#### Q1) Why should you learn test automation? Q2) Why modern jobs now require people with automation skills? Q3) Why should you learn programming? Q4) Why is it important to pick one tool and learn it completely and then move to next one? Let's discuss!! Obviously every one from the industry must know - moving from manual background to automation is challenging, but still there are lot of people who invest time and money in learning automation skills right? But why? Because they know market and jobs have moved on and now they require much more. Skills of testing is important but applying those skills into test automation is a gold mark for our profiles! Now let's discuss some basic advantages of test automation:- 1) Build Validations takes less time and focus and automated reporta can be shared within team 2) Production readiness can be tested with scheduled jobs and then can be deployed based on results received 3) Testers can focus more on new features and defects fixed 4) Data Driven testing can help in overall reduced time of execution 5) Cross browser testing 6) Cross platform testing and so on.. Now how can you learn test automation? I am listing down simple steps that you can follow to learn it! 1) Pick a programming language - Learn it's basics (already shared a post with free resources to learn from, can review in my profile posts section) 2) Pick what you want to automate? Web? - Learn selenium Mobile? - Learn appium Window? - WinAppDriver API? Postman (already shared a post with free resources to learn from, can review in my profile posts section) (Framework building, reporting included) 3) Pick which version control you want to learn? My recommendation - GitHub 4) Pick a cloud service to do remote testing - my recommendation - AWS 5) Learn basics of Linux - because most the remote systems will be having - Linux installed 6) Learn a CD tool - my recommendation - Jenkins This is enough to get a high paying job with your existing testing skills and addition of automation skills. Note - The skills listed are in a specific order so you can learn in this order or its upto you! Also try to incorporate both web and API test automation for getting good job.

What do you think can be roles and responsibilities of QA Engineer?  
What can be different roles in QA?  
What does a Test Analyst do?  
  
Let's Discuss!  
  
I have kept a basic overview of each role listed below.  
  
QA Engineer --> Tests software to detect bugs and errors.  
- Checks whether a product complies with the requirements.  
- A detective who knows where the bugs can hide, even where no one expects them to.  
- Tests the system using attention, deduction, and sometimes special software.  
  
Test Analyst --> Guru of project documentation. The first one to decide what to test and how.  
- Knows exactly what the product should do.  
- Systemizes the information to ease the QA engineer’s life.  
  
Test Architect -->  Looks for ultimate solutions that will meet the client’s demands and align with the team’s resources.  
- Has a complete vision of the software system.  
- Knows every little feature and how it interacts with other features.  
  
Test Manager --> Takes full responsibility for the project’s success (or fail).  
- Prepares test strategy, defines the scope of work for other members, controls test execution.  
  
QA Team Lead --> The Supervisor.  
- May take part in any process mentioned above.  
- Usually just checks the status and manages the team.  
- Conducts interviews.  
- Hires and mentors new members.  
- Deals mostly with managerial tasks rather than tech tasks.

Types of variables in Java –

* **Local Variable:**  
  Defined within a method or a code block.  
  Scope: accessible within the method/block of code where it is declared.  
  Remains in memory as long as the method executes.  
  Does not require any special keyword.  
  Requires to be initialized before it is used.
* **Instance Variable:**  
  Defined outside a method at the class level.  
  Is accessible throughout the class.  
  Remains in memory as long as the object is in memory.  
  Does not require any special keyword but any access specifier (private, protected or public) can be specified. Typically, private or protected is used.  
  Is given default value based on its data type, so does not require to be initialized before it is used.
* **Static Variable:**  
  Defined outside a method at the #class level.  
  Is accessible throughout the class.  
  Remains in memory as long as #program executes.  
  Requires the static keyword to be specified. In addition, any access specifier (#private, #protected or #public) can be specified. Typically, public is used.  
  Is given default value based on its #datatype, so does not require to be initialized before it is used.

Want to improve your automation testing skills?

Before attempting automation testing, it is recommended to acquire knowledge of the following top 10 things:  
  
----------------------------------------  
  
 Programming:  
 A good understanding of programming concepts is essential for automation testing. Knowledge of a programming language such as Java, Python, C#, or JavaScript is crucial.  
-----------------------------------------  
  
 Test fundamentals:   
It's important to understand the fundamentals of software testing, including types of testing, test plans, test cases, and test scripts.  
----------------------------------------  
  
 Testing frameworks:  
 Familiarity with testing frameworks like JUnit, TestNG, NUnit, or MSTest is important as it can help in writing effective automated tests.  
--------------------------------------  
  
 Test automation tools:   
Knowledge of popular automation tools like Selenium WebDriver, Appium, TestComplete, or Robot Framework can be helpful in automating web or mobile applications.  
----------------------------------  
  
 Test management tools:   
Knowledge of test management tools like JIRA, HP ALM, or TestRail can help in managing test cases and test results.  
---------------------------------  
  
 APIs and Web Services:   
Understanding the basics of APIs and web services is important for automating APIs, web services, or microservices.  
---------------------------------  
  
 Version Control Systems:   
Knowledge of version control systems like Git, SVN, or TFS can be helpful in managing automation scripts and versioning.  
-------------------------------  
  
 Continuous Integration/Continuous Delivery (CI/CD):   
Understanding the basics of CI/CD and DevOps is important for integrating automation testing into the software development lifecycle.  
------------------------------  
  
 Database fundamentals:   
Basic knowledge of SQL and databases can be helpful in automating database testing.  
------------------------------  
  
 Problem-solving skills:   
Finally, problem-solving skills are crucial for automation testing. You need to be able to identify issues in the application and debug automation scripts effectively.  
--------------------------------

𝙬𝙝𝙖𝙩 𝙞𝙨 𝙮𝙤𝙪𝙧 𝙖𝙥𝙥𝙧𝙤𝙖𝙘𝙝 𝙩𝙤 𝙧𝙚𝙜𝙧𝙚𝙨𝙨𝙞𝙤𝙣 𝙩𝙚𝙨𝙩𝙞𝙣𝙜?

 During a software testing interview, you may be asked about your approach to regression testing.  
  
Here are some tips for answering this question:  
1. Start by explaining the importance of regression testing in ensuring that the software is working as intended after changes have been made.  
  
2. Describe your process for identifying the areas of the software that are likely to be impacted by the changes. This can include reviewing the change request, talking to the developers, and analyzing the impacted areas of the software.  
  
3. Explain your approach to creating a regression test suite, including identifying the test cases that cover the impacted areas of the software, prioritizing the test cases based on the criticality of the features being tested, and using automation tools to execute the tests.  
  
4. Discuss how you execute the regression test suite after each change to the software, and how you use the results to catch any defects that may have been introduced during the development process.  
  
5. Finally, describe how you continuously review and update the regression test suite to ensure that it is aligned with the current functionality of the software.  
  
 By answering this question in a clear and concise manner, you can demonstrate your understanding of the importance of regression testing and your ability to execute an effective testing strategy.

Finding maximum and minimum numbers in a list using streams and Comparator interface:

max() and min() methods are used for finding the maximum and minimum element in a stream. comparing( Integer::valueOf ) method of the Comparator interface is passed as an argument for the max/ min methods:  
  
Consider a list,  
  
List<Integer> list = new ArrayLis t<Integer> (List.of(11,2,4,13,12,18,1));  
Integer minNumber = [list.stream](http://list.stream/)().min(Comparator.comparing(Integer::valueOf)).get();  
  
List<Integer> list2= new ArrayList<Integer>(List.of(11,2,4,13,12,18,14,22,6));  
Integer maxNumber = [list2.stream](http://list2.stream/)().max(Comparator.comparing(Integer::valueOf)).get();  
  
In the first part of the code, min method is used which has a Comparator type argument. comparing() method in the Comparator interface compares every element with other element in the list and then minimum element is obtained. The get method is used to get the resultant minimum element. Final result would be stored in minNumber variable. Same logic goes with finding maximum number but max method is used.

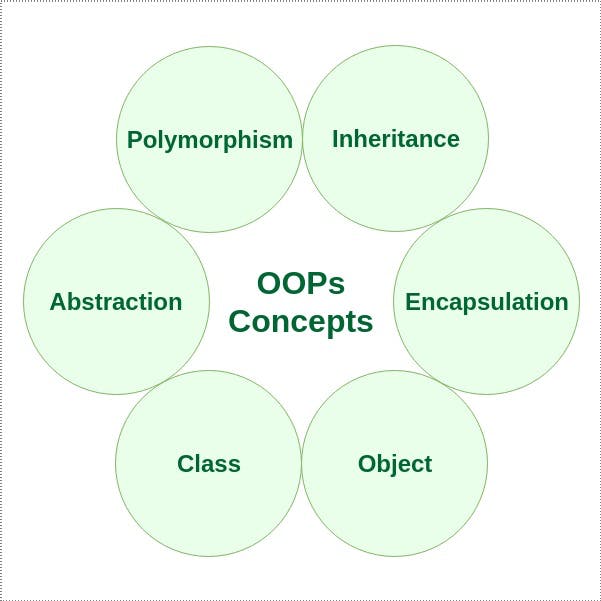
OOPS concept ;) (in Java programming language)

Do you know its possible to hide data in programming? Do you know its possible to construct a skeleton of code without even actual logic and still programmers understand what and when to use it? Do you know just like we inherit traits from our parents/grand-parents we can inherit important code in different ways?  
  
Lets Discuss the fundamentals of OOPS with as described in Infographic attached below. Moving Anti-Clockwise  
  
Abstraction = When we hide internal working mechanism for a process and make visible only certain details which are useful for a potential user, Similarly in programming we prepare skeleton of code (functions, interfaces, abstract classes) and then programmers can use these already constructed code as per there requirement whenever required.  
So what do we understand from this? - It basically hides how the things are implemented to be precise how code is implemented!!  
  
Polymorphism = Process in which we can actually take different forms as per requirement. If we require a function to perform addition of 2 integers, after sometime we need to perform addition of 2 float numbers and so on.  
As per requirement taking different forms is polymorphism.  
It can be of 2 types as described in infographic -  
Compile time polymorphism means = These are processed when code is being compiled.  
Run time polymorphism means = These are processed when code is in execution.  
  
Inheritance = Properties defined in parents are inherited or derived into children. This helps in code-reusability and reduces code complexity in program files.  
Single Inheritance = 1 Parent properties inherited into 1 child  
Multilevel Inheritance= 1 Grand-parent properties inherited into 1 Parent then into 1 Child  
Hierarchical Inheritance = 1 Parent properties inherited into different children  
Hybrid Inheritance = 1 Grand-parent properties inherited into 1 parent then into different children.  
  
Encapsulation = When wrapping data (program/code) into a single code file. Now for this we can keep all the members to be private. By this process we can keep getters and setters to access this code file.  
This process helps us to keep our code unusable outside and gives Read-Only access.  
We can try to implement encapsulation in other ways as well above is a common example.  
Different access modifiers:  
1) Private = Usable within a class  
2) Protected = usable within a class and package inherited  
3) Default = usable within a package defined  
4) Public = No boundaries - accessible everywhere

## OOPs (Object-Oriented Programming Systems)

**Object** means a real-world entity such as a pen, chair, table, computer, watch, etc. **Object-Oriented Programming** is a methodology or paradigm to design a program using classes and objects. It simplifies software development and maintenance by providing some concepts:

* [Object](https://www.javatpoint.com/object-and-class-in-java)
* Class
* [Inheritance](https://www.javatpoint.com/inheritance-in-java)
* [Polymorphism](https://www.javatpoint.com/runtime-polymorphism-in-java)
* [Abstraction](https://www.javatpoint.com/abstract-class-in-java)
* [Encapsulation](https://www.javatpoint.com/encapsulation)



## Object:

Any entity that has a state and behavior is known as an object. For example, a chair, pen, table, keyboard, bike, etc. It can be physical or logical.

An Object can be defined as an instance of a class. An object contains an address and takes up some space in memory. Objects can communicate without knowing the details of each other's data or code. The only necessary thing is the type of message accepted and the type of response returned by the objects.

**Example:** A dog is an object because it has states like color, name, breed, etc. as well as behaviors like wagging its tail, barking, eating, etc.

## Class:

The collection of objects is called class. It is a logical entity.

A class can also be defined as a blueprint from which you can create an individual object. Class doesn't consume any space.

### Inheritance:

When one object acquires all the properties and behaviors of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

### Polymorphism:

If one task is performed in different ways, it is known as polymorphism. For example: to convince the customer differently, to draw something, for example, a shape, triangle, rectangle, etc.

In Java, we use method overloading and method overriding to achieve polymorphism.

Another example can be to speak something; for example, a cat speaks meow, a dog barks woof, etc.

# Abstraction:

Hiding internal details and showing functionality is known as abstraction. For example phone call, we don't know the internal processing.

