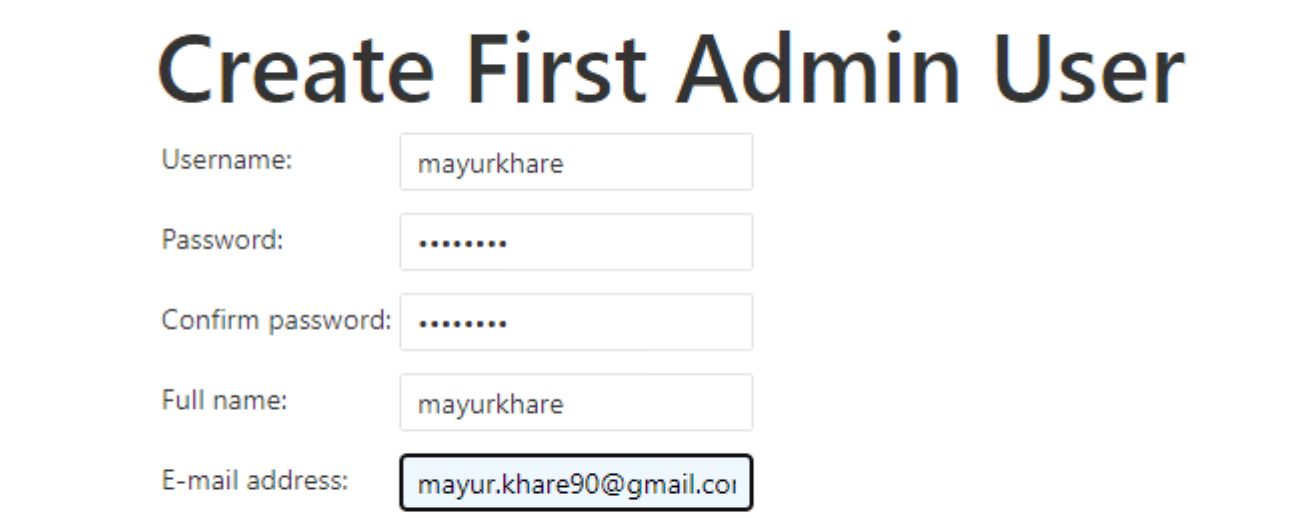
Jenkins

28 March 2022 20:10



Pswrd - Lilly@34

Java_Concept

01 April 2022 21:17

Que - What is the difference between Abstract Class and Interface?
Ans -

Que - Difference between check and unchecked exception?
Ans-

Que - How you handle exceptions in java?
Ans-

Que - Can we write try block without catch and finally?
Ans -

Que - What is finally block?
Ans -

Que - What is the difference between classnotfound and class not def?
Ans-

Que - Can we throw exception deliberatley
Ans

Que - Is there any condition if we can ignore finally block?
Ans

Que - Automate the calendar of make my trip?
Ans

Que - Difference between Exception and Errors?
Ans - Exceptions are occur mainly due to our programs and Error are something which occur due to lack of system resources like les s RAM, slow processor, so programmer cannot do anything.

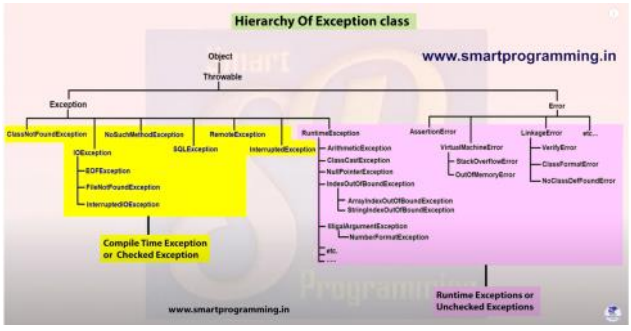
Exceptions can be recoverable(means we can handle them from exception handling) and error is not recoverable(Programmer canno t handle the error, for this programmer need to contact IT person)

Exceptions are two type "Compile time exception"/Checked exception and "Runtime exception/Unchecked exception
Errors are only "Runtime exception/Unchecked exception

CompileTime Exception - Compile time exception are those exception which Java can catch at compile time that this exception can occur in future.

RunTime Exception - Runtime exception are those exception which Java cannot catch at compile time and they occur at runtime.

Heriarchy of exception class



Que - How we can handle exception...?
Ans - Exception can be handled by 5 keywords
try, catch, final, throw, throws

Que - How to do exception testing in testng?

```
import org.testng.annotations.Test;

public class ExceptionTest
{
    @Test(expectedExceptions={ArithmeticException.class})
    public void exceptionTesting()
    {
        int i = 1/0;
        System.out.println("Value of i = "+i);
    }
}
```

Que - Can we use finally block with try?
Ans - yes, but exceptional handling will not be done.

Que - Can we use finally block with try and without catch?
Ans - yes, but exceptional handling will not be done.

Que - What is finally keyword?
Ans - finally will be executed every time either exception occurs or not.

Que - Can we use finally block without try and catch alone
Ans - No

****

Note-

1. We can use multiple catch blocks with one try block but we can only use single finally block with one try block, not multiple .
2. The statements present in finally block executes even if the try block contains control transfer statements (i.e. jump statem ents) like return, break or continue.

3. There are 4 possibilities where finally block will not execute after try block those are given below

a. Using of system.exit() method.
b. Causing a fatal error that causes the process to abort.
c. Due to an exception arising in finally block itself.
d. The death of the thread.

****

Que - Can we write alone try, catch and finally?
Ans - No

Que - Can we use multiple catch block with one try block?
Ans - yes, but if first catch block contains the parent exception class then we cannot write the second catch block which contains its child class.

```
try
{
    //any code
}
catch( Exception e)
{
    e.stacktrace
}
catch ( ArithmeticException e)
```

{

}

This code will not work as its parent class exception is already present in the first catch block so we cannot write a catch block which contains its child exception class (Exception is a parent class of all exception classes)

Que - Can we write nested try-catch block (for ex- can we write try under any try block)?
Ans - yes we can write nested try catch block but for every try block there should be catch block.

