**2 weeks**

**100 regression (manual testing 30 mins)**

**People required for executing 100 testcases in 10 days (5 persons)**

**Selenium is an open-source tool** that automates web browsers. It provides a single interface that lets you write test scripts in programming languages like Ruby, Java, NodeJS, PHP, Perl, Python, and C#, among others.

**Introduction:**

Selenium is an open source automation testing tool.

It is used exclusively for web based applications

You can work on multiple operating systems using selenium

* Platforms Supported by Selenium
* Windows
* OS X
* Linux
* Solaris

**Following languages are used with selenium.**

* Java
* C#
* Ruby
* Python
* PHP
* Pearl

**Selenium Browsers Support:**

* Internet Explorer
* Firefox
* Chrome
* Safari
* Edge browser

**Interview Questions**

1. What makes Selenium Unique from other Automation tools?

Before selenium came into market we had QTP/UFT

1. Explain WebDriver Architecture.
2. Why not older Version Selenium RC 1.0?
3. What are different Versions of Selenium?
4. What are different browsers does Webdriver Support?
5. In how many languages we can write Selenium code?

**\*\*Selenium Features\*\***

* Selenium is open Source Automation Testing tool
* It is exclusively for Web Based applications.
* Selenium supports multiple browsers -   
  Chrome, Firefox, Internet Explorer, Safari
* Selenium works with Multiple Platforms   
  Windows, Apple OS X, Linux
* Selenium can be coded in multiple languages -   
   Java, C#, Python, Javascript, Python, php,Ruby
* Difference between Selenium and Webdriver?

Selenium is a pack of tools:

Ide

Webdriver

GridT

**Steps to follow to install and Selenium :**

1. **Install Java**
2. **Setup Java home(Java path) in System Variables**

* Right Click on **This PC**
* Click on **Properties**
* Click on Advanced system settings
* Click on Environment variables(present at the end)
* Click on New button present under system variables section
* Enter Variable name as **“JAVA\_HOME”**
* **For Variable value follow below:**
* Go to this path : **C:\Program Files\Java**
* We should see both jdk and jre folders
* Get into JDK folder
* Copy folder path and enter in Variable value field
* Click OK
* **Follow below for setting up PATH:**
* Select **Path** option available from the system variables section and click on **edit**
* Click on **new** button in edit environment variables popup and enter this path : %JAVA\_HOME%\bin
* Click on **OK, OK, OK**

1. **Install Eclipse (editor) (Eclipse IDE for Java Developers)**
2. **Create a sample project (Maven with selenium dependencies)**

Selenium WebDriver Architectue Simplified:



* After you trigger the Test, complete Selenium code (Client) which we have written will be converted to Json format
* Generated Json is sent to Browser Driver (Server) through http Protocol

Note: Each browser contains a separate browser driver

* . Browser drivers communicate with its respective browser and executes the commands by interpreting Json which It received on the browser.
* Browser Driver receives responses back from the browser and it sends Json response back to Client.

Creating a sample project:

1. Create a maven project
2. Open pom.xml file and add selenium java dependency in there, click save
3. You should see Maven Dependency folder created in your project
4. Instantiate driver with the below code:

**System.*setProperty*("webdriver.chrome.driver", "Path of chromedriver.exe file");**

**JAVA\_HOME C:\Program Files\Java\jdk1.8.0\_341**

**WebDriver driver = new ChromeDriver(); (driver as object or reference variable)**

**Cross Browser Testing:**

Launching tests different browsers.

**Core Java:**

**Variables and datatypes:**

**Integers: ………………… -5, -4, -3, -2 ,-1, 0, 1, 2, 3, 4, 5 …………….infinite**

**Core Java Basics for Automation:**

* **Variables and Datatypes in Java**
* **Working with Arrays**
* **Loops & Conditions**
* **Strings and its functions**
* **Array Lists & its Operations.**
* **Conversion of Array to List**
* **Methods (Declaration)**
* **Accessing Methods in class**
* **Static keyword**

**Arrays:** Java array is **an object which contains elements of a similar data type**. Additionally, The elements of an array are stored in a contiguous memory location. It is a data structure where we store similar elements. We can store only a fixed set of elements in a Java array.