In Java, we use abstract classes and interface to achieve abstraction.

### Encapsulation:

Binding (or wrapping) code and data together into a single unit are known as encapsulation. For example, a capsule, it is wrapped with different medicines.

A java class is an example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.

### Subscribe to my new

Automation Test Metrics.

 As automation testing becomes increasingly popular, it's essential to have a set of reliable metrics to track and measure its effectiveness. In this post, let's take a closer look at some common types of automation test metrics:  
  
------------------------------------  
 Test Coverage:  
  
This metric measures the percentage of code or features covered by automated tests. It helps to identify areas where testing is lacking, enabling teams to add tests to ensure better coverage.  
  
 FORMULA  
Test Coverage = (Number of Features Covered by Tests ÷ Total Number of Features in the Application) x 100  
---------------------------------  
  
 Test Execution Time:  
  
This metric measures how long it takes to execute automated tests. By tracking execution time, teams can identify tests that take too long to run and optimize them accordingly.  
---------------------------------  
  
 Test Failure Rate:  
  
This metric tracks the number of failed tests and helps teams to identify the areas where defects are occurring most frequently. By analyzing failure patterns, teams can focus on addressing the underlying issues.  
  
 FORMULA:  
Test Failure Rate = (Number of Failed Tests ÷ Total Number of Tests Executed) x 100  
  
For example, let's say you executed a total of 100 automated tests and 10 of those tests failed. To calculate the test failure rate, you would plug in these values into the formula:  
Test Failure Rate = (10 ÷ 100) x 100 = 10%  
-------------------------------------------  
  
 Test Effectiveness:  
  
This metric measures how effective automated tests are in identifying defects. It's important to track this metric to ensure that the tests are providing the expected results.  
--------------------------------------------  
  
 Defect Leakage:  
  
This metric tracks the number of defects that are missed by automated tests and are discovered later in the development process. It helps teams to identify areas where tests need to be improved or enhanced.  
  
 FORMULA:  
Defect Leakage = (Number of Defects Found after Release ÷ Total Number of Defects Found) x 100  
-----------------------------------  
  
These are just a few examples of the types of automation test metrics that can help teams to monitor and measure their testing efforts.  
  
By using these metrics, teams can identify areas of improvement, optimize their testing processes, and ultimately deliver better quality products to their customers.

**What is difference between Defect and Bug**  
  
Ans - Defect - an issue found by a tester during testing  
Bug - When a tester raises and defect and a developer accepts it that yes this issue exists in system and needs to be resolved.  
  
**Who decides Defect/Bug priority and severity?**  
  
Ans - Severity is decided by a Tester  
Priority usually decided by Test Lead/Product-Owner/Scrum-Master  
  
**How to decide Severity for a defect/Bug?**  
  
Ans - Severity can be decided on the basis of How badly it affects our System/Application  
  
**How to decide Priority for a defect/Bug?**  
  
Ans - Priority can be decided on the basis of How badly it hampers/affects our Business Flow  
  
Note - Always provide lot examples while explaining these concepts to your interviewer - it helps to clear out the picture about your understanding.  
  
**How to Raise a Bug/Defect?**  
  
Ans - Steps:-  
1) Replicate the issue  
2) If still exists review the requirements and Test Plan  
3) Record - Test Data, screenshots and videos  
4) Steps to replicate  
5) Explain the issue also keep the requirement attached in your defect  
6) Defect headline - should be clear and precise  
  
How to communicate about a defect which you consider is high severity and priority but team is not accepting it?  
  
Ans - Always be clear with your words and Let them know about the defect and how it affects our business flow.  
Once explained - let them know from your side - you can give a conditional sign-off  
  
**How and When to involve PO/Scrum Master?**  
  
Ans - When the developer is not accepting your defect or simply says that its a feature (which is not possible - if you understand requirements clearly) - Discuss this with your PO and Scrum Master

**Part 1:  
==============**  
  
1) Bug leakage - When end-user finds out the issue and was not catched by QA team.  
Bug release - When a release happens knowingly of a bug presence because the priority and severity of that issue is low.  
  
2) Difference in Test scenario and Use case -  
Test scenario - Created by a tester that validates both front-end and back-end is working correctly and no failures occur for our SUT.  
Use case - When a tester replicates the usage for an application as per end-users usage.  
  
3) Bug Triage - Analyze the bug  
Assign priority for it  
Assign severity for it  
Assign the bug to bug owner  
  
4) Test Metric - It's a measure of total number of tests created and executed it includes -  
Total Test  
Test runs  
Test passed  
Test failed  
Tests deferred  
  
5) Requirement Traceability Matrix - Backward-Forward and Bidirectional traceability  
  
6) Verification (static testing) - means checking code, test-plans, test-strategy-document, BRD, SRS etc.  
Validation (Dynamic testing) - Testing executed code, Executing test-plan - performing regression, smoke testing etc.  
  
7) Build - It's for internal use within a team - more specifically it's given to testing team by Dev team to validate the latest changes and fixes.  
Release - When latest version of software is released to production environment post testing and UAT.  
  
8) Ad-hoc Testing - Performed by experienced resources and used to break the application.  
Performed post formal testing is completed. Not documented.  
Understanding about the application is required.  
Exploratory testing - performed by experienced resources. understanding of the application is done on the go. Less documentation is kept. Part of formal testing process.  
  
9) Defect Cascading - It's a process in which one defect triggers another one and due to this it becomes difficult to reach to the root defect.  
  
10) Defect Clustering - Combining/Clubbing similar defect under same umbrella so that it reduces the overall time-required to resolve them.  
Example - Login form and Signup forms require email fields but it aceepts Injection of HTML code which is an issue - so these are clubbed within a same defect and fixed together.  
  
  
**Part 2:  
==================**  
  
1) What is latent defect?  
Ans A defect for which the conditions to trigger it are hardly met. So for an end user's perspective these defects can't be triggered. Hence can be released. These are usually discussed within team and technical graphs like heat-maps are used to investigate.  
  
2) What is client side validation?  
Ans Validations which are specific to a browser. The UI which a user access - certain validations are kept like Password validations for signup process.  
  
3) What is server side validations?  
Ans These are specific to backend. These validations are kept so that incorrect data cannot be stored. Usually through front-end we cannot access server side validations but there are ways to use them like - API testing, Security testing.  
  
4) What is risk based testing?  
Ans These are conducted depending upon what type of risks that our end user can be impacted from. So during this testing we priortize risks and test them.  
  
5) What is globalization testing?  
Ans Testing which helps an application to be globally used in different languages and regions.  
  
6) What is a critical bug?  
Ans Bug due to which a part of application becomes unusable.  
Depending upon the functionality importance towards end-user we mark critical.  
  
7) What is Error seeding?  
Ans Deliberately adding errors to an application so that we can understand error tolerance of n application.  
Moreover this can be used to evaluate a tester's skill-set to identify these errors.  
  
8) What is showstopper defect?  
Ans A defect due to which user cannot move/use application further. A condition which feels like a crash or is a crash.  
  
9) What is pesticide paradox?  
Ans It is a condition which we are unable to find new issues/defects and hence we need to change our approach towards testing the application. Like moving from functional testing to security testing etc. Changing the test scenarios to test more edge scenarios.  
  
10) What is Authentication?  
Ans A process in which system/application identify a user. If a user has valid account then can be authenticated.  
  
11) What is Authorization?  
Ans A process in which system/application identifies what a user is capable to do? what user can access within an application?. Hence means what privileges are provided to a user.  
  
  
**Part 3:  
==============**  
Q1) What is MR?  
Ans: Modification Request - when client/users ask for modification of an existing feature.  
  
Q2) Why do we need negative testing?  
Ans: If a user performs actions which are not as per a expected user flow - at that time we require Negative testing.  
Negative Testing - tests a feature which its not supposed to do.  
Eg: In a login page it expects - Username and Password for login process - But you as a tester Enter Email ID in username field - Then should it work?  
  
Q3) What is End-to-End testing?  
Ans: System testing can be also called as E2E testing, Now in this we basically test whole application by prioritizing the positive flows of an application and then negative flows.  
  
Q4) How to we execute the following?  
1) Regression Testing  
2) System Testing  
3) Sanity Testing  
4) Smoke Testing  
5) Acceptance testing  
Ans: 4-3-1-2-5  
  
Q5) What is Shift Right Testing?  
Ans: When developers deploy build in QA/Staging/Pre-Prod env and then testers perform there testing.  
  
Q6) What is Shift Left Testing?  
Ans: When testers take part in testing of newly developed features which are yet to be deployed to QA/Staging/Pre-Prod/Prod env.  
  
Q7) What is Recovery Testing?  
Ans: This is executed to tested how quickly system can recover if goes into crash/failure.  
Who will perform this? Usually Devops.  
  
Q8) What is A/B testing?  
Ans: A means version-1 and B means version-2.  
Then these versions are released into different market regions and tested by different end-users.  
The version which has higher ratings are released world-wide.  
Eg: Usually done when features are region specific.  
  
Q9) What is Crowd-Source-Testing?  
Ans: A completely unknown pool of testing resources test your application, you can judge the quality of your product on the basis of number of bugs reported.  
  
Q10) What is Out-Source-Testing?

Ans: A dedicated team is present to handle your testing needs we can say it’s a third party which is unknown to you, test your application or product with a fresh set of mind.  
Can also be called in some cases as Beta testing.

**1. What is the difference between String and StringBuilder in Java?**  
Ans: A String is immutable in Java, while a StringBuilder is mutable in Java.  
An immutable object is an object whose content cannot be changed after it is created.  
  
When we try to concatenate two Java strings, a new String object is created in the string pool.  
This can be demonstrated by comparing HashCode for String object after every concat operation.  
  
**2. Write a program for swapping two numbers without using third variable?**  
Ans:  
public class JavaSwapNumbers {  
public static void main(String args[]){  
int a = 5, b = 6;  
System.out.print("Before swap:\na = " + a + "\nb = " + b);  
a = a + b; //a becomes 11  
b = a - b; //b becomes 5  
a = a - b; //a becomes 6  
System.out.print("\n\nAfter swap:\na = " + a + "\nb = " + b);  
}  
}  
  
**3. Write a program to reverse string without using any function?**  
public class ReverseString {  
  
 public static void main(String[] args) {  
  
  String str = "Hello world";  
  String revString = "";  
  
  for (int i = str.length() - 1; i >= 0; --i) {  
   revString += str.charAt(i);  
  }  
  System.out.println(revString);  
 }  
}  
  
  
**4. Write a program to find the common element from two arrays?**  
import java.util.Scanner;  
  
public class CodeScanner  
{  
   public static void main(String[] args)  
   {  
      int[] arrOne = new int[5];  
      int[] arrTwo = new int[5];  
      int i, j;  
      Scanner s = new Scanner(System.in);  
       
      System.out.print("Enter 5 elements for the first array: ");  
      for(i=0; i<5; i++)  
         arrOne[i] = s.nextInt();  
      System.out.print("\nEnter 5 elements for the second array: ");  
      for(i=0; i<5; i++)  
         arrTwo[i] = s.nextInt();  
       
      System.out.println("\nCommon elements are:");  
      for(i=0; i<5; i++)  
      {  
         for(j=0; j<5; j++)  
         {  
            if(arrOne[i]==arrTwo[j])  
               System.out.print(arrOne[i]+ " ");  
         }  
      }  
   }  
}  
  
**5. What is the difference between = = and equals methods?**  
== is used to compare the reference variable of two String object & equals is used to compare the content of two Strings object .Both returns the boolean value.  
  
Example...  
String s1="Java";  
String s2="Java";  
System.out.print(s1.equals(s2));//true because content is same.  
System.out.print(s1.==s2);//true because both reference variable are refering to the same String.  
  
**6. What is the difference between method overloading and method overriding?**  
Method Overloading      
1)    Method overloading is used to increase the readability of the program.      
2)    Method overloading is performed within class.      
3)    In case of method overloading, parameter must be different.      
4)    Method overloading is the example of compile time polymorphism.      
5)    In java, method overloading can't be performed by changing return type of the method only. Return type can be same or different in method overloading. But you must have to change the parameter.      
  
Method Overriding  
1)    Method overriding is used to provide the specific implementation of the method that is already provided by its super class.  
2)    Method overriding occurs in two classes that have IS-A (inheritance) relationship.  
3)    In case of method overriding, parameter must be same.  
4)    Method overriding is the example of run time polymorphism.  
5)    Return type must be same or covariant in method overriding.  
  