Que - Can we write try block under catch block?
Ans - yes (but every try should have its catch block)

Que - Can we write try catch block under finally block?
Ans yes we can write try catch block in a finally block.

Que - Can this sequence is correct (try, finally and then catch)?
Ans - No.

Que - Can we write any statement (any line of code) in between try and catch?
Ans - No

Que - Is this sequence correct try finally then try catch?
Ans yes

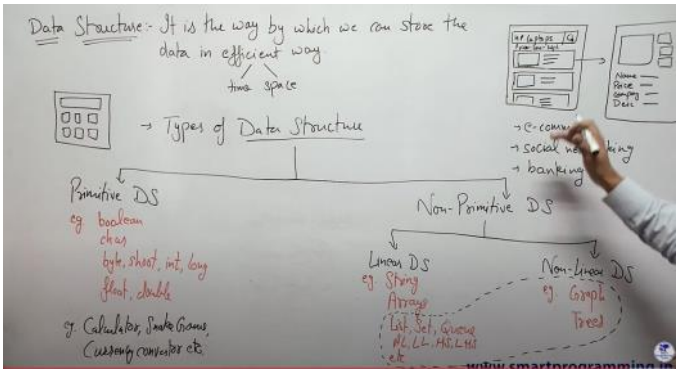
Que - What is the syntax of throw keyword?
Ans - throw new ExceptionclassName(<----any string/message---->);

Que - Difference between throws and throw keyword?
Ans -

Sr. no.	Basis of Differences	throw	throws
1.	Definition	Java throw keyword is used throw an exception explicitly in the code, inside the function or the block of code.	Java throws keyword is used in the method signature to declare an exception which might be thrown by the function while the execution of the code.
2.	Type of exception Using throw keyword, we can only propagate unchecked exception i.e., the checked exception cannot be propagated using throw only.	Using throws keyword, we can declare both checked and unchecked exceptions. However, the throws keyword can be used to propagate checked exceptions only.	
3.	Syntax	The throw keyword is followed by an instance of Exception to be thrown.	The throws keyword is followed by class names of Exceptions to be thrown.
4.	Declaration	throw is used within the method.	throws is used with the method signature.
5.	Internal implementation	We are allowed to throw only one exception at a time i.e. we cannot throw multiple exceptions.	We can declare multiple exceptions using throws keyword that can be thrown by the method. For example, main() throws IOException, SQLException.

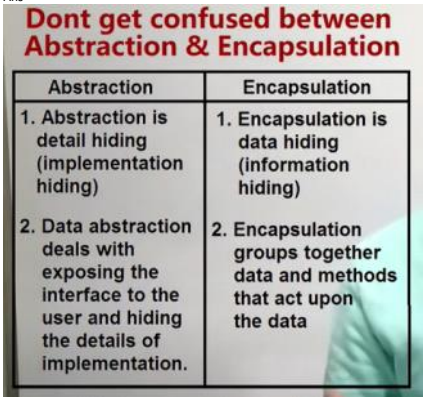
Que : What is the use of throws keyword?
Ans - throws keyword is used to declare the exception i.e. it indicates the caller method that given exception can occur so we have
To handle it with using try and catch block or again declare it by using throws keyword.

Que - What is Data Structure?



Que - What is Abstraction?
Ans - Abstraction is hiding internal implementation & just highlighting the setup services that we are offering.

Que - Is Abstraction & Encapsulation is same.?
Ans



Que - How we can achieve Abstraction?
Ans - There are two ways to achieve abstraction

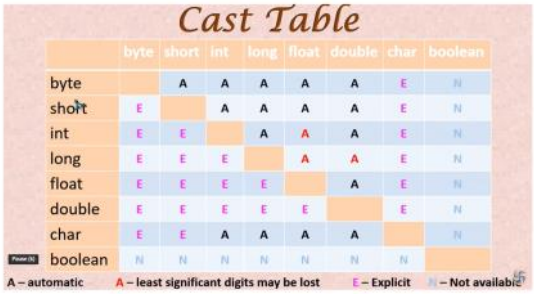
1. By using Abstract Class (100% or 0% abstraction can be achieved)
2. By Interfaces (100% abstraction can be achieved)

a. A method without a body (no implementation) is known as an abstract method
b. A method must always be declared in an abstract class, or we can say that if a class has an abstract method, class should be declared abstract as well.
c. If a class is mentioned as an abstract class then it's not mandatory that the method should also be an abstract method, non -abstract method will also work.
d. If a regular class extends an abstract class, then the class must have to implement all the abstract methods of the abstract parent class or it has to be declared abstract as well.
e. Abstract classes cannot be instantiated, which means we can't create an object of Abstract class.

Que - What is Selenium Webdriver?
Ans -

1. Selenium webdriver is the main component of selenium.
2. Selenium web driver is nothing but a collection of API's whose purpose is to automate web applications.

Que - Conversion chart one data type to another?



****

HashMap

Que -1 What is HashMap?
Ans - HashMap is an associative array data structure, Stores data in the form of key -value pair

- HashMap works on Hashing
- Java uses special method which is coming from object class and called Hash code method.

Ans - So, suppose as we can see in use hashmap we use put method as can be seen in Diagram 1, Now as put method gets call, we are sending key and value pair (Naveen,100), so now as it will go inside the put method, now first hash (int hash, Diagram 2) will get calculated with the help of hashcode method.

Que -2 How HashMap actually works?

Hash code will be calculated for only the key from hashcode method (hashCode(key), Diagram 2) not for the key value pair.
Now for Naveen one hashcode is generated suppose it's been calculated like [hash("Naveen")= 210678], so now one index will be calculated for this code say 'index 4' [int index = hash & (n -1), Diagram 2], So the [Key, value, HashCode, Next(this will be null)] these all four will get store in index 4

So now for Tom suppose hashcode generated like [hash("Tom")=210780], so now one index will be calculated for this code say 'index 9' [int index = hash & (n -1), Diagram 2], So the [Key, value, HashCode, Next] these all four will get store in index 9.