/\*

\* for(int i = 0; // initialization i<arr1.length;// condition check i = i+1 //

\* increment operation ) { System.out.println(arr1[i]); // statement }

\*/

Increment operators;

i++ (i=i+1 )

++i (i=i+1 )

**String Array:**

**Locators:**

ID

Name

Class name

LinkText

PartialLinkText

Xpath – absolute & relative xpath (Parent and descendant concept )

Css selector

Html nodes:

1. Tagname
2. Attribute name and value
3. LinkText(Optional)

<input - tagname

type="text" id="twotabsearchtextbox" value="mobile" name="field-keywords" autocomplete="off" placeholder="" class="nav-input nav-progressive-attribute" dir="auto" tabindex="0" aria-label="Search">

type="text" is an attribute

id="twotabsearchtextbox"

value="mobile"

Best Sellers

<input type="text" class="inputtext \_58mg \_5dba \_2ph-" data-type="text" name="firstname" value="" aria-required="true" placeholder="" aria-label="First name" id="u\_0\_b\_4l">

What are anchor tags?

If you have tagname as “a” in a html node of a webelement, such tagnames are called tags. And Anchor tags mostly have href attributes and we use linktext or partial link text locators to find element with href attribute.

https://www.amazon.in/

[**https://www.amazon.in/b/?node=16192220031&ref\_=map\_1\_b2b\_GW\_FT**](https://www.amazon.in/b/?node=16192220031&ref_=map_1_b2b_GW_FT)

**Xpath:**

**Absolute xpath**

**Relative xpath:** Always a relative xpath starts with two forward slashes //

**1)Find element by attribute**

**2)Find element by text**

**Find element by attribute:**

<a – tagname

Attributename = “Attribute value”

href="/gp/bestsellers/?ref\_=nav\_cs\_bestsellers"

class="nav-a "

tabindex="0"

data-csa-c-type="link"

data-csa-c-slot-id="nav\_cs\_0"

data-csa-c-content-id="nav\_cs\_bestsellers"

data-csa-c-id="wv3lno-rpdec6-jicdi1-qptw25"

>Text<

>Best Sellers</a>

Syntax for xpath by attribute: relative xpath

**//tagname[@AttritubeName= ‘AttributeValue’] (is written by attribute name and value)**

**//a[@data-csa-c-content-id= nav\_cs\_bestsellers’]**

**<div role="treeitem" class="\_p13n-zg-nav-tree-all\_style\_zg-browse-item\_\_1rdKf \_p13n-zg-nav-tree-all\_style\_zg-browse-height-small\_\_nleKL"><a href="/gp/bestsellers/mobile-apps/ref=zg\_bs\_nav\_0">Apps for Android</a></div>**

**Find element by attribute:**

(//div[@role="treeitem"])[3]

//input[@placeholder="Username"]

//input[@type="password"]

//input[@value="rmbrUsername"]

**Find element by text:**

**//tagname[text()= ‘text’]**

**Examples:**

(//button[text()='Visit us'])[2]

(//h1[text()='Rahul Shetty Academy'])[2]

//h2[text()='Now India will say "Bought it on Amazon!"']

**Regular expressions in xpath:**

1. **contains**
2. **Starts-with**
3. **Ends-with**

EG: Facebook username (Dynamic applications)

id="u\_4\_b\_cj"

id="u\_2\_b\_n7"

id="u\_6\_b\_b0"

Web application are of twpo types:

1} static application – upon refresh the html attributes of any web element does not change.

2} Dynamic Application - upon refresh the html attributes of any web element will change.

//input[@id="u\_"]