**7. Give an example of method overloading in selenium?**  
This is a simple overriding example of the "findElement" method.  
  
    public static WebElement findElement(By Locator){  
  
    WebElement anElement = fluentWait.until(new Function<WebDriver, WebElement>() {  
        @Override  
        public WebElement apply(WebDriver webDriver) {  
            webDriver=sampleDriver;  
            return sampleDriver.findElement(Locator);  
        }  
    });  
  
    return anElement;  
}  
  
**8.SQL query to fetch 2nd max salary from the table?**  
select  
  (SELECT MAX(Salary) FROM Employee) maxsalary,  
  (SELECT MAX(Salary) FROM Employee  
  WHERE Salary NOT IN (SELECT MAX(Salary) FROM Employee )) as [2nd\_max\_salary]  
  
**9.What is the difference between Array and ArrayList?**  
Array: A straightforward data structure with a continuous memory location, an array stores its contents with the same name but distinct index numbers for each element of the array it contains.  
It is imperative that all of the data stored in an array be of the same type. After an array has been declared, its size cannot be changed.  
Array List: The Java collection framework contains a data structure known as an ArrayList, which is dynamic in nature.  
Additionally, it has components that are of the same type. In this case, it is not necessary for us to specify the length of the list.

Array :  
  
1. Array is a non primitive data type , used for storing multiple values .  
2. We can store same type of data in an Array or an Array contains homogeneous data  
3. Array is static , as we need to specify the length of the array .  
4. We can get the size of the array using arrayvar.length  
Example : arrayName.length  
5. We can traverse the array using for loop / for-each loop  
6. Array is fixed in length.  
7. To perform some specific operations on array like sorting and searching we can use Arrays class which is a class available under util package.  
java.util.Arrays;  
   
ArrayList :  
  
1. ArrayList is a class under java collection framework.  
2. ArrayList can contain heterogeneous elements if we will not specify the type .  
Example:  
ArrayList al = new ArrayList( ); // can hold int , float , string type ect  
ArrayList <Integer> al = new <Integer> ArrayList( ); // can hold only integers  
3. ArrayList is dynamic , without bothering about the size we can add n number of elements i.e we do not have to specify the size.  
4. We can get the size of the ArrayList using size( ) method.  
Example : ArrayListObject.size ( )  
5.We can traverse the ArrayList using iterator with while loop. //Iterator is an interface  
6. ArrayList is variable length.  
7. To perform specific operations on ArrayList like getting max , min , sort  
etc

**10. How to input text in the text box without sendkeys?**  
Selenium Automation Testing Testing Tools We can input text in the text box without the method sendKeys with thehelp of the JavaScript Executor.  
Selenium executes JavaScript commands with the help of the executeScript method. The JavaScript command to be run is passed as parameter to the method.  
  
**11.How to handle scroll bar using selenium?**  
Using Javascript executor  
Basically, there are of 2 types:  
  
Horizontal Scroll bar  
Vertical Scroll bar  
#1) Horizontal Scroll bar  
  
A horizontal scroll bar lets the user scroll towards the left or right to view all the content on the window.  
JavascriptExecutor js = (JavascriptExecutor)dr;  
          js.executeScript("window.scrollBy(0,70)");  
  
A vertical scroll bar lets the user scroll up-down or vice versa to view the complete content on the window.  
  
**12. What is the return type of findElement?**  
Ans: driver.findelement returns WebElement  
  
**13. WebElement is an interface or class.**  
Ans: WebElement is an Interface.WebElement represents an HTML element. Generally, all interesting operations to do with interacting with a page will be performed through this interface.  
All the frequently used methods like clear(), click(), findElement, findElements(), sendKeys, getText() and many others are associated with this interface.  
  
**14.How to perform double click on any component?**  
Ans: e can perform double click on elements in Selenium with the help of Actions class.  
In order to perform the double click action we will use moveToElement () method, then use doubleClick () method.  
Finally use build ().perform () to execute all the steps.  
  
**15.What is the difference between Interface and Abstract?**  
Ans:  
Abstract class can have abstract and non-abstract methods.Interface can have only abstract methods. Since Java 8, it can have default and static methods also  
Abstract class doesn't support multiple inheritance.Interface supports multiple inheritance.  
An abstract class can be extended using keyword "extends". An interface can be implemented using keyword "implements".  
  
**16.Can Interface be extended to another interface?**  
Ans: Yes, you can do it. An interface can extend multiple interfaces, as shown here:  
  
interface Maininterface extends inter1, inter2, inter3 {    
  // methods  
}  
  
**17. What do you mean by XPath expressions?**  
XPath defines a pattern or path expression to select nodes or node sets in an XML document. These patterns are used by XSLT to perform transformations. XPath specifies seven types of nodes that can be output of the execution of the XPath expression.  
  
Root  
Element  
Text  
Attribute  
Comment  
Processing Instruction  
Namespace  
  
**18.What is XPath syntax?**  
The XPath syntax specifies the different nodes, path expressions, path notation, predicates and URLs which are used to define the XML document.  
  
**19.What do you mean by XPath Axes?**  
XPath axes are used to identify elements by their relationship like parent, child, sibling, etc. in the same manner like path defines the location of the node.  
  
**20.What do you understand by XPath number functions?**  
In XPath, number functions are used to fetch the different type of values from the expressions. For example, ceiling value, floor value etc.  
  
There are four types of number functions in XPath:  
  
ceiling()  
floor()  
round()  
sum()

**21. How to mouse hover over a web element?**  
Actions class utility is used to hover over a web element in Selenium WebDriver  
Instantiate Actions class.  
    Actions action = new Actions(driver);  
In this scenario, we hover over search box of a website  
  actions.moveToElement(driver.findElement(By.id("id of the searchbox"))).perform();  
  
**22. How to take screenshots in WebDriver?**  
TakeScreenshot interface can be used to take screenshots in WebDriver.  
getScreenshotAs() method can be used to save the screenshot  
File scrFile = ((TakeScreenshot)driver).getScreenshotAs(outputType.FILE);  
  
**23.Is there a way to type in a textbox without using sendKeys()?**  
Yes! Text can be entered into a textbox using JavaScriptExecutor  
JavascriptExecutor jse = (JavascriptExecutor) driver;  
jse.executeScript("document.getElementById(‘email').value=“[abc.efg@xyz.com](mailto:abc.efg@xyz.com)”);  
  
**24.What do you mean by the assertion in Selenium?**  
An assertion is a method of testing whether a particular condition is true or false. In Selenium, assertions are used to verify the state of elements on a page or the results of an action.

**25. How is the relationship among nodes defined?**  
The nodes that are available in XPath are given below. And these are related to each other in a structure like tree. The nodes are-  
Parents  
Children  
Siblings  
Ancestors  
Descendants  
  
**26. How is dynamic text handled in XPath?**  
The text function finds the elements with the help of the text present in them.  
<button type = “button”>Blueberry</button>     (Blueberry is text here.)  
  
XPath with text : //button[ text() = ‘Blueberry’ ]  
But contains function has to be used along with text function when the text is dynamic. And some part of the string needs to be static for this to work.  
<button type = “button”>\*\*\*berry</button>  
  
XPath with text : //button[ contains( text(), ‘berry’) ]  
  
**27. Can the XPath be used inside a frame?**  
The frame is nothing else but another webpage which is shown as a different web page. Therefore, we can use the XPath just like a normal web page once we go into the frame.  
  
**28. Write the XPath for a button.**  
The XPath for a button is written as given below. The button can be formed in two ways- by using input tag or button tag.  
// form button using the input tag  
<input type = “button’ value = ‘ABCD’>  
XPath : // input [ @type = ‘button’ ]      
  
// form button using the button tag  
<button>ABCD</button>  
XPath : // button

How did you design and implement the Test automation framework in your project?  
  
**Answer:** (Source Linkedin)  
- Define the testing requirements: Understand the testing requirements and expected outcomes from the testing.  
  
- Create a test strategy that outlines the testing approach, test levels, types of tests, and acceptance criteria.  
  
- Identify critical functionality and prioritize for automation: Understand the critical functionality that the services provide and prioritize those for automation.  
  
- Evaluate testability and assess the value of automation: Assess how easy it is to automate testing for each service, analyze the potential benefits of automating testing for each service. Take into account maintainability and limitations of automation.  
  
- Design the test architecture: Plan out the overall structure of the framework, such as the modular design, the flow of control, the design pattern, and the technology stack to be used.  
  
- Choose a test automation tool: Select a suitable tool for automating the testing.  
  
- Develop reusable components: Develop components such as helper functions, utility classes, and common assertion methods that can be used across multiple test scripts.  
  
- Design test data management: Decide on how test data will be managed, such as through spreadsheets, databases, or external files, and implement necessary interfaces or classes.  
  
- Implement test runners: Develop a way to execute test scripts in a batch, either sequentially or parallelly, as per the requirement.  
  
- Handle error and exception: Design a way to catch and handle the errors, exceptions, and edge cases that may occur during the testing.  
  
- Implement logging: Include a logging mechanism in the framework to capture information such as test execution progress, test results, and any error messages that occur during testing.  
  
- Implement reporting: Include a reporting mechanism in the framework to generate test execution reports that provide an overview of the test results, including information such as passed/failed test cases, test execution time, and any errors that occurred.  
  
- Integrate into the CI/CD pipeline: Configure the test automation framework to be executed as part of the continuous integration and continuous deployment processes.  
  
- Maintenance: Keep the framework, its logging and reporting mechanisms, and its integration with the CI/CD pipeline up to date with any changes to the services or requirements.

What is Accessibility Testing ?

Ans:  
It is a subset of Usability Testing. People with disabilities use assistive technology, which helps them in operating a software product.  
  
Examples of such software are:  
  
Speech recognition software – Converts the spoken word to text, which serves as input to the computer.  
Screen reader software – Used to read out the text that is displayed on the screen.  
Screen Magnification Software– Used to enlarge the monitor and make reading easy for vision-impaired users.  
Special keyboard made for users for easy typing who have motor control difficulties  
  
Note: In Accessibility Testing for Disabled audience one very major point to consider is Accessing the Website without the use of the mouse. A person should be able to complete access the website the links, buttons, radio buttons, checkboxes, pop-ups, dropdown, all the controls should be completely accessible and operable through the keyboard.  
For any Video and Audio, there should be Alt text for deaf people.  
Some Test Cases:  
1.     application provides keyboard equivalents for all mouse operations and windows?  
2.     tabs are ordered logically to ensure smooth navigation?  
3.     shortcut keys are provided for menus?  
4.     response time of each screen or page is clearly mentioned so that End Users know how long to wait?  
5.     all labels are written correctly in the application?  
6.     the color of the application is flexible for all users?  
7.     images or icons are used appropriately so it’s easily understood by the end users?  
8.     an application has audio alerts?  
9.     user is able to adjust audio or video controls?  
10. user can override default fonts for printing and text displays?  
11.  user can adjust or disable flashing, rotating, or moving displays?  
12.  color-coding is never used as the only means of conveying information or indicating an action.  
13. highlighting is viewable with inverted colors? Testing of color in the application by changing the contrast ratio  
14. audio and video-related content is properly heard by disabled people? Test all multimedia pages with no speakers on websites  
15. training is provided for users with disabilities that will enable them to become familiar with the software or application?  
  
Axe dev tools browser extension is one of the tools used for Accessibility testing where it analyses the web page and gives results.

Here are some of the most useful shortcuts for Eclipse and Visual Studio Code:

Eclipse:  
Ctrl + Shift + R: Open resource by name  
Ctrl + Shift + T: Open type by name  
Ctrl + Shift + L: Show all shortcuts  
Ctrl + Shift + O: Organize imports  
Ctrl + /: Comment/uncomment line  
Ctrl + Shift + F: Format code  
Ctrl + 3: Quick access to commands and resources  
Ctrl + 1: Quick fixes and quick assist  
Ctrl + Q: Go to last edit location  
Ctrl + K: Delete to end of line  
Ctrl + Shift + K: Delete to start of line  
Ctrl + L: Go to line  
Ctrl + M: Maximize/minimize active editor  
Ctrl + Shift + W: Close all editors  
Ctrl + Shift + A: Show all commands  
Ctrl + O: Open outline view  
Ctrl + Shift + R: Open resource  
Alt + Shift + W: Show/hide views  
Alt + Arrow Up/Down: Move line up/down  
Ctrl + D: Delete line  
Ctrl + T: Open type hierarchy  
  
Visual Studio Code:  
Ctrl + Shift + P: Command palette  
Ctrl + P: Go to file  
Ctrl + Shift + O: Go to symbol  
Ctrl + Shift + L: Select all occurrences of current selection  
Ctrl + K, Ctrl + C: Comment line  
Ctrl + K, Ctrl + U: Uncomment line  
Ctrl + Shift + D: Duplicate line  
Ctrl + Shift + Alt + Arrow Up/Down: Move line up/down  
Ctrl + Shift + K: Delete line  
Ctrl + D: Add selection to next occurance  
Ctrl + U: Remove selection  
Ctrl + T: Go to symbol in the file  
Ctrl + G: Go to line  
Ctrl + W: Close active editor  
Ctrl + Shift + T: Reopen closed editor  
Ctrl + P: Go to file  
F12: Go to definition  
Shift + F12: Peek definition  
Alt + F12: Show definition in new window  
Ctrl + Shift + F: Format document  
Ctrl + Shift + Alt + F: Format selection  
Ctrl + . : Quick fix

What is Devops ? Pipeline ? Cotninuous testing ? etc

What is Devops?  
What is a pipeline?  
Why do we need continuous testing?  
What all is required to construct a successful pipeline ?  
  