So now for Lisa suppose hashcode generated like [hash("Naveen")=210678], this is possible that for multiple key same hash code generated like here Lisa and Naveen has same hashcode so now one index will be calculated for this code and that will be obviously 'index 4' [int index = hash & (n -1), Diagram 2], So the [Key, value, HashCode, Next] these all four will get store in index 4, but now these 4 will get store in the 'Null' section of Naveen node, So now this will be called as Linked List and also when we get same index for multiple key then this condition called 'HashMap Collision'.

Diagram 1

```
Map<String, Integer> marks = new
HashMap<String, Integer>();

        marks.put("Naveen", 100);
        marks.put("Tom", 200);
        marks.put("Lisa", 300);
        marks.put("Peter", 400);
        marks.put("Robby", 600);
```

Diagram 2

```
public V put(K key, V value){

    int hash = hashCode(key);

    int index = hash & (n-1);

n = 16 (map size)
}
```

Que -3 What is the requirement for an object to be used as key or value in HashMap?
Ans - Both k and v should implement hashcode() and equals() method.

Que -4 What will happen if we try to store a key which is already present in HashMap?
Ans - Nothing will happen, just it will update the value of the key with the new value of that key, it will not throw any exception.
For ex - marks.put("Mayur",100) and now if we add marks.put("Mayur",400), it will replace the 100 and will store the 400 value, so when you will try to fetch the value of Mayur with the help of get method then 400 will be the outcome.

Que -5 Can we store a null key in Java HashMap?
Ans - Yes, we can store only one Null key in HashMap.

Que -6 Can we store a null value in Java HashMap?
Ans - Yes we can store multiple null values but only one null key.

Que - 7How does HashMap handle collisions in Java?
Ans - This is explained in above question, marked in yellow.

Que -8 Which Data Structure HashMap represents?
Ans - HasHMap represent HasTable Data Structure by using LinkedList and BalancedBinaryTree.

Que -9 Can we store duplicate key in HashMap?
Ans - No.

Que -10 Can we store duplicate value in HashMap?
Ans - Yes but with different key.

Que - 11 Is HashMap thread-safe in Java?
Ans - No this is not thread-safe or not synchronized.

Que - 12 What will happen if you use HashMap in a multithreaded Java application?
Ans - If we use HashMap in a multithreaded java application then hashmap can get corrupted because of working of different object on same map.

Que - 13 What are the different ways to iterate over HashMap in Java?
Ans - By using keyset and iterator
By using entrySet and iterator
By using entrySet and enhanced for Loop
By using keySet and get() method
By using forEach with lambda,
Can see the highlighted below in yellow

Que - 14 How do you remove a mapping while iterating over HashMap in Java?
Ans - use remove method

Que 15 - In which order mappings are stored in HashMap?
Ans - Random order because HashMap doesn't provide any ordering guarantee for keys, values or entries. When you iterate over a HashMap, you may get a different order every time you iterate over it.

Que- 16 Can you sort HashMap in Java?
Ans - No, we cannot sort HashMap in java with the help of key and value, but we can convert this to LinkedHashMap which maintains the insertion order(Sorting)

Que 17- What is the load factor in HashMap?
Ans - A load factor is a number that controls the resizing of HashMap when a number of elements in the HashMap cross the load factor as if the load factor is 0.75 and when becoming more than 75% full then resizing trigger which involves array copy.

Que 18 - How does resizing happens in HashMap?
Ans - same ans above

Que 19 - How many entries you can store in HashMap? What is the maximum limit?
Ans - Until run you out of the bucket, run out of the memory, no limit.

Que - 20 What is the difference between the capacity and size of HashMap in Java?
Ans - The capacity denotes how many entries HashMap can store, and size denotes how many mappings or key/value pair is currently present.

Que 21 - What will happen if two different keys of HashMap return the same hashcode()?
Ans - if same hashcode appears then collision will occur and they will maintain Linkedlist internally and after jdk 1.8, if threshold value reached to 8 then from 9th node onward it will get converted into Balanced BinaryTree

Iteration in HashMap

```
public static void main(String[] args) {

		HashMap<String, String> capitalmap = new HashMap<String, String>();
		capitalmap.put("India","Delhi");
		capitalmap.put("USA","Washington");
		capitalmap.put("UK","London");
		capitalmap.put("Spain","Madrid");

		System.out.println("********************Iterator1*************************");

		//iterator - 1 : over the keys: by using keySet()

		Iterator<String> it = capitalmap.keySet().iterator();

		while(it.hasNext())
		{
			String key = it.next();// to get the key
			String value = capitalmap.get(key);  //to get the value
			System.out.println(key+" "+value);

		}

		System.out.println("");
		System.out.println("********************Iterator2*************************");

		//iterator - 2 : over the set (pair): by using entrySet

		Iterator<Entry<String, String>> it1 = capitalmap.entrySet().iterator();

		while(it1.hasNext())
		{
			Entry<String, String> keyValue = it1.next();// to get the keyvalue pair
			System.out.println(keyValue.getKey()+ " "+keyValue.getValue());


		}

		System.out.println("");
		System.out.println("********************Iterator3*************************");

		//iterator - 3 : using java 8 for each and lambda

		capitalmap.forEach((k,v)->System.out.println(k+" "+v));

	}
```

Que - How to sort HashMap?

```
public static void main(String[] args) {


		Map<String, Integer> capitalmapUnsorted = new HashMap<String, Integer>();
		capitalmapUnsorted.put("India",100);
		capitalmapUnsorted.put("USA",100);
		capitalmapUnsorted.put("UK",100);
		capitalmapUnsorted.put("Spain",100);

		Iterator<Entry<String, Integer>> it = capitalmapUnsorted.entrySet().iterator();

		while(it.hasNext())
		{
			Entry<String, Integer> entry = it.next();
			System.out.println(entry.getKey()+ " "+entry.getValue());
		}

		System.out.println("---------");
		TreeMap<String, Integer> treemap = new TreeMap<String, Integer>(capitalmapUnsorted);

		Iterator<String> it1 = treemap.keySet().iterator();

		while(it1.hasNext())
		{
			String key = it1.next();
			Integer value = treemap.get(key);
			System.out.println(key+" "+ value);
		}

	}
```

****

Que - What is local, static and Instance variable?