**Regular expressions in xpath:**

1. **contains**
2. **Starts-with**
3. **Ends-with**

**Contains (attribute name): PartialAttributeValue’ is that part that is staying constant even after refresh**

**//tagname[contains(@AttributeName, ‘PartialAttributeValue’)]**

**Example:**

**(//input[contains(@id,"u\_")])[1]**

**Contains (with text())**

**//tagname[contains(text(), ‘Partial text’)]**

**//div[contains(text(),'Create a new')]**

**//a[contains(@data-csa-c-content-id,'\_cs\_best')]**

**//a[contains(text(),'st Sell')]**

**Starts-with**

**//a[starts-with(@data-csa-c-content-id,'nav\_cs\_bes')]**

**//a[starts-with(text(),'Best')]**

**Ends-with**

**//a[ends-with(text(),'t Sellers')]**

**//a[ends-with(@data-csa-c-content-id, ‘cs\_bestsellers')]**

**Parent and Child/descendants:**

**Parent:**

**(//div[@class="error-message-fixed "])[1]/parent::div/parent::div/parent::div**

**(//div[@class="error-message-fixed "])[1]/../../..**

**/.. = /parent::div**

**Child/descendants:**

**(//div[@class="clearfix search-wrap"]/div/div/input)[3]**

**//div[@class="clearfix search-wrap"]/div/div/input[@id='onward\_cal']**

**//div[@class="clearfix search-wrap"]/descendant::input[@id='src']**

**//div[@class="clearfix search-wrap"]/descendant::label[@for='src']**

**ArrayList:**

**Strings**

**Class:**

**Class is a collection of different types of variables and methods**

**Trim() method eliminates spaces at the start and end of a string**

**Conditional loops in Java:**

**ForLoop**

**(Enhanced for loop) For each loop**

**While**

**Do while**

**Nested Loops**

**Java is Object Oriented Programming System – (OOPS)**

**Structure:**

* **Class**
* **Object**
* **Method**
* **Attributes**

**Principals of OOP**

* **Inheritance**
* **Abstraction**
* **Polymorphism**
* **Encapsulation**

**Interface:**

**Interface is similar to a class but it contains only method declaration but not definition or body.**

**All the methods which are written inside a interface are abstract – abstract methods are those that does not have body declared.**

**Implements is the keyword used to import methods from interface.**

**How to achieve interface implementation.**

**By using Implements keyword.**

**Abstraction:**

**Abstraction is a process of hiding the implementation details from the user, Only the functionality will be provided to the user. The user will have the information on what the object does instead how it does.**

**Rules:**

**At least one methods should be abstract**

**Class should start with keyword abstract**

**Implementation is done with the help od extends keyword**

IndigoooFlights flight1 = **new** IndigoooFlights();

WebDriver driver = **new** ChromeDriver();

Why driver object in the above code is referring to WebDriver interface instead of ChromeDriver class

Because we require our code to run in all the browsers. So we have to crate a code which has methods common to all the browsers. So Webdriver interface will have methods common to all the browsers.

What kind of object instantiation is possible?

ChildClass object = new ChildClass();

ParentClass object = new ChildClass();

ParentClass object = new ParentClass(); (not possible for interfaces and abstract class)

Eg : FlightAbstractClass flight3 = FlightAbstractClass();

Differences between interface and abstraction:

Interface:

All the methods are by default abstract

Implementation of interface is done by implements keyword

**Interface should start with keyword Interface**

ParentClass object = new ParentClass(); not possible

Abstraction:

Atleast one method should be abstract

Implementation of abstract class is done by extends keyword

**Class should start with keyword abstract**

ParentClass object = new ParentClass(); not possible

Inheritance:

**Parent class - Vehicle:**

**petrolEngine**

**desielEngine**

**gears**

**braking**

**Child class -**

**Type**

**Colour**

ChildCar swift = **new** ChildCar();

swift.braking();

swift.gears();

swift.desielEngine();

swift.colour();

swift.type();

ChildCar swift = **new** ChildCar();

And you have method with same name in both parent and child class.

This java will pick method that is from child class because the child class object is referring to child class

ParentClass swift1 = **new** ChildCar();

swift1.braking();

swift1.gears();

swift1.desielEngine();

swift1.colour();

ParentClass swift1 = **new** ChildCar();

And you have method with same name in both parent and child class.

Then java will pick method that is from child class though the child object is referring to parent, the common methods is overwritten after inheritance

And THIC object will have access only to the methods coming from parent

System.***out***.println();

ParentClass swift2 = **new** ParentClass();

swift2.braking();

swift2.gears();

swift2.desielEngine();

swift2.colour();

ChildCar swift3 = **new** ParentClass(); this declaration is not possible because Parent object cannot refer to child class because parent class is inherited by child but not the reverse way.