Let's discuss ;)  
  
Devops - it's a process which unites different teams to collaborate.  
Initially all team like Dev, QA, IT operations etc used to complete there assigned tasks and then code used to deploy in production.  
Now New features, functionality are developed, tested and then deployed on the go.  
This can be implemented to small/large scale projects no matter what complexity.  
  
Pipeline - Consists a set of steps which are supposed to be followed to deploy code into production - we call this combination of steps a pipeline.  
Now what are commonly used steps?  
- Coding a feature  
- Deploy this feature into QA env  
- Test this feature  
- Raise Bugs  
- Fix these Bugs through patches  
- Final Regressions  
- Deployment  
  
Now in this process mostly tests are automated and the pipeline is constructed through which even  
Non-Technical team members can deploy code to Production env (post signoff)  
  
Continuous Testing - When a developer developes a feature,  
so they execute a set of unit test scenarios (which can be automated or manual) - Once result is successful then this feature is deployed into QA env.  
  
Post this automated testing (QA) is started for build acceptance and then testing the feature in depth  
(this is where we implement in-sprint automation)  
  
If all results are successful then code is deployed into production.  
  
This process of testing is called as continuous testing.  
  
What is required to create a Devops pipeline?  
  
- Continuous integration tool  
- Continuous deployement tool  
- Version control (GitHub)  
- Continuous Testing  
- Test Automation  
- Continuous monitoring  
- Deploy code to production

What is Schema Validation in API testing ? How to do it in Postman

1. What is Schema Validation  
   2. Why to do Schema Validation  
   3. How to do Schema Validation in Postman.  
     
   So let's get started…  
     
   Most of us use Postman for Functional Black Box testing and the focus is always on Status Code, Response Body and Response time.  
     
   Which is good, for sure.  
     
   But you can improve the Quality of Your Tests by including Schema Validation for your Response body  
     
   1. What is Schema Validation, in Simple Words:  
     
   -If you want to make sure your API responses are returning exactly what you expect?  
   -Then you should, you go to solution is Schema Validation.  
   Schema validation in API testing is the process of ensuring that the structure and data types of the request and response payloads or BODY conform to a pre-defined schema.  
     
     
   2.Benefit of Schema Validation?  
     
   -Schema validation helps ensure that the API is returning the correct data and that the client is sending the correct data to the API.  
   -It can be used to catch errors early in the development process and improve the overall quality and reliability of the API.  
     
   3. How to do Schema Validation in Postman  
     
   - Pre Requisite: You should be aware of postman variables for this and understand Postman Object.  
   - Now, You can specify a JSON Schema to validate the response against in the TEST TAB. Simply provide the schema in a string format and use the pm.response.to.have. jsonSchema() command in your test script.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Snippet\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*:  
pm.test("Schema validation", function () {  
pm.response.to.have.jsonSchema(JSONSCHEMA)  
})  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
You can create JSON SCHEMA using the website:  
URL <https://lnkd.in/dCjhR2CJ>  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
Example For Converting JSON Object to JSON SCHEMA:  
Suppose you a simple JSON Object Like:  
{  
"message" : "Success"  
}  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
The Corresponding JSON SCHEMA will look Like  
{  
"type": "object",  
"properties": {  
"message": {  
"type": "string"  
}  
},  
"required": [  
"message"  
]  
}  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
The Postman Test will Look Like:  
  
pm.test("Schema validation", function () {  
pm.response.to.have.jsonSchema(  
  
{  
"type": "object",  
"properties": {  
"message": {  
"type": "string"  
}  
},  
"required": [  
"message"  
]  
}  
)  
})  
  
Just pass the JSON SCHEMA to the pm.response.to.have.jsonSchema()  
  
Explanation:  
Here we are checking:  
1. the structure of the response payload contains a key name message.  
2. Value of the message should be String

What are the top few assertions you want to put on your API tests/backend api level automated checks?

Ans. There are approx. 6 assertions that we can put on our API checks -:  
  
1. HTTP response code validation (200/400/500/etc.) - It depends upon the scenario and expected outcome.  
  
2. Response time validation (200 ms /500 ms/etc.) - It depends on your performance thresholds or product non-functional KPI metrics. It can easily be calculated and validated using the helper methods given by #restassured , #postman and other tools.  
  
3. Response header validation - It depends on expected response header key-value pairs.  
  
4. Response body validation - It includes the expected JSON/XML/other response validation based on the input parameters.  
  
5. DB/storage/cache Validation - It includes verifying the response with the source of truth i.e the database or the storage layer.  
  
6. Response schema validation - It includes validating the schema of the API response thus helping to ensure the contracts do not break with subsequent code changes. These contracts can be between two microservices or between the frontend and backend layers. A good placeholder for #contracttests to ensure no change in the producer breaks the consumer logic.

can we store object ?

Storing Objects in an array

Yes, since objects are also considered as datatypes (reference) in Java, you can create an array of the type of a particular class and, populate it with instances of that class.

Example

Following Java example have a class named Std and later in the program we are creating an array of type Std, populating it, and invoking a method on all the elements of the array.

class Std {

   private static int year = 2018;

   private String name;

   private int age;

   public Std(String name, int age){

      this.name = name;

      this.age = age;

   }

   public void setName(String name) {

      this.name = name;

   }

   public void setAge(int age) {

      this.age = age;

   }

   public void display(){

      System.out.println("Name: "+this.name);

      System.out.println("Age: "+this.age);

      System.out.println("Year: "+Std.year);

   }

}

public class Sample {

   public static void main(String args[]) throws Exception {

      //Creating an array to store objects of type Std

      Std st[] = new Std[4];

      //Populating the array

      st[0] = new Std("Bala", 18);

      st[1] = new Std("Rama", 17);

      st[2] = new Std("Raju", 15);

      st[3] = new Std("Raghav", 20);

      //Invoking display method on each object in the array

      for(int i = 0; i<st.length; i++) {

         st[i].display();

         System.out.println(" ");

      }

   }

}

Output

Name: Bala

Age: 18

Year: 2018

Name: Rama

Age: 17

Year: 2018

Name: Raju

Age: 15

Year: 2018

Name: Raghav

Age: 20

Year: 2018

Basic XPath revision for Inteviews :

1. //𝐢𝐧𝐩𝐮𝐭[@𝐢𝐝='𝐮𝐬𝐞𝐫𝐧𝐚𝐦𝐞'] - Selects an input element with the attribute "id" equal to "username".  
     
   2. //𝐢𝐧𝐩𝐮𝐭[@𝐭𝐲𝐩𝐞='𝐬𝐮𝐛𝐦𝐢𝐭' 𝐚𝐧𝐝 @𝐯𝐚𝐥𝐮𝐞='𝐒𝐢𝐠𝐧 𝐈𝐧'] - Selects an input element with the attribute "type" equal to "submit" and the attribute "value" equal to "Sign In".  
     
   3. //𝐢𝐧𝐩𝐮𝐭[@𝐭𝐲𝐩𝐞='𝐬𝐮𝐛𝐦𝐢𝐭' 𝐨𝐫 @𝐯𝐚𝐥𝐮𝐞='𝐒𝐢𝐠𝐧 𝐈𝐧'] - Selects an input element with either the attribute "type" equal to "submit" or the attribute "value" equal to "Sign In".  
     
   4. //𝐢𝐧𝐩𝐮𝐭[𝐧𝐨𝐭(@𝐯𝐚𝐥𝐮𝐞='𝐒𝐢𝐠𝐧 𝐈𝐧')] - Selects all input elements that do not have the attribute "value" equal to "Sign In".  
     
   5. //𝐢𝐧𝐩𝐮𝐭[𝐬𝐭𝐚𝐫𝐭𝐬-𝐰𝐢𝐭𝐡(@𝐢𝐝, '𝐮𝐬𝐞𝐫')] - Selects all input elements with an attribute "id" that starts with "user".  
     
   6. //𝐢𝐧𝐩𝐮𝐭[𝐜𝐨𝐧𝐭𝐚𝐢𝐧𝐬(@𝐢𝐝, '𝐧𝐚𝐦𝐞')] - Selects all input elements with an attribute "id" that contains the string "name".  
     
   7. //𝐢𝐧𝐩𝐮𝐭[𝐞𝐧𝐝𝐬-𝐰𝐢𝐭𝐡(@𝐢𝐝, '𝐧𝐚𝐦𝐞')] - Selects all input elements with an attribute "id" that ends with "name".  
     
   8. //𝐢𝐧𝐩𝐮𝐭[𝐭𝐞𝐱𝐭()='𝐒𝐢𝐠𝐧 𝐈𝐧'] - Selects all input elements with text content equal to "Sign In".  
     
   9. //𝐢𝐧𝐩𝐮𝐭[𝐜𝐨𝐧𝐭𝐚𝐢𝐧𝐬(𝐭𝐞𝐱𝐭(), '𝐈𝐧')] - Selects all input elements with text content that contains the string "In".  
     
   10. //𝐢𝐧𝐩𝐮𝐭[𝐬𝐭𝐚𝐫𝐭𝐬-𝐰𝐢𝐭𝐡(𝐭𝐞𝐱𝐭(), '𝐒𝐢𝐠𝐧')] - Selects all input elements with text content that starts with "Sign".  
     
   11. //𝐢𝐧𝐩𝐮𝐭[𝐞𝐧𝐝𝐬-𝐰𝐢𝐭𝐡(𝐭𝐞𝐱𝐭(), '𝐈𝐧')] - Selects all input elements with text content that ends with "In".

Useful Methods in Arrays Class :

1. Arrays.sort(ArrayName) :  
void method which we can use to sort any array in ascending order.  
  
Example :  
int arr[] = {1,8,3,4,5};  
      Arrays.sort(arr);  
      for(int i:arr) {  
         System.out.print(i + " ");  
      }  
// output : - 1 3 4 5 8  
  
  
2. Arrays.BinarySearch(ArrayName, search element) :  
To search any element in array we can use this method . It returns int as  
the index of the search element.  
  
Example:  
int arr[] = {1,8,3,4,5};  
      int res=Arrays.binarySearch(arr, 5);  
      System.out.println(res);  
// output : 4 (index of 5 in arr)  
  
3. Arrays.equals(ArrayVar1,ArrayVar2) :  
Returns boolean , we can use this method to compare the array  
elements are similar for two different arrays or not.  
  
Example :  
int arr[] = {1,8,3,4,5};  
     int brr[] = {1,8,3,4,5};   
  
       if(Arrays.equals(arr, brr))  
          {  
          System.out.println("Equal");  
          }  
       else  
          {  
          System.out.println("Not equal");  
          }  
  
// Output : Equal

How to add multiple elements in one go to a List , ArrayList , LinkedList ?

Tips : Use List.of( )  
  
\* We can use List.of( ) instead of add( ).  
\* add( ) allows us to store only one value at a time where as using List.of( ) we  
can store multiple values in one go.  
\* List.of( ) is not applicable for Set.  
  
Example :  
// Adding multiple values to List  
List <Integer> rollno = List.of(10,20,30,40,50);  
  
//Adding multiple values to LinkedList  
LinkedList<String> NameList =     
new LinkedList<String>(List.of("gayatri", "sunidhi","mishra"));   
  
//Adding multiple values to ArrayList  
ArrayList<Double> sal =    
newArrayList<Double>(List.of(56000.90,98000.89));

What is Array of objects ?

Ans:  
Array of objects stores different values String ,integer , double etc as objects.  
Sample code :  
  
Object arr[] = new Object[size]  
We can initialize the object array too.  
Ex : Object obj[] = {1,"Hello",3,4,5,'P',7,90.78,9,10};  
Object obj1[] = {10,28,89,67,54};  
Object obj2[] = {"java","python"}  
Object obj3[] = {10.9, 45.5,87.3}

## Quick Tip on Arrays in Java - nCopies method

Array List declaration with Collections. Copies method:

This creates an ArrayList with 5 copies of the String "Apple" using the Collections.nCopies method.