Ans - **Local Variabe** - Local variable is a variable declared inside the body of the method paramenter.

Syntax -

```
public void SetValue()
{
    int i;

}
```

Instance variable - The variable which is declared at class level (in class) but outside the methods are called instance variable

Syntax - the 'i' here is instance variable

```
public class Test {

   int i;

   public void show()

   {
       System.out.println(i);
   }

}
```

Static Variable - A variable which is declared with the help of static keyword is called static variable, this should be declared at the class level and not inside the method.
These variable generates only single copy while local and instance variable generates multiple copies number of time object is been made of the class

Syntax -

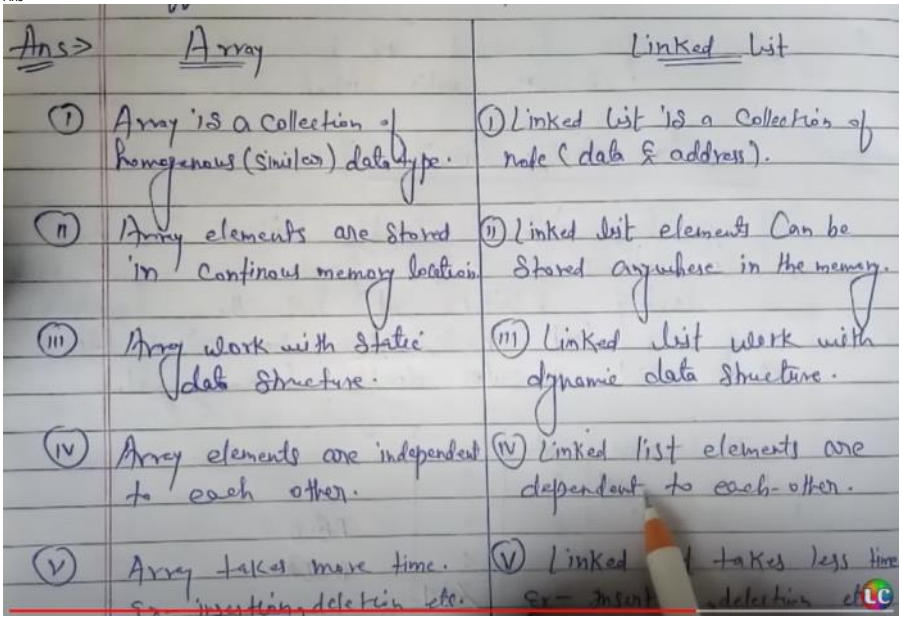
```
public class Test
{

    static int i;

}
```

Que - What is 'this' keyword ?
Ans - This keyword is used to assign the value of the instance variable of same name.

Que - What is super keyword?
Ans - 'Super' keyword is use to differentiate between the method and variable name if the name is same in the child class and super class.

Que - Can we overload and override the main method?
Ans -

Que - Difference between ArrayList and LinkedList?
Ans -



Que - What is the difference between Arrray and ArrayList?

Properties	Array	ArrayList
Static/ Dynamic	Array is **static** in size.	ArrayList is **dynamic** in size. It can be resized itself when needed.
Resizable	Array is a **fixed** length data structure	ArrayList is a **variable** length Collection class.
Primitive/ Generic type	An array can store both **objects and primitives type.**	We **cannot store primitive type** in ArrayList. It automatically converts primitive type to object.
Performance	It performs **fast** in comparison to ArrayList because of fixed size.	**resize() operation :** Automatic resize will **slow** down the performance as it will use temporary array to copy elements from the old array to new array.(by 50%) **add() or get() operation :** almost **same** performance , as for ArrayList object these operations run in constant time.

Que - Difference between ArrayList and LinkedList?
Ans -

Properties	ArrayList	LinkedList
Internal Implementation	ArrayList internally uses a **dynamic array** to store its elements.	LinkedList uses **Doubly Linked List** to store its elements.
Implementation	ArrayList implements only **List**.	LinkedList implements **List as well as Queue**. It can acts as a queue as well.
Manipulation / add / delete operation	Manipulation with ArrayList is **slow** because it internally uses an array. If any element is removed from the array, all the other elements are shifted in memory.	Manipulation with LinkedList is **faster** than ArrayList because it uses a doubly linked list, so no bit shifting is required in memory.
Access/ get operation	ArrayList is **faster** in storing and accessing data as internally Array is used which has random access.	LinkedList is **slower** than ArrayList in storing and accessing data as access requires node by node traversal.
Reverse Iterator	there is no descendingIterator() in ArrayList	LinkedList **can be iterated in reverse** direction using descendingIterator()

Properties	ArrayList	LinkedList
Initial Capacity	If the constructor is not overloaded , then ArrayList creates an empty list of initial **capacity 10**	There is no case of default capacity in a LinkedList. Hence an **empty list** is created when a LinkedList is initialized.
Memory Overhead	In ArrayList Memory overhead is **less** as each index only holds the actual object(data).	Memory overhead in LinkedList is **more** as compared to ArrayList as node in LinkedList needs to maintain the addresses of next and previous node.

Conclusion: LinkedList element deletion and addition is faster compared to ArrayList.

Reason: LinkedList's each element maintains two pointers (addresses) which points to the both neighbor elements in the list. Hence removal only requires change in the pointer location in the two neighbor nodes (elements) of the node which is going to be removed. While In ArrayList all the elements need to be shifted to fill out the space created by removed element.

Hence if there is a requirement of **frequent addition and deletion** in application then **LinkedList** is a best choice and if more **search operations** requirement, **ArrayList** would be your best bet.