Inheritance types in Java:

1. Single Inheritane
2. Multilevel
3. Hierarchal
4. Hybrid – Not possible (It is the combination of Multiple and Hierarchal) Diamond problem in Java
5. Multiple – Note Possible

**Single Inheritane:**

**One parent and one child**

Parent🡪Child

**Multilevel**

Parent--🡪 Child-🡪 GrandChild

**Hierarchical**

Parent🡪child1

Parent🡪child2

Parent🡪child3

**`Hybrid**

**It is the mix of more than one type of inheritance**

Parent🡪child1🡪 GrandChild

Parent🡪child2

Parent🡪child3

Mutiple Inheritance is not subborted in java

Parent1 , Parent2 🡪 child

**public** **class** Child1 {

// Child1 methods

}

**public** **class** Child2 {

// Child2 methods

}

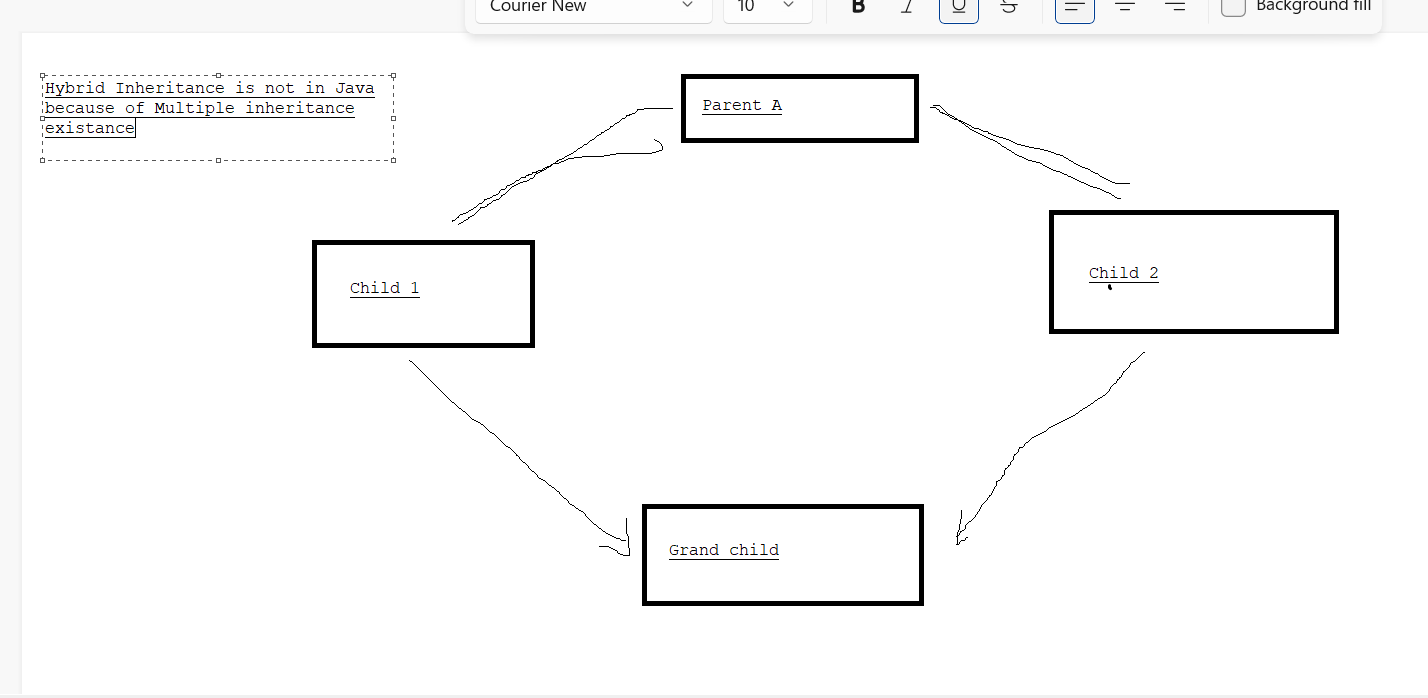
**public** **class** MultipleInheritance **extends** Child1, Child2 {