ArrayList<String> myList = new ArrayList<String>(Collections.nCopies(5, "Apple"));

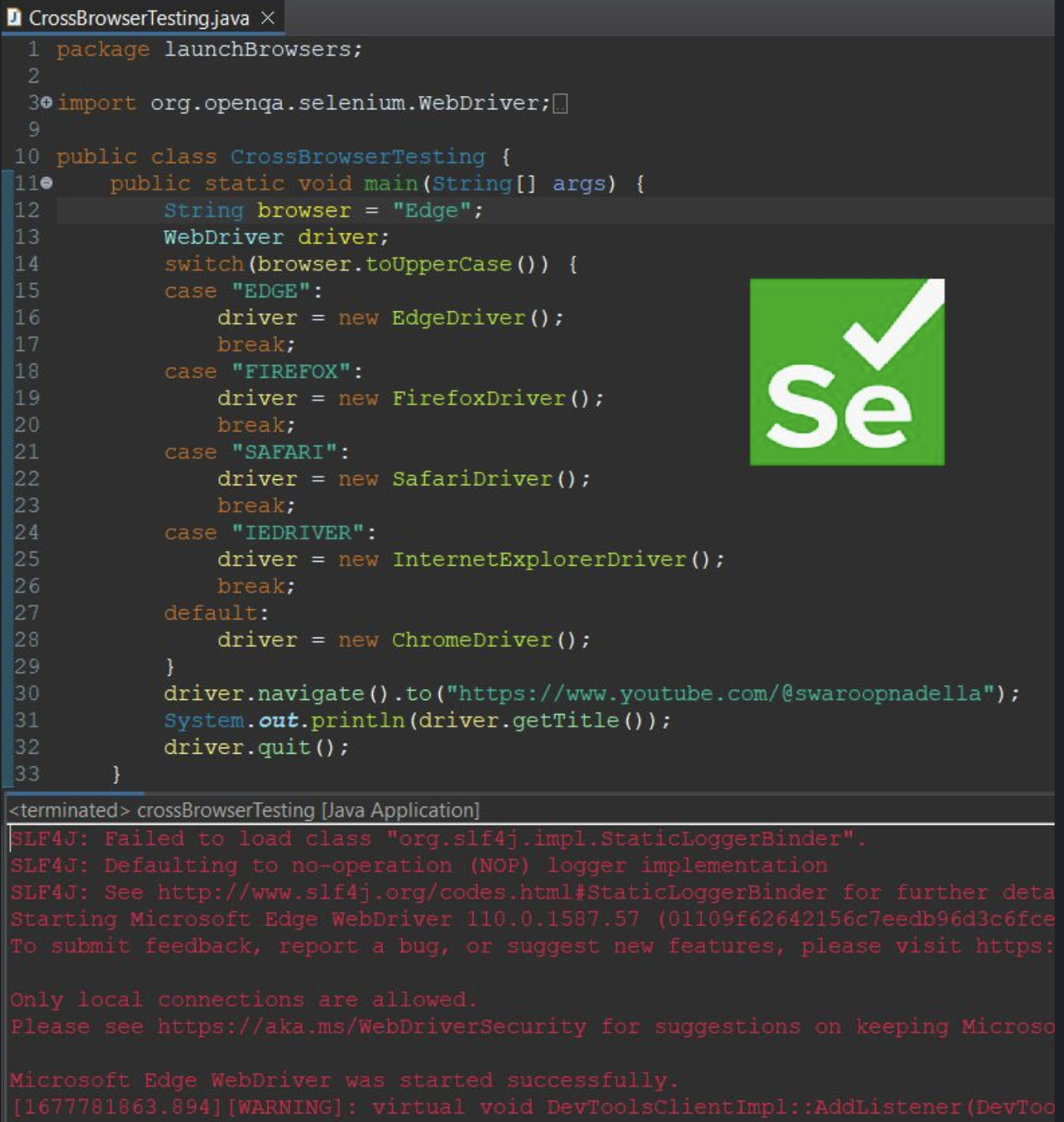
System.out.println(myList.size()) // 5

System.out.println(myList) // [Apple,Apple,Apple,Apple,Apple]

Exception handling using try , catch and finally :

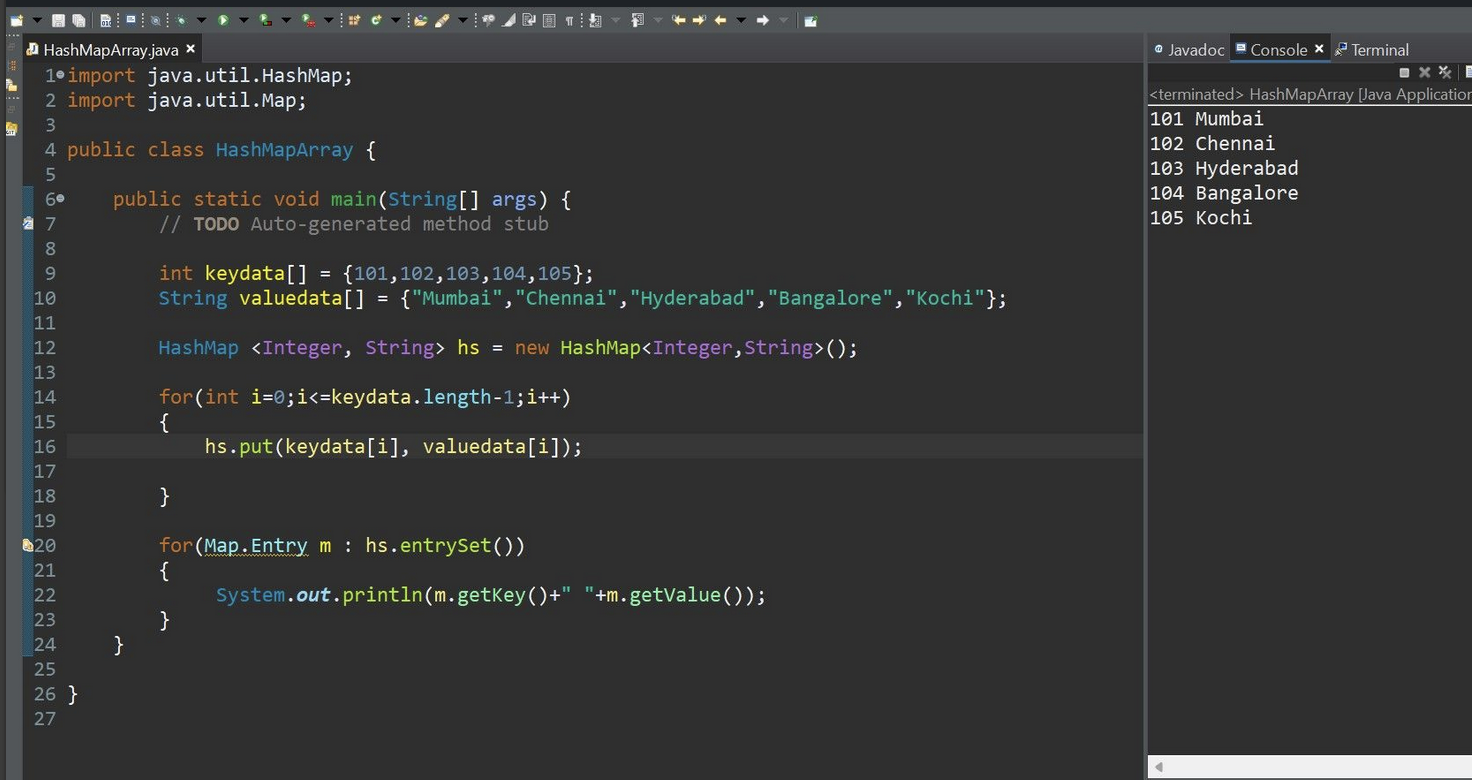
try , catch and finally are the keywords which we can use to handle different exceptions in java.  
  
In try block we can keep the code that might throw the exception .  
Example:  
try {  
          int number=800;  
         System.out.println(number/0); // Divide the number by 0  
        }  
  
We associate exception handler with try block by providing catch block.  
Example:  
catch(ArithmeticException e)  
          {  
          System.out.println("This is : "+ e);  
          }  
  
The finally block always executes when the try block exits. The finally block is executed even if an unexpected exception occurs.  
Example of try with finally block :  
 try {  
          int number=800;  
         System.out.println(number/0); // Divide the number by 0  
           }  
 finally{  
          System.out.println("This is finally block");  
           }  
  
Note :  
1. Each try block can have one catch block .  
2. We can have multiple try and catch blocks to handle more than one exception.  
3. We can have a single try with multiple catch blocks as well.  
4. We can have try with finally block .  
5. Only one finally block is allowed to execute.

**Q1) What is Client-Side validations?**  
Ans: Client-side validation is the one which is basically done at the browser level where the user’s input is validated at the browser itself with no involvement of the server.  
  
The client-side validation is usually done by script language such as JavaScript etc.  
  
Example - Form Validations.  
  
**Q2) What do you understand by Server-side validation?**  
Ans: Server-side validation occurs where the validation and processing of user requests require the response from the server. To understand it more clearly, the user’s input is being sent to the server and validation is done using server-side scripting languages.  
  
After the validation process, feedback is sent back to the client in the form of a dynamically generated web page.  
  
When compared to the Client-Side validation process, the Server-side validation process is more secure because here application is protected against malicious attacks and users can easily bypass client-side scripting language.  
  
**Q3) What is a Proxy server?**  
Ans: The proxy server is a server that acts as an intermediary or is the one that lies between the client and the main server.  
  
The communication between the main server and client-server is done through a proxy server as the client request of any connection, file, resources from the main server is sent through a proxy server and again the response from the main server or local cached memory to client-server is done through the proxy server.  
  
**Q4) What if an organization's growth is so rapid that standard testing procedures are no longer feasible? What should you do in such a situation?**  
  
Ans: This is a very prevalent issue in the software industry, especially with the new technologies that are being used in product development. In this case, there is no simple answer; however, you could:  
- Hire people who are good at what they do.  
- Quality issues should be ‘fiercely prioritized' by management, with a constant focus on the client.  
- Everyone in the company should understand what the term "quality" implies to the end-user.  
  
**Q5) What does the term ‘quality' mean when testing?**  
  
Ans: Quality in testing refers to the degree to which a product meets its intended requirements, as well as the degree to which it satisfies customer needs and expectations. It includes both the functional and non-functional aspects of the product. Quality assurance is ensuring that the product meets its requirements, while quality control focuses on testing to ensure that the product meets its needs.

**Why we use WebDriver driver = new ChromeDriver() ?**  
Ans: [Cross Browser Testing]  
  
If we create driver object like ChromeDriver driver = new ChromeDriver();  
> We are creating instance of ChromeDriver class and it will only work with Chrome browser  
> If we want to test Firefox browser, need to create another new driver object. FirefoxDriver driver\_1= new FirefoxDriver();  
> Cross Browser testing not possible in Run time if we create multiple driver objects this way.  
  
When we automate any Web Applications, we must cover Automation Framework for all the supported browsers. It is called as Cross Browser testing.  
  
To perform Cross Browser testing in Run time, we can create reference variable of parent WebDriver interface and cast it to Required Browser class as below  
WebDriver driver = new ChromeDriver();  
> If we want to test for Firefox browser then we can use the same driver object and cast it to FirefoxDriver class  
driver = new FirefoxDriver();  
  
WebDriver driver instance has access to all the methods of WebDriver. This happens because every Browser Driver class extends RemoteWebDriver class and RemoteWebDriver implements WebDriver interface.  
  
I have used the latest Selenium 4.8.1 version for this code (Selenium Manager changes available from 4.6.0 version onwards).  
> System.setproperty() is no longer needed for setting up browser driver exe file  
> WebDriverManager is no longer required to setup driver binaries  
  


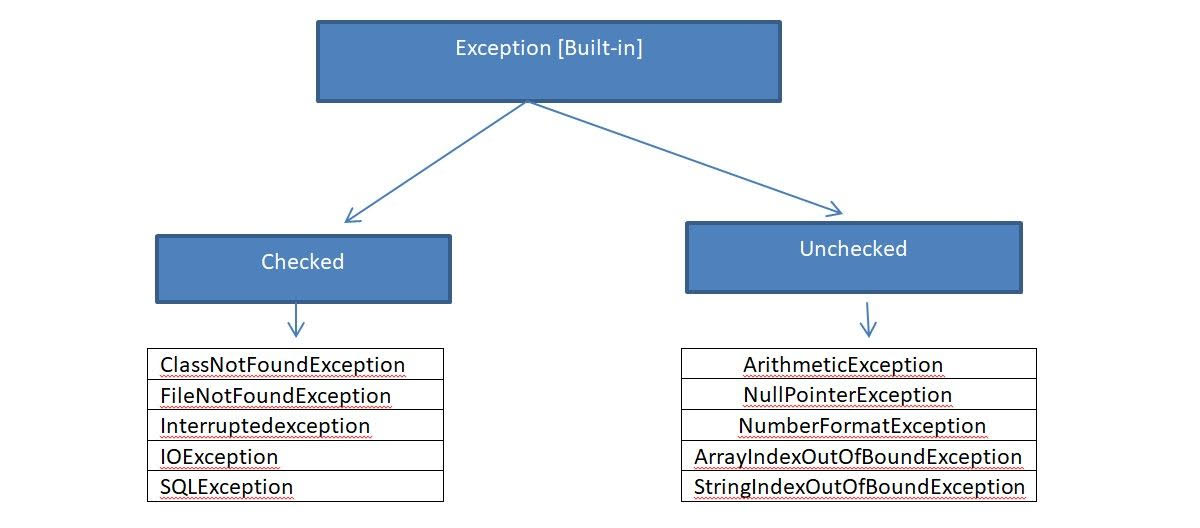
How to load the keys and values from array to hashmap ?