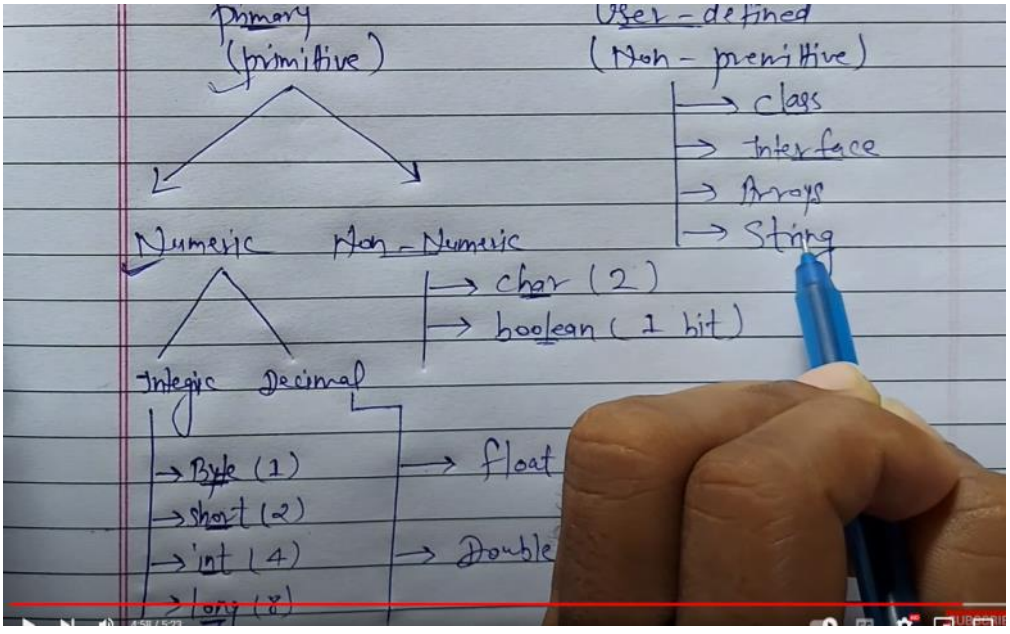
Que - Difference between List and Set?

Ans-

S.No.	List	Set
1	List is an index based data structure.	Set is not an index based data structure.It stores the data according to the hashcode values
2	List can store duplicate elements.	Set does not allow to store duplicate elements
3	List can store any number of null values.	Set can store only one null value
4	List follows the insertion order.	Set does not follows the insertion order.
5	We can iterate(get) the list element by Iterator & ListIterator	We can iterate the set elements by Iterator.

Que - What is Primitve and Non Primitive Data Types?

Ans -



Screen clipping taken: 25-08-2022 15:17

Rest_Assured

26 April 2022 00:38

1. How can we do assertion on Json Response Body and Headers?

Ans -

Interview Questions

22 July 2022 23:30

Please click on the respected links to see the questions:

Jenkins

TestNG

Selenium

Java

Java Logic Programme

String Programs

Functional Testing Questions

Jenkins

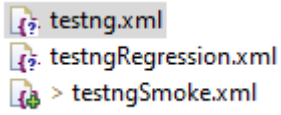
08 August 2022 18:21

TestNG

08 August 2022 18:21

Que - How can you rerun the failed test cases?
Ans -

Que - How can we run multiple TestNG files?
Ans -




```
<suite name="Suite">

    <suite-files>

        <suite-file path="testngRegression.xml"></suite-file>
        <suite-file path="testngSmoke.xml"></suite-file>

    </suite-files>

</suite> <!-- Suite -->
```

Selenium

08 August 2022 18:21

Que 1. Sorting in table in webtable?

Que 2. How you manage soft assertion to give result?

Que 3. Why we use constructor in Page Object Model in Selenium?
Ans - We use constructor to maintain the same instance of the driver throughout the project.

Que 4. How to rerun the failed test case with the help of TestNG?
Ans - There is one interface IRetryAnalyzer by using this interface we can achieve rerun of the failed test case

We have to create a class which implements IRetryAnalyzer and then need to define the method 'retry' which accepts result as a argument
Then in the @Test we have to pass the argument like below

```
@Test(retryAnalyzer =  RerunTheFailedTestCase.RetryAnalyzer.class)
	public void Test1()
	{
		Assert.assertEquals(false, true);
	}

public class RetryAnalyzer implements IRetryAnalyzer{

	int iCounter = 0;
	int iretryLimit = 3;


	@Override
	public boolean retry(ITestResult result) {
		if(iCounter<iretryLimit)
		{
			iCounter++;
			return true;
		}
		return false;
	}

}
```

Que 5. What is the key feature of Selenium 4?
Ans -

- Selenium 4.0 was announced in 2018.
- 1st stable version was in oct 2021.
- Selenium is now W3C compliant.
- Relative Locators
- Better Window/Tab Management
- Improved Selenium Grid
- Improved Selenium IDE
- New API's fpr CDP (Chrome DevTools Protocol)
- Depricated of Desired Capabilities
- Modifications in the Actions Class

Changes in Selenium WebDriver

- Selenium is now W3C (World Wide Web Consortium) compliant
- Previously selenium has a request response architecture via JSON Wire protocol, this was used to encode and decode the commun ication between selenium and browser
- JSON Wire Protocol is been removed now and communicaiton is happening directly.
- Any software following W3C standard protocol can be integerated with selenium with no compatibility issue.
- All the major browsers Chrome, IE and Safari are already W3C compliant so the communication is much faster now.

Relative Locators

- Functions to locate nearby elements by specifying directions
- New locator strategies like 'above', 'below', 'toLeftOf', 'toRightOf', near.

Better Window/Tab Management

- Work with multiple windows or tabs in the same session
- Can now open multiple windows/tabs without creating new driver object
- Open new window - driver.switchTo().newWindow(WindowType.WNDOW);
- Open new tab - driver.switchTo().newWindow(WindowType.TAB);

1. **What is Automation Testing?**
Automation testing is the process of testing the software using an automation tool to find the defects. In this process, exec uting the test scripts and generating the results are performed automatically by automation tools. Some most popular tools to do automation testing are HP QTP/UFT, Selenium WebDriver, etc.,

2. **What are the benefits of Automation Testing?**

- Saves time and money. Automation testing is faster in execution.
- Reusability of code. Create one time and execute multiple times with less or no maintenance.
- Easy reporting. It generates automatic reports after test execution.
- Easy for compatibility testing. It enables parallel execution in the combination of different OS and browser environments.
- Low-cost maintenance. It is cheaper compared to manual testing in a long run.
- Automated testing is more reliable.
- Automated testing is more powerful and versatile.
- It is mostly used for regression testing. Supports execution of repeated test cases.
- Minimal manual intervention. Test scripts can be run unattended.
- Maximum coverage. It helps to increase the test coverage.