//MultipleInheritance methods

}

Arrays

**Hybrid Inheritance/Diamond Problem:**

****

Polymorphism means "**many forms**",

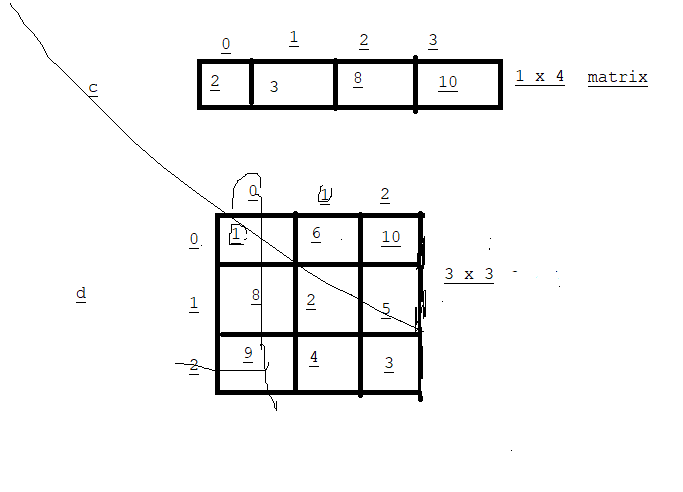
**Method Overloading:**

If there are multiple methods in a class and those have same name with different attributes or data types. Such methods are called overloaded or this kind of declaration is called over loading

Method overloading can only be done in same class

**Method Overriding:**

A method from parent class is Overridden in child class by just changing it’s definition or body. This achieved by inheritance. That means we should have at least two classes to perform overriding.

****

**Date**

**Simple Date Format**

**Calendar**

**Constructors:**

**It is a piece of code which is executed as soon as an object is created for a class.**

**Its name should be always same as class name**

**They will not have any return**

**Types of Constructors:**

1. **Parametrized**
2. **Default**
3. **Implicit**

{ 8, 6, 10 }, { 8, 2, 5 }, { 1, -9, 0 }

1. **Find out the max number:**
2. **Find out the min number from the maximum numbered row**

**Super keyword and its applications** – It is used to point or call parent variables, methods & Constructors

* With variable
* With method
* With Constructors

**public** ChildDemo() {

**super**(5);// super keyword should be written at first when used inside constructor

System.***out***.println("I am in Child Constructor");

* }

**This keyword and its applications** – This keyword is used to point to variables defined globally at class level.

Types of vaiables:

* Instance Variables
* Local Variables
* Class Variables

1. Static keyword and Its usage
2. Final

Final methods

//

**final** **void** getData() {

System.***out***.println("I am a final methods");

}

this, super, public, private, protected, try, catch, finally, final, static, extends, implements, polymorphism, inheritance, interfaces, strings

**Packages: set of classes, interfaces**

**Java.lang - java compliler – default package**

**Java.util -** collection packages

**import** java.util.ArrayList;

when you are creating object for a class which is from different package. Then import statement must be used.

If the classes are in same package. Import is not required.

**Packages: We are automating Amazon:**

Package1:

Buynow: Buying a product by entering card details.

Package2:

public, private, protected, try, catch, finally,

Access Modifiers:

Default, public, private, protected,

Public:

When you declare a method/variable as public. It can be accessed anywhere in your project.

Default:

When a method is declared as default. It can only be accessed within the package.

Private:

When a method is declared as private, it can only be accessed within the same class.

Protected:

When a method is declared as protected.

1. It can be accessed within same package with the help of objects.
2. This method is accessible in different packages by using inheritance

Exceptions Handling:

try, catch, finally,

**Arrays:** Java array is **an object which contains elements of a similar data type**.

**Java Collections Framework:**

**Is a collection of interfaces and classes which helps in storing and processing the data efficiently.**

**Collections objection contains elements of same or different datatypes. We can make collections object to store strictly single data types**

**Collection has three types processes or classes:**

* **List**
  + **ArrayList - discussed**
  + **LinkedList – please do research**
  + **Vector - please do research**
* **Set**
  + **HashSet - discussed**
  + **LinkedHashSet - please do research**
  + **TreeSet - please do research**
* **Map**
  + **HashMap**
  + **Hash Table**
  + **TreeMap**

**Difference between Hash Map and HahTable**

1. **Synchronization/Thread safe**
   1. **HashMap – not synchronised**
   2. **HashTable – synchronized**
2. **Null keys & Vaalues**
   1. **HashMap – accepts null keys and values**
   2. **HashTable – does not accepts null keys and values**
3. **Iterating the values**
   1. **Hashmap - Iterator**
   2. **HashTable - Enumerator**

**List**

**ArrayList:**