**Steps :**  
Step-1 : Initialize all the keys in a single dim array.  
Step-2 : Initialize all the values in a single dim array.  
Step-3 : In side the traversal loop use  
hashmapObject.put(keys1[i] , values1[i])



What are the different types of exceptions [Built-in] in Java?

Exception can be both built-in and user defined.  
  
Different types of built-in exceptions in java :  
  
Major two type of Exception(Built-in):  
  
\*\*Unchecked Exception :  
Unchecked exceptions are run time exceptions .  
  
Example of unchecked exception:  
1. ArithmeticException  
2. NullPointerException  
3. NumberFormatException  
4. ArrayIndexOutOfBoundException  
5. StringIndexOutOfBound  
6. DateFormatException  
  
\*\*Checked Exception :  
These are the compile time exception .  
  
Examples of checked exception :  
1. ClassNotFoundException  
2. FileNotFoundException  
3. InterruptedException  
4. IOException  
5. SQLException



Here are some important Selenium exceptions :

>The WebDriverException serves as a base class for all exceptions thrown by the WebDriver API.  
  
->When attempting to interact with an element that is no longer attached to the DOM, the StaleElementReferenceException is thrown. This can happen if the element has been deleted or if the page has been refreshed since the element was located. "Stale" typically indicates something is no longer fresh or current.  
  
->The TimeoutException is thrown when a command times out. This can occur if the page takes too long to load or if an element is not found within the specified time period.  
  
->When it is unable to establish a connection to a web browser, the UnreachableBrowserException is thrown.  
  
->The NoSuchElementException is thrown when it is unable to find an element on the page that it is trying to interact with.  
  
->When it is unable to locate the frame that it is attempting to switch to, the NoSuchFrameException is thrown.  
  
->The NoSuchWindowException is thrown when it is unable to find the window that it is attempting to switch to.  
  
->When trying to switch to an alert that is not present, the NoAlertPresentException is thrown.  
  
->The ElementClickInterceptedException is thrown when an element that a program is attempting to click is blocked from being clicked. This can happen if another element on the page is covering the element that the program is trying to click on or if the element is obscured by a modal dialog or pop-up window.  
  
->When an operation is attempted on an element that is not in a valid state for the operation, the InvalidElementStateException is thrown. For instance, if you try to click a button on a webpage using Selenium, but the button is not visible or disabled.

REST vs gRPC vs GraphQL

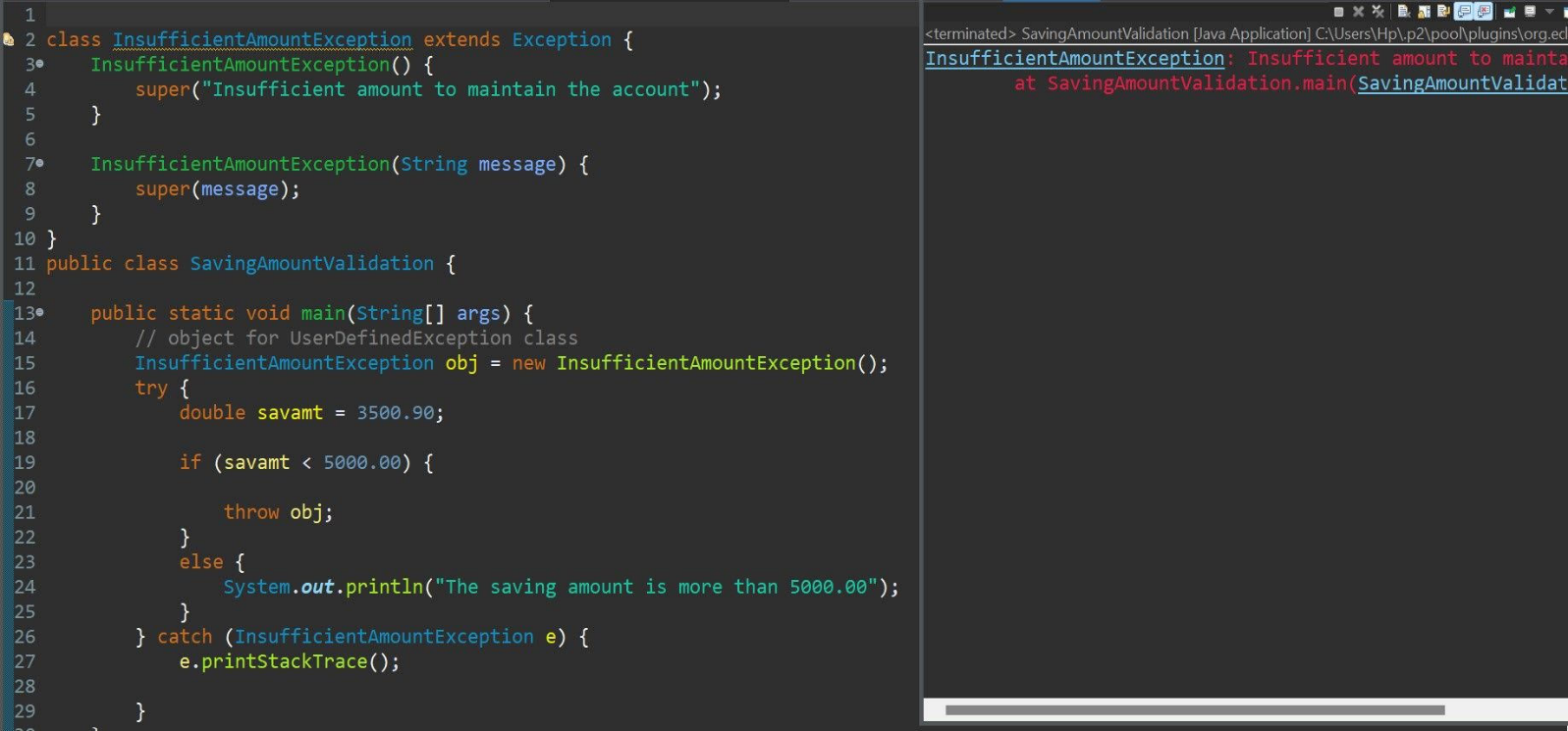
REST vs gRPC vs GraphQL  
  
REST, gRPC, and GraphQL are three popular types of API architectures used for building web-based applications.  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
REST (Representational State Transfer)  
  
> REST is an architectural style used for building web-based APIs that uses HTTP methods such as GET, POST, PUT, and DELETE to interact with resources or data.  
  
> REST APIs are typically stateless, meaning that each request contains all the information needed to process that request.  
  
> REST APIs are commonly used for building web applications that require access to data and resources.  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
gRPC (Remote Procedure Call)  
  
> gRPC is a modern high-performance open-source framework used for building scalable APIs.  
  
> gRPC uses the Protocol Buffers language to define the structure of the API, and supports bi-directional streaming, flow control, and blocking or non-blocking calls.  
  
> gRPC APIs are commonly used for building microservices and distributed systems.  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
GraphQL  
  
> GraphQL is a query language and runtime for APIs that allows clients to define the structure of the data they need.  
  
> GraphQL APIs use a single endpoint and allow clients to specify the data they need using a flexible query language.  
  
> GraphQL APIs are commonly used for building web applications that require access to complex data and resources.

SDET to DevOps RoadMap

f you are someone who is new into the world of DevOps or if you're a SDET and want to grow into the field of DevOps, then this DevOps Roadmap 2023 will help you out.  
  
𝐒𝐭𝐞𝐩 1: 𝐂𝐥𝐨𝐮𝐝 𝐂𝐨𝐦𝐩𝐮𝐭𝐢𝐧𝐠 :  
- Learn the basics of cloud computing and understand the major cloud providers such as AWS, Azure, and GCP.  
- Doesn't Matter what you Cloud Service Provider pick  
- Learn how to deploy and manage applications on cloud platforms.  
- Understand the basics of cloud security and compliance  
  
𝐒𝐭𝐞𝐩 2: 𝐂𝐨𝐧𝐭𝐚𝐢𝐧𝐞𝐫𝐢𝐳𝐚𝐭𝐢𝐨𝐧 𝐚𝐧𝐝 𝐊𝐮𝐛𝐞𝐫𝐧𝐞𝐭𝐞𝐬:  
- Learn the basics of containerization and understand how to use Docker and Kubernetes to deploy and manage containerized applications.  
- Understand how to scale and manage containerized applications  
  
  
𝐒𝐭𝐞𝐩 3: 𝐀𝐮𝐭𝐨𝐦𝐚𝐭𝐢𝐨𝐧 𝐚𝐧𝐝 𝐒𝐜𝐫𝐢𝐩𝐭𝐢𝐧𝐠:  
- Learn how to use automation and scripting tools such as Ansible and Python to automate repetitive tasks and manage infrastructure.  
- Understand how to use Infrastructure as Code (IaC) tools such as Terraform for managing infrastructure is very important.  
  
𝐒𝐭𝐞𝐩 4: 𝐂𝐨𝐧𝐭𝐢𝐧𝐮𝐨𝐮𝐬 𝐈𝐧𝐭𝐞𝐠𝐫𝐚𝐭𝐢𝐨𝐧 𝐚𝐧𝐝 𝐂𝐨𝐧𝐭𝐢𝐧𝐮𝐨𝐮𝐬 𝐃𝐞𝐩𝐥𝐨𝐲𝐦𝐞𝐧𝐭 (𝐂𝐈/𝐂𝐃):  
- Learn how to implement CI/CD pipelines using tools such as Jenkins, GitLab CI, and Travis CI.  
- Understand how to automate the software development process and deploy applications quickly and efficiently  
  
𝐒𝐭𝐞𝐩 5: 𝐌𝐨𝐧𝐢𝐭𝐨𝐫𝐢𝐧𝐠 𝐚𝐧𝐝 𝐋𝐨𝐠𝐠𝐢𝐧𝐠:  
- Learn how to monitor and log the performance of applications and infrastructure using tools such as Prometheus, Grafana, and Elasticsearch.  
- Understand how to identify and troubleshoot issues using monitoring and logging data.  
  
𝐒𝐭𝐞𝐩 6: 𝐒𝐞𝐜𝐮𝐫𝐢𝐭𝐲:  
- Learn how to implement security best practices and use encryption and secure protocols.  
- Understand how to use security tools such as OWASP, SELinux, and  
AppArmor to protect applications and infrastructure.  
  
𝐒𝐭𝐞𝐩 7: 𝐀𝐠𝐢𝐥𝐞 𝐌𝐞𝐭𝐡𝐨𝐝𝐨𝐥𝐨𝐠𝐲:  
- Learn how to work in an agile environment and understand Agile development methodologies such as Scrum and Kanban.  
- Understand how to manage backlog, sprints and releases in an Agile environment.

How to work on user defined exception in java ?

\* We can define our own customized exceptions .  
\* These exceptions can be thrown explicitly using the keyword throw.  
\* Inside the try block can check the condition for exception and use throw keyword and specify the exception object which must be thrown.  
  
Sample User defined exception class example :  
  
class InsufficientAmountException extends Exception {  
   InsufficientAmountException() {  
      super("Insufficient amount to maintain the account");  
   }  
  
   InsufficientAmountException(String message) {  
      super(message);  
  
   }  
  
How to use throw keyword :  
  
public class SavingAmountValidation {  
  
   public static void main(String[] args) {  
      // object for UserDefinedException class  
      InsufficientAmountException obj = new InsufficientAmountException();  
      try {  
         double savamt = 3500.90;  
  
         if (savamt < 5000.00) {  
  
            throw obj;  
         }  
         else {  
            System.out.println("The saving amount is more than 5000.00");  
         }  
      } catch (InsufficientAmountException e) {  
         e.printStackTrace();  
  
      }  
   }  
}



REST API - Path param vs Query param vs Form param

 In RESTful API, there are three common ways to pass data between the client and server: path parameter, query parameter, and form parameter.  
  