3. **What type of tests have you automated?**
Our main focus is to automate test cases to do Regression testing, Smoke testing, and Sanity testing. Sometimes based on the project and the test time estimation, we do foc us on End to End testing.

4. **How many test cases you have automated per day?**
It depends on Test case scenario complexity and length. I did automate 2 -5 test scenarios per day when the complexity is limited. Sometimes just 1 or fewer test scenarios in a day when the complexit y is high.

5. What is a Framework?
A framework defines a set of rules or best practices which we can follow in a systematic way to achieve the desired results. There are different types of automation frameworks and the most common ones are:

- Data Driven Testing Framework
- Keyword Driven Testing Framework
- Hybrid Testing Framework

6. Have you created any Framework?
If you are a beginner: No, I didn't get a chance to create a framework. I have used the framework which is already available.
If you are an experienced tester: Yes, I have created a framework. Or I have involved in the creation of the framework.

7. Can you explain the Framework which you have used in your Selenium Project?
Here we have clearly explained each component of Framework.

Programming Language
Java
Type of Framework
Different Packages
Page Object
Resources - base class, common utilities, ExtentReporter, data.properties, Custom Listners
TestData
TestCases
CI/CD

8. Why do you prefer Selenium Automation Tool?

- Free and open source
- Have large user base and helping communities
- Cross browser compatibility
- Platform compatibility
- Multiple programming languages support

9. What is Selenium?
Selenium is an open source (free) automated testing suite to test web applications. It supports different platforms and brows ers. It has gained a lot of popularity in terms of web -based automated testing and giving a great competition to the famous commer cial tool HP QTP (Quick Test Professional) AKA HP UFT (Unified Functional Testing).
Selenium is a set of different software tools. Each tool has a different approach in supporting web based automation testing.
It has four components namely,
i Selenium IDE (Integrated Development Environment)
ii Selenium RC (Remote Control) – selenium 1
iii Selenium WebDriver – selenium 2 & 3
iv Selenium Grid

10. What is Selenium IDE?
Selenium IDE (Integrated Development Environment) is a Firefox plugin. It is the simplest framework in the Selenium Suite. It allows us to record and playback the scripts. Even though we can create scripts using Selenium IDE, we need to use Selenium R C or Selenium WebDriver to write more advanced and robust test cases.

11. What is Selenese?
Selenese is the language which is used to write test scripts in Selenium IDE.

12. Which is the only browser that supports Selenium IDE to be used?
Firefox

13. What is Selenium RC?
Selenium RC AKA Selenium 1. Selenium RC was the main Selenium project for a long time before the WebDriver merge brought up S elenium 2. Selenium 1 is still actively supported (in maintenance mode). It relies on JavaScript for automation. It supports Java, Javascript, Ruby, PHP, Python, Perl and C#. It supports almost every browser out there.

14. What is Selenium WebDriver?
Selenium WebDriver AKA Selenium 2 is a browser automation framework that accepts commands and sends them to a browser. It is implemented through a browser-specific driver. It controls the browser by directly communicating with it. Selenium WebDriver supports Java, C#, PHP, Python, Perl, Ruby.

15. What is Selenium Grid?
Selenium Grid is a tool used together with Selenium RC to run tests on different machines against different browsers in parallel. That is, running multiple tests at the same time against different machines running different browsers and operating systems.
In simple words, it is used to distribute your test execution on multiple platforms and environments concurrently.

16. When do you use Selenium Grid?
Selenium Grid can be used to execute same or different test scripts on multiple platforms and browsers concurrently so as to achieve distributed test execution

17. What are the advantages of Selenium Grid?
It allows running test cases in parallel thereby saving test execution time.
It allows multi-browser testing
It allows us to execute test cases on multi -platform

18. What is a hub in Selenium Grid?
A hub is a server or a central point that controls the test executions on different machines.

19. What is a node in Selenium Grid?
Node is the machine which is attached to the hub. There can be multiple nodes in Selenium Grid.

20. What are the types of WebDriver APIs available in Selenium?
- Firefox Driver
- Gecko Driver
- InternetExplorer Driver
- Chrome Driver
- HTMLUNIT Driver
- Opera Driver
- Safari Driver
- Android Driver
- iPhone Driver
- EventFiringWebDriver

21. Which WebDriver implementation claims to be the fastest?
The fastest implementation of WebDriver is the HTMLUnitDriver. It is because the HTMLUnitDriver does not execute tests in the browser.

22. What are the Programming Languages supported by Selenium WebDiver?
- Java
- C#
- Python
- Ruby
- Perl
- PHP

23. What are the Operating Systems supported by Selenium WebDriver?
- Windows
- Linux
- Apple

24. What are the Open-source Frameworks supported by Selenium WebDriver?
- JUnit
- TestNG
- CUCUMBER
- JBHEAVE

25. What are the Locators available in Selenium?
Different types of locators are:
1 ID –
2 ClassName –
3 Name –
4 TagName –
5 LinkText –
6 PartialLinkText –
7 XPath –
8 CSS Selector –

26. What is a XPath?
XPath is used to locate the elements. Using XPath, we could navigate through elements and attributes in an XML document to lo cate web elements such as textbox, button, checkbox, Image etc., in a web page.

27. What is the difference between "/" and "//"
Single Slash "/" – Single slash is used to create XPath with absolute path i.e. the XPath would be created to start selection from the document node/start node
Double Slash "//" – Double slash is used to create XPath with relative path i.e. the XPath would be created to start selection from anywhere with in the document.

