary info ko user se hide kre taki usko main functionality par focus rahe na k internal logic pr.

abstract keyword agar kisi class ke sath lagaya hai to hum uska objects nahi bna sakte.

Aur agar hum kisi class ko abstract banate hai to hame usme kuch logic dene ki jaroorat nahi hai. Aur ye ek imp property hai abstract method ki.

Abstract method bannna hai to uski class ko bhi abstract bannna padega.

Interface methods are by default public and abstract.

Abstract method ke andar hum concrete functions bna sakte hai , par **interface** ke andar kuch nahi bna sakte.

Interface ko use karne ke liye hum **implements** keyword use karte h.

Java mai ek class ke multiple parent nahi ho sakte bcs java multiple inheritance support ni krta. Iske liye hum **interface** ka use karte hai.

Literals means are the string values, integer values, comments, keyword.

INTERVIEW QUESTIONS

1] Explain Data Types?
2] What is Static and Non Static method ?
3] Explain variables ?
4] What is String class?
5] Difference between String. String Builder, String Buffer ?
6] Explain coupling ?
7] Constructor ?
8] What is Casting and its types?
9] What is This and Supper Keyword?
10] What is Abstract ?
11] What is Inheritance and why it is used in Java?
12] Different between Inheritance and Encapsulation?
13] Different between method Over Riding and Method Overloading?
14] Can we achieve multiple Inheritance in java .How you can achieved multiple inheritance in java ?
No….
15] Can we declare Abstract class as static?
16] Give real life example of Polymorphism? Explain
17] Diff between Interface and Abstract ?
18] What is interface ?
19] Explain All access Specifier ?
20] Can we declare constructor as private?
21] What is difference between Final. Finalize and Finally in java?
22] Can we create object of Interface?