Path param vs Query param vs Form param  
  
  
Here's an example of how to use each parameter type with RestAssured in Java:  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 Path parameter  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
Path parameters are used to identify a specific resource in the URL path. They are specified in the URL path and preceded by a colon (:). In RestAssured, we can pass path parameters using the pathParam() method.  
  
 EXAMPLE  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
Suppose we have a RESTful API endpoint that retrieves a user's information based on their ID, which is passed as a path parameter:  
  
GET /users/:id  
  
CODE with Rest Assured  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
To pass the path parameter using RestAssured, we can use the pathParam() method like this:  
  
int userId = 123;  
given()  
.pathParam("id", userId)  
.when()  
.get("/users/{id}")  
.then()  
.statusCode(200);  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  Query parameter  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
Query parameters are used to filter or sort data. They are specified in the URL query string and separated by an ampersand (&). In RestAssured, we can pass query parameters using the queryParam() method.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  EXAMPLE  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
Suppose we have a RESTful API endpoint that retrieves a list of users based on their gender, which is passed as a query parameter:  
  
GET /users?gender=female  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
CODE with Rest Assured  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
To pass the query parameter using RestAssured, we can use the queryParam() method like this:  
  
given()  
.queryParam("gender", "female")  
.when()  
.get("/users")  
.then()  
.statusCode(200);  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
   Form parameter  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
Form parameters are used to submit data to the server in an HTTP form. They are specified in the request body as key-value pairs. In RestAssured, we can pass form parameters using the formParam() method.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
   EXAMPLE  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
Suppose we have a RESTful API endpoint that creates a new user based on their name and email, which are passed as form parameters:  
  
POST /users  
Content-Type: application/x-www-form-urlencoded  
name=Sidharth&email=[sidharth@gmail.com](mailto:sidharth@gmail.com)  
  
\*\*\*\*\*\*\*\*\*\*\*\*  
CODE  
\*\*\*\*\*\*\*\*\*\*\*\*  
  
To pass the form parameters using RestAssured, we can use the formParam() method like this:  
  
given()  
.formParam("name", "Sidharth")  
.formParam("email", "[sidharth@gmail.com](mailto:sidharth@gmail.com)")  
.when()  
.post("/users")  
.then()  
.statusCode(201);

static in Java:

static in Java:  
  
static is a keyword in Java that is used before a variable, method, block and an inner class.  
  
If static is used for a variable, single copy of the variable is used through the program.  
Ex: static int pin = 12345;  
  
If a method is declared as static then it can be called without using the object of the class. Main method is also declared as static.  
Ex: public static getData(){  
//code  
}  
  
If a block is declared as static then it will be executed before any methods. It runs at the time of class loading.  
Ex: static {  
//code  
}  
  
A class inside another class which is called as inner class can be declared as static  
Ex:  
class MainClass {  
static class InnerClass {  
// code  
}  
}

## some important Selenium WebDriver and WebElement commands in the Java API:

**WebDriver Commands:**  
get(String url) : - Load a new web page in the current browser window.  
getCurrentUrl():- Get the URL of the current page.  
getTitle(): Get the title of the current page.  
getWindowHandle(): Get a unique identifier for the current window.  
getWindowHandles(): Get a set of unique identifiers for all open windows.  
switchTo().window(String windowHandle): Switch to the window with the given window handle.  
navigate().to(String url): Navigate to a new URL.  
navigate().back(): Move the browser back one page.  
navigate().forward(): Move the browser forward one page.  
navigate().refresh(): Refresh the current page.  
  
**WebElement Commands:**  
click(): Click the element.  
submit(): Submit a form.  
getText(): Get the visible text of the element.  
getAttribute(String name): Get the value of a specified attribute of the element.  
sendKeys(CharSequence... keysToSend): Simulate typing into an element.  
clear(): Clear the content of an element.  
isDisplayed(): Determine if the element is displayed on the page.  
isEnabled(): Determine if the element is enabled or not.  
isSelected(): Determine if the element is selected or not.  
getSize(): Get the size of the element.  
getLocation(): Get the location of the element.

What is the role of QA engineer in a Sprint planning meeting?

* Many consider that QA person do not need to contribute to the sprint planning. However, that could be a wrong perspective.
* My insights so far on the QA’s role in [#sprintplanning](https://www.linkedin.com/feed/hashtag/?keywords=sprintplanning&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7031481350079619072)
* Identity the user stories that could be ready for Validation in the same sprint and user stories QA ready in n+1 sprint.
* Multiple spike stories or multiple Development stories might crowd QA efforts in a single sprint
* Determine the dependency with other teams to execute test cases
* Raise a flag on the pending bugs from previous sprint to be included in the next sprint
* Analyze the Production Stories and Non Prod stories and how they are spread across the sprint.
* These pointers could help plan the QA efforts better and would reduce surprises in between the sprints.

What is the Assertions (also known as Asserts) ?

* [#Assertions](https://www.linkedin.com/feed/hashtag/?keywords=assertions&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) are used for validating a [#testcase](https://www.linkedin.com/feed/hashtag/?keywords=testcase&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) and helps us understand if a test case has passed or failed. The [#assertion](https://www.linkedin.com/feed/hashtag/?keywords=assertion&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) is considered to be met if the actual result of an application matches with that of the expected result.  
  While [#automating](https://www.linkedin.com/feed/hashtag/?keywords=automating&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) web applications using [#Selenium](https://www.linkedin.com/feed/hashtag/?keywords=selenium&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848), we need to validate our tests to verify if they are working as expected or not (that is, if a test case result is pass/fails).  
    
  Actual result is compared with expected result with the help of Assertion. For creating assertion we are going to use the [#Assert](https://www.linkedin.com/feed/hashtag/?keywords=assert&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) class provided by [#TestNG](https://www.linkedin.com/feed/hashtag/?keywords=testng&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848).  
    
   Types Of Assertions In Selenium  
  There are two types of assertions in Selenium and the categorization depends on how the assertion behaves after a condition is pass or fail.  
  1) Hard Assertions  
  2) Soft Assertions
* Hard Assertions (Or Simply Assertions)  
   hard assertion does not continue with [#execution](https://www.linkedin.com/feed/hashtag/?keywords=execution&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) until the assertion condition is met.  
  Hard assertions usually [#throw](https://www.linkedin.com/feed/hashtag/?keywords=throw&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) an Assertion Error whenever an assertion condition has not been met. The test case will be immediately marked as Failed when a hard assertion condition fails.  
  A [#scenario](https://www.linkedin.com/feed/hashtag/?keywords=scenario&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) to use this kind of assertion is that, when you want to verify if you have logged in correctly and fail the test if you haven’t made a successful login, as there is no point in proceeding further if the pre-condition(login) itself fails.  
    
  Syntax:  
  String ActualTitle = driver.getTitle();  
  String ExpectedTitle = “Welcome to Amazon”;  
  Assert.assertEquals(ActualTitle, ExpectedTitle);
* Soft Assertions  
  A soft assertion continues with the next step of the test execution even if the assertion condition is not met.  
  Soft [#Assertions](https://www.linkedin.com/feed/hashtag/?keywords=assertions&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) are the type of assertions that do not throw an [#exception](https://www.linkedin.com/feed/hashtag/?keywords=exception&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) automatically when an assertion fails unless it is asked for. This is useful if you are doing multiple [#validations](https://www.linkedin.com/feed/hashtag/?keywords=validations&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) in a form, out of which only a few validations directly have an impact on deciding the test case status.  
  Here, we use a class called [#SoftAssert](https://www.linkedin.com/feed/hashtag/?keywords=softassert&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848) and the method assertAll() is called to throw all exceptions caught during [#execution](https://www.linkedin.com/feed/hashtag/?keywords=execution&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A7037318334534606848). When softAssert is used, it performs assertion and if an exception is found, its not thrown immediately, rather it continues until we call the method assertAll() to throw all exceptions caught.  
    
  Syntax:  
  SoftAssert softassert = new SoftAssert();  
  String ActualTitle = driver.getTitle();  
  String ExpectedTitle = “Welcome to Amazon”;  
  softassert.assertEquals(ActualTitle, ExpectedTitle);  
  softassert.assertAll();

Git is a popular version control system that is widely used in industry for managing source code and collaborating on software development projects. Here are some of the most commonly used Git commands along with detailed explanations of each step:  
  
1. git init  
This command initializes a new Git repository in the current directory. It creates a new .git directory, which contains all the necessary files and folders for version control.  
  
2. git clone  
This command is used to create a local copy of a remote Git repository. It downloads all the files and folders from the remote repository to your local machine.  
  
3. git add  
This command adds files to the staging area, which prepares them for committing. It is used to tell Git which files should be included in the next commit.  
  
4. git commit  
This command creates a new commit, which is a snapshot of the changes made to the files in the staging area. It is used to save changes to the local repository.  
  
5. git push  
This command is used to upload the local repository to a remote Git repository. It sends all the new commits to the remote repository, making them available to other team members.  
  
6. git pull  
This command is used to download the latest changes from a remote Git repository. It fetches all the new commits from the remote repository and merges them into the local repository.  
  
7. git branch  
This command is used to create a new branch in the Git repository. It allows you to work on multiple features or bug fixes in parallel without affecting the main branch.  
  
8. git checkout  
This command is used to switch between different branches in the Git repository. It allows you to work on a specific branch and make changes without affecting other branches.  
  
9. git merge  
This command is used to merge two or more branches in the Git repository. It combines the changes made in different branches and creates a new commit with the merged changes.  
  
10. git log  
This command is used to view the commit history of the Git repository. It shows a list of all the commits made to the repository, along with the author, date, and commit message.

Install QR code ZXing jars with Selenium :

For **Maven** Users:

<dependencies>

<dependency>

<groupId>com.google.zxing</groupId>

<artifactId>core</artifactId>

<version>3.3.0</version>

</dependency>

<dependency>

<groupId>com.google.zxing</groupId>

<artifactId>javase</artifactId>

<version>3.3.0</version>

</dependency>

</dependencies>

People who just want to download jars, they have to download two jar files:

* ZXing Core : **https://github.com/zxing/zxing/releases**
* ZXing Java SE Extensions

: **https://mvnrepository.com/artifact/com.google.zxing/javase**

**Steps to Read QR Code in Selenium :**

1. Have the QR code image ready in your local machine, download below Image. I have stored qr\_code\_text as the text in the bar code.

2.

1. Create a File object for the image, so that the system understands the image as File.

File file = new File("path of QR image in your local system"); String decodedText = null;

1. So far we are calling a thing on the computer as an image, in above step we have converted that thing into File, Now we should make the system to understand that the File is Image. We use ImageIO classes to read the file into an image.

// store the file as an image

BufferedImage bufferedImage = ImageIO.read(file);

1. Now we have to clear all the other parts in the image and convert the image into a bitmap.

// process the image

LuminanceSource source = new BufferedImageLuminanceSource(bufferedImage); BinaryBitmap bitmap = new BinaryBitmap(new HybridBinarizer(source));

1. Decode the details from the bitmap, and store it in the Result object. In the below code we are extracting only the text.