28. What is the difference between Absolute Path and Relative Path?
Absolute XPath starts from the root node and ends with desired descendant element's node. It starts with top HTML node and ends with input node. It starts with a single forward slash(/) as shown below.
/html/body/div[3]/div[1]/form/table/tbody/tr[1]/td/input
Relative XPath starts from any node in between the HTML page to the current element's node(last node of the element). It starts with a single forward slash(//) as shown below.
//input[@id='email']

29. What is the difference between Assert and Verify in Selenium?

1 2 3 4 5	WebDriver driver = new FirefoxDriver(); instead of creating FirefoxDriver driver = new FirefoxDriver();	
1	driver.findElement(By.linkText("Software Testing Material Website")).click();	

1		
1		driver.get("http://www.softwaretestingmaterial.com");

3 1	4 boolean elePresent = driver.findElement(By.xpath("xpath")).isDisplayed();	
3 1	4 boolean eleSelected= driver.findElement(By.xpath("xpath")).isSelected();	

3 1	4 boolean eleEnabled= driver.findElement(By.xpath("xpath")).isEnabled();

Assert: In simple words, if the assert condition is true then the program control will execute the next test step but if the condition is false, the exec ution will stop and further test step will not be executed.
Verify: In simple words, there won't be any halt in the test execution even though the verify condition is true or false.
For detailed post check the below link.

30. What are Soft Assert and Hard Assert in Selenium?
Soft Assert: Soft Assert collects errors during *@Test* Soft Assert does not throw an exception when an assert fails and would continue with the next step after the assert statement
Hard Assert: Hard Assert throws an AssertException immediately when an assert statement fails and test suite continues with next *@Test*

Hard Assert

Assert.asserttrue("Mayur","Khare");
here it will throw error

Soft Assert
SoftAssert sftas = new SoftAssert();
sftas .assertrue("Mayur","Khare");

31. What are the verification points available in Selenium?
In Selenium IDE, we use Selenese Verify and Assert Commands as Verification points
In Selenium WebDriver, there is no built-in features for verification points. It totally depends on our coding style. some of the Verification points are
To check for page title
To check for certain text
To check for certain element (text box, button, drop down, etc.)

32. How to launch a browser using Selenium WebDriver?
WebDriver is an Interface. We create Object of a WebDriver Interface.
<2.53 – no geckodriver
3.x – geckodriver for FF
To launch Firefox Driver:WebDriver driver = new FirefoxDriver();
To launch Chrome Driver:WebDriver driver = new ChromeDriver();
To launch Internet Explorer Driver:WebDriver driver = new InternetExplorerDriver();

33. Is the FirefoxDriver a Class or an Interface?
FirefoxDriver is a Java class, and it implements the WebDriver interface.

34. What is the super interface of WebDriver?
SearchContext.

35. Explain the line of code Webdriver driver = new FirefoxDriver(); ?
'WebDriver' is an interface and we are creating an object, reference of type WebDriver instantiating an object of FirefoxDriv er class.

36. Why we do create a reference variable 'driver' of type WebDriver
WebDriver driver = new FirefoxDriver();
instead of creating
FirefoxDriver driver = new FirefoxDriver();
What is the purpose of doing this way?
if we create a reference variable driver of type WebDriver then we could use the same driver variable to work with any browse r of our choice such as IEDriver, SafariDriver etc.,
//FirefoxDriver driver = new FirefoxDriver();
ChromeDriver driver = new ChromeDriver();
driver.get("http://www.google.com");
WebDriver driver = new FirefoxDriver();

System.setProperty("webdriver.chrome.driver","D:\\ChromeDriver\\chromedriver.exe");

From <https://www.browserstack.com/guide/setproperty-in-selenium>

37. What are the different exceptions you have faced in Selenium WebDriver?

N - 4
E - 2
S - 1
T - 1
I - 1

- NoSchSessionException - this exception occure when we try to perform any step just after closing the browser
- NoSuchWindowException - this exception is thrown when any window is not present or didn't get open in which the element was present to perform any op eration.
- TimeoutException - It took too long to launch any website then It gives this exception
- NoAlertPresentException - this exception is when you expecting an alert but alert doesn't appear and we have not handled theat exception then system wi ll throw this exception
  Alert alert = driver.switchTo().alert();
  alert.dismiss()
- NoSuchElementException - this exception is when we want to click on an element but that element is not present in the screen
- IllegalStateException - this exception occurs when your system property is not set properly (webdriver.chrome.driver) and in this statement we write capital 'W' instead of small w then it will this error
- ElementClickInterceptedException - any button is not clickable because of any other element which comes in front of that button for few seconds like loading sig n or something.
- StaleElementReference - This exception occurred when I selected any value from the dropdown and whole page got refereshed and the button which I want to click next is no longer in the dom and new reference for that element is been created.
- ElementNotInteractableException - exception which is thrown to indicate that although an element is present on the HTML DOM, it is not in a state that can be i nteracted with
  (for ex:- there is a button but it is hidden, this is possible when you pass the attribute 'Hidden' in a HTML code, so button is therein DOM but on screen it is not present or hidden)

37 a.) How to handle WebDriverTimOut Exception..?
Ans - we can use so many thing by which we can handle this exception
1. Implicit Wait
2. Explicit Wait
3. Fluent Wait
4. PageLoadTimeOut Exception - driver.manage().timeOut().pageLoadTimeOut(30, TimeUnit.SECONDS)
5. Thread.sleep

38. How To Login Into Any Site If It Is Showing Any Authentication Pop-Up For Username And Password?
To do this we pass username and password with the URL
http://username:password@url
e.g. http://admin:admin123@xyz.com

39. What are the types of waits available in Selenium WebDriver?
In Selenium we could see three types of waits such as Implicit Waits, Explicit Waits and Fluent Waits.
- Implicit Waits – driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
- Explicit Waits – By using the Explicit Wait command, the WebDriver is directed to wait until a certain condition occurs before proceeding with executing the code.
- Fluent Waits –
- PageLoadTimeOut - driver.manage().timeouts().pageLoadTimeout(100, SECONDS);
- Thread.sleep() – static wait

Can you write a code for Explicit wait.?
WebDriverWait wait = new WebDriverWait(driver,30);
wait.until(ExpectedConditions.elementToBeClickable(By.xpath("//div[contains(text(),'COMPOSE')]")));
driver.findElement(By.xpath("//div[contains(text(),'COMPOSE')]")).click();

After selenium 4 - WebDriverWait wait = new WebDriverWait(driver,Duration.ofSeconds(10));

40. What is Implicit Wait In Selenium WebDriver?
Implicit waits tell to the WebDriver to wait for a certain amount of time before it throws an exception. Once we set the time , WebDriver will wait for the element based on the time we set before it throws an exception. The default setting is 0 (zero). We need to set some wait time to make WebDriver to wait for the required time.