// store the details of the QR code

Result result = new MultiFormatReader().decode(bitmap); decodedText = result.getText();

1. Now **compare the text** retrieved from the image with the expected image.

// testng assertion

Assert.assertEquals(decodedText, "your expected text");

**Java Selenium Code for QR automation code :**

package com.qa.dummytest;

import java.awt.image.BufferedImage; import java.io.IOException;

import java.net.URL; import java.time.Duration;

import javax.imageio.ImageIO; import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver; import org.testng.Assert;

import org.testng.annotations.AfterClass; import org.testng.annotations.BeforeClass; import org.testng.annotations.Test;

import com.google.zxing.BinaryBitmap; import com.google.zxing.LuminanceSource; import com.google.zxing.MultiFormatReader; import com.google.zxing.NotFoundException; import com.google.zxing.Result;

import com.google.zxing.client.j2se.BufferedImageLuminanceSource; import com.google.zxing.common.HybridBinarizer;

public class QR\_Code {

public static WebDriver *driver*; @BeforeClass

public void setup()

{

*driver* = new ChromeDriver(); *driver*.manage().window().maximize(); *driver*.manage().deleteAllCookies();

*driver*.manage().timeouts().pageLoadTimeout(Duration.*ofSeconds*(30)); *driver*.manage().timeouts().implicitlyWait(Duration.*ofSeconds*(25)); *driver*.get("[http://qrcode.meetheed.com/qrcode\_examples.php"](http://qrcode.meetheed.com/qrcode_examples.php));

}

@Test(priority=1)

public void Validate\_QR\_code\_text1() throws IOException, NotFoundException

{

//Get Image path src link as URL - storing as string

String qr\_code\_text = *driver*.findElement(By.*xpath*("//img[@alt='QR Contact Example']")).getAttribute("src");

//Create an object of URL class URL url = new URL(qr\_code\_text);

// If you have Image file path then remove above line of code & add File() -path to it

//File file = new File("path of QR code image");

//pass URL class object to store the file as image BufferedImage bufferedimage = ImageIO.*read*(url);

//Process the image

LuminanceSource luminanceSource = new BufferedImageLuminanceSource(bufferedimage); BinaryBitmap binaryBitmap = new BinaryBitmap(new HybridBinarizer(luminanceSource));

//To Extract information from QR code;

Result result = new MultiFormatReader().decode(binaryBitmap); String decodedText = result.getText(); System.*out*.println(decodedText);

//Assertion

Assert.*assertEquals*(decodedText, "MECARD:N:Joe;EMAIL:Joe@bloggs.com;;");

}

@Test(priority=2)

public void Validate\_QR\_code\_text2() throws IOException, NotFoundException

{

//Get Image src link as URL - storing as string

String qr\_code\_text = *driver*.findElement(By.*xpath*("//img[@alt='QR Phone Number Example']")).getAttribute("src");

//Create an object of URL class URL url = new URL(qr\_code\_text);

//pass URL class object to store the file as image BufferedImage bufferedimage = ImageIO.*read*(url);

//Process the image

LuminanceSource luminanceSource = new BufferedImageLuminanceSource(bufferedimage); BinaryBitmap binaryBitmap = new BinaryBitmap(new HybridBinarizer(luminanceSource));

//To Extract information from QR code;

Result result = new MultiFormatReader().decode(binaryBitmap); String decodedText = result.getText(); System.*out*.println(decodedText);

//Assertion

Assert.*assertEquals*(decodedText, "tel:07777777777"); //write your expected text

}

@AfterClass

public void teardown()

{

*driver*.close();

}

}

: 𝑺𝒆𝒍𝒆𝒏𝒊𝒖𝒎 vs. 𝑪𝒚𝒑𝒓𝒆𝒔𝒔 

 𝑺𝒆𝒍𝒆𝒏𝒊𝒖𝒎 is a popular 𝒐𝒑𝒆𝒏-𝒔𝒐𝒖𝒓𝒄𝒆 𝒕𝒆𝒔𝒕𝒊𝒏𝒈 𝒇𝒓𝒂𝒎𝒆𝒘𝒐𝒓𝒌 that has been around for years. It's a powerful tool that supports multiple programming languages and provides a wide range of testing capabilities. On the other hand, 𝑪𝒚𝒑𝒓𝒆𝒔𝒔 is a 𝒓𝒆𝒍𝒂𝒕𝒊𝒗𝒆𝒍𝒚 𝒏𝒆𝒘 𝒕𝒆𝒔𝒕𝒊𝒏𝒈 𝒇𝒓𝒂𝒎𝒆𝒘𝒐𝒓𝒌 that has gained popularity in recent years for its ease of use and fast test execution speed.

 Use 𝑺𝒆𝒍𝒆𝒏𝒊𝒖𝒎 when:

𝘠𝘰𝘶 𝘯𝘦𝘦𝘥 𝘵𝘰 𝘵𝘦𝘴𝘵 𝘢 𝘸𝘪𝘥𝘦 𝘳𝘢𝘯𝘨𝘦 𝘰𝘧 𝒘𝒆𝒃 𝒂𝒑𝒑𝒍𝒊𝒄𝒂𝒕𝒊𝒐𝒏𝒔 𝘢𝘤𝘳𝘰𝘴𝘴 𝒅𝒊𝒇𝒇𝒆𝒓𝒆𝒏𝒕 𝒃𝒓𝒐𝒘𝒔𝒆𝒓𝒔 𝘢𝘯𝘥 𝒑𝒍𝒂𝒕𝒇𝒐𝒓𝒎𝒔.

You need to test complex scenarios that require advanced interaction with the web page, such as 𝒑𝒐𝒑-𝒖𝒑 𝒘𝒊𝒏𝒅𝒐𝒘𝒔 𝒂𝒏𝒅 𝒊𝒇𝒓𝒂𝒎𝒆𝒔.

You need to 𝒊𝒏𝒕𝒆𝒈𝒓𝒂𝒕𝒆 𝒚𝒐𝒖𝒓 𝒕𝒆𝒔𝒕𝒔 𝒘𝒊𝒕𝒉 𝒐𝒕𝒉𝒆𝒓 𝒕𝒐𝒐𝒍𝒔 𝒂𝒏𝒅 𝒇𝒓𝒂𝒎𝒆𝒘𝒐𝒓𝒌𝒔, 𝒔𝒖𝒄𝒉 𝒂𝒔 𝑱𝒆𝒏𝒌𝒊𝒏𝒔, 𝑻𝒆𝒔𝒕𝑵𝑮, 𝒂𝒏𝒅 𝑴𝒂𝒗𝒆𝒏.

 Use 𝑪𝒚𝒑𝒓𝒆𝒔𝒔 when:

You need a 𝒇𝒂𝒔𝒕 𝒂𝒏𝒅 𝒓𝒆𝒍𝒊𝒂𝒃𝒍𝒆 𝒕𝒆𝒔𝒕𝒊𝒏𝒈 𝒇𝒓𝒂𝒎𝒆𝒘𝒐𝒓𝒌 𝒇𝒐𝒓 𝒎𝒐𝒅𝒆𝒓𝒏 𝒘𝒆𝒃 𝒂𝒑𝒑𝒍𝒊𝒄𝒂𝒕𝒊𝒐𝒏𝒔 that use 𝑱𝒂𝒗𝒂𝑺𝒄𝒓𝒊𝒑𝒕 𝒇𝒓𝒂𝒎𝒆𝒘𝒐𝒓𝒌𝒔 𝒔𝒖𝒄𝒉 𝒂𝒔 𝑹𝒆𝒂𝒄𝒕, 𝑨𝒏𝒈𝒖𝒍𝒂𝒓, 𝒂𝒏𝒅 𝑽𝒖𝒆.

You need a 𝒔𝒊𝒎𝒑𝒍𝒆 𝒂𝒏𝒅 𝒆𝒂𝒔𝒚-𝒕𝒐-𝒖𝒔𝒆 𝒕𝒐𝒐𝒍 that requires 𝒎𝒊𝒏𝒊𝒎𝒂𝒍 𝒔𝒆𝒕𝒖𝒑 𝒂𝒏𝒅 𝒄𝒐𝒏𝒇𝒊𝒈𝒖𝒓𝒂𝒕𝒊𝒐𝒏.

You need a tool that provides 𝒃𝒖𝒊𝒍𝒕-𝒊𝒏 𝒗𝒊𝒔𝒖𝒂𝒍 𝒕𝒆𝒔𝒕𝒊𝒏𝒈 𝒄𝒂𝒑𝒂𝒃𝒊𝒍𝒊𝒕𝒊𝒆𝒔 and helps you catch UI regressions early in the development cycle.

: Ultimately, the choice between 𝑺𝒆𝒍𝒆𝒏𝒊𝒖𝒎 and 𝑪𝒚𝒑𝒓𝒆𝒔𝒔 depends on your testing needs and preferences. Both tools have their 𝒔𝒕𝒓𝒆𝒏𝒈𝒕𝒉𝒔 and 𝒘𝒆𝒂𝒌𝒏𝒆𝒔𝒔𝒆𝒔, and it's up to you to decide which one is best for your project.

Resume TipS

𝑻𝒊𝒑 1: 𝐊𝐞𝐞𝐩 𝐢𝐭 𝐒𝐦𝐚𝐥𝐥  
One common mistake that people do is their resume is 3 or 4 pages long.  
There cannot be any reason that your resume needs to be more than 2 page long.  
Ideal length of a resume has to 1 page only.  
  
In todays time no one has time to read a 4 page CV. Do you even watch the a full videos on Youtube? Hell no  
  
Companies gets 1000s of resume for a one position so make sure to get to the point and just write what's important.  
  
  
𝑻𝒊𝒑 2: 𝐒𝐨𝐜𝐢𝐚𝐥 𝐏𝐫𝐨𝐟𝐢𝐥𝐞 𝐚𝐧𝐝 𝐑𝐞𝐩𝐨 𝐋𝐢𝐧𝐤  
2023 is all together a different time right now. You need to have a updated  
linked in profile and if you are technical person you should have a  
github or gitlab public repo where you can show case your learning.  
Create a sample automation framework so that interviewer can see how you write codes and how you will be building framework.  
  
This will help the interviewer get confidence on your profile and really help you stand out more real and professional.  
  
𝑻𝒊𝒑 3: 𝐀𝐝𝐝𝐞𝐝 𝐏𝐫𝐨𝐩𝐞𝐫 𝐬𝐤𝐢𝐥𝐥𝐬.  
Please don't tell you know excel, powerpoint, Bugzilla and that you know how to work on windows OS.  
  
I have attached important skills a SDET folks needs to have.  
  
𝑻𝒊𝒑 4: 𝐄𝐝𝐮𝐜𝐚𝐭𝐢𝐨𝐧.  
Once you are graduated. No need to add your 10th Marks ,12 th Marks. Where you did schooling.  
Only information necessary is your graduation and post graduation.  
  
𝐓𝐢𝐩 5:  
Work Experience:  
Please dont explain what your project does and domain in resume.  
Your Project Experience needs to be only consist three 3 things:  
1. 𝐑𝐨𝐥𝐞𝐬 𝐚𝐧𝐝 𝐑𝐞𝐬𝐩𝐨𝐧𝐬𝐢𝐛𝐢𝐥𝐢𝐭𝐢𝐞𝐬  
2. 𝐓𝐞𝐜𝐡𝐧𝐨𝐥𝐨𝐠𝐢𝐞𝐬 𝐔𝐬𝐞𝐝 𝐛𝐲 𝐲𝐨𝐮  
3. 𝐀𝐜𝐡𝐢𝐞𝐯𝐞𝐦𝐞𝐧𝐭.  
  
𝘈𝘥𝘥𝘪𝘯𝘨  
 𝘈𝘤𝘩𝘪𝘦𝘷𝘦𝘮𝘦𝘯𝘵𝘴 𝘢𝘳𝘦 𝘮𝘢𝘯𝘥𝘢𝘵𝘰𝘳𝘺 𝘦𝘷𝘦𝘳𝘺 𝘰𝘯𝘦  
𝘳𝘦𝘴𝘶𝘮𝘦 𝘴𝘩𝘰𝘸𝘴 𝘵𝘩𝘢𝘵 𝘺𝘰𝘶 𝘬𝘯𝘰𝘸 𝘩𝘰𝘸 𝘵𝘰  
𝘮𝘦𝘢𝘴𝘶𝘳𝘦 𝘺𝘰𝘶𝘳 𝘸𝘰𝘳𝘬 𝘢𝘯𝘥 𝘩𝘰𝘸 𝘺𝘰𝘶 𝘸𝘦𝘳𝘦 𝘢𝘣𝘭𝘦  
𝘵𝘰 𝘣𝘳𝘪𝘯𝘨 𝘢𝘶𝘵𝘰𝘮𝘢𝘵𝘪𝘰𝘯 𝘢𝘯𝘥 𝘢𝘥𝘥 𝘷𝘢𝘭𝘶𝘦 𝘵𝘰  
𝘱𝘳𝘰𝘫𝘦𝘤𝘵 𝘲𝘶𝘢𝘭𝘪𝘵𝘺.  
  
𝐀𝐧 𝐈𝐜𝐢𝐧𝐠 𝐨𝐧 𝐭𝐡𝐞  
𝐂𝐚𝐤𝐞 𝐢𝐬 𝐚 𝐂𝐨𝐯𝐞𝐫 𝐥𝐞𝐭𝐭𝐞𝐫 𝐰𝐡𝐲 𝐲𝐨𝐮 𝐬𝐡𝐨𝐮𝐥𝐝 𝐛𝐞  
 𝐡𝐢𝐫𝐞𝐝 𝐚𝐧𝐝 𝐰𝐡𝐚𝐭 𝐲𝐨𝐮 𝐰𝐢𝐥𝐥 𝐝𝐨 𝐢𝐦𝐩𝐫𝐨𝐯𝐞 𝐭𝐡𝐞  
𝐪𝐮𝐚𝐥𝐢𝐭𝐲 𝐨𝐟 𝐭𝐡𝐞 𝐩𝐫𝐨𝐣𝐞𝐜𝐭. 