41. What is WebDriver Wait In Selenium WebDriver?
WebDriverWait is applied on a certain element with defined expected condition and time. This wait is only applied to the specified element. This wait can also throw an exception when an element is not found.

42. What is Fluent Wait In Selenium WebDriver?
FluentWait can define the maximum amount of time to wait for a specific condition and frequency with which to check the condition before throwing an "ElementNotVisibleException" exception.

43. How to input text in the text box using Selenium WebDriver?
By using sendKeys() method
WebDriver driver = new FirefoxDriver();
driver.get("https://www.gmail.com");
driver.findElement(By.xpath("xpath")).sendKeys("test");

44. How to input text in the text box without calling the sendKeys()?
// To initialize js object
JavascriptExecutor JS = (JavascriptExecutor)driver;
// To enter value in search box
JS.executeScript("document.getElementById('twotabsearchtextbox').value='Selenium'");

45. How to clear the text in the text box using Selenium WebDriver?
By using clear() method
WebDriver driver = new FirefoxDriver();

driver.get("https://www.gmail.com");
driver.findElement(By.xpath("xpath_of_element1")).sendKeys("Software Testing Material Website");
driver.findElement(By.xpath("xpath_of_element1")).clear();

46. How to get a text of a web element?
By using getText() method

```
driver.findElement(By.xpath("xpath_of_element1")).getText();
```

47. How to get an attribute value using Selenium WebDriver?
By using getAttribute(value);

```
driver.findElement(By.xpath("xpath_of_element1")).getAttribute("id");
```

48. How to click on a hyperlink using Selenium WebDriver?
We use click() method in Selenium to click on the hyperlink
driver.findElement(By.linkText("Software Testing Material Website")).click();

49. How to submit a form using Selenium WebDriver?
We use "submit" method on element to submit a form
driver.findElement(By.id("form_1")).submit();
Alternatively, you can use click method on the element which does form submission

50. How to press ENTER key on text box In Selenium WebDriver?
To press ENTER key using Selenium WebDriver, We need to use Selenium Enum Keys with its constant ENTER.
driver.findElement(By.xpath("xpath")).sendKeys(Keys.ENTER);

51. How to pause a test execution for 5 seconds at a specific point?
By using java.lang.Thread.sleep(long milliseconds) method we could pause the execution for a specific time. To pause 5 seconds, we need to pass parameter as 5000 (5 seconds)
Thread.sleep(5000)

52. Is Selenium Server needed to run Selenium WebDriver Scripts?
When we are distributing our Selenium WebDriver scripts to execute using Selenium Grid, we need to use Selenium Server.

53. What happens if I run this command. driver.get("www.softwaretestingmaterial.com") ;
An exception is thrown. We need to pass HTTP protocol within driver.get() method.
driver.get("http://www.softwaretestingmaterial.com");

54. What is the alternative to driver.get() method to open an URL using Selenium WebDriver?
Alternative method to driver.get("url") method is driver.navigate.to("url")

55. What is the difference between driver.get() and driver.navigate.to("url")?
driver.get(): To open an URL and it will wait till the whole page gets loaded
driver.navigate.get(): To navigate to an URL and It will not wait till the whole page gets loaded

56. Can I navigate back and forth in a browser in Selenium WebDriver?
We use Navigate interface to do navigate back and forth in a browser. It has methods to move back, forward as well as to refresh a page.
driver.navigate().forward(); – to navigate to the next web page with reference to the browser's history
driver.navigate().back(); – takes back to the previous webpage with reference to the browser's history
driver.navigate().refresh(); – to refresh the current web page thereby reloading all the web elements
driver.navigate().to("url"); – to launch a new web browser window and navigate to the specified URL

57. What are the different types of navigation commands?
Refer above question (Can I navigate back and forth in a browser)

58. How to fetch the current page URL in Selenium?
To fetch the current page URL, we use getCurrentURL()
driver.getCurrentUrl();

59. How can we maximize browser window in Selenium?
To maximize browser window in selenium we use maximize() method. This method maximizes the current window if it is not already maximized
driver.manage().window().maximize();

60. How to delete cookies in Selenium?
To delete cookies we use deleteAllCookies() method
driver.manage().deleteAllCookies();

61. What are the ways to refresh a browser using Selenium WebDriver?
There are multiple ways to refresh a page in selenium
- Using driver.navigate().refresh() command as mentioned in the question 45
- Using driver.get("URL") on the current URL or using driver.getCurrentUrl()
- Using driver.navigate().to("URL") on the current URL or driver.navigate().to(driver.getCurrentUrl());
- Using sendKeys(Keys.F5) on any textbox on the webpage

62. What is the difference between driver.getWindowHandle() and driver.getWindowHandles() in Selenium WebDriver?
driver.getWindowHandle() It r– eturns a handle of the current page (a unique identifier)
driver.getWindowHandles() – It returns a set of handles of the all the pages available.

63. What is the difference between driver.close() and driver.quit() methods?
Purpose of these two methods (driver.close and driver.quit) is almost same. Both allow us to close a browser but still, there is a difference.
driver.close(): To close current WebDriver instance
driver.quit(): To close all the opened WebDriver instances

64. What is the difference between driver.findElement() and driver.findElements() commands?
The difference between driver.findElement() and driver.findElements() commands is -
- findElement() returns a single WebElement (found first) based on the locator passed as parameter. Whereas findElements() retu rns a list of WebElements, all satisfying the locator value passed.
- Syntax of findElement()-
- WebElement textbox = driver.findElement(By.id("textBoxLocator"));
- Syntax of findElements()-
- List <WebElement> elements = driver.findElements(By.id("value"));
- Another difference between the two is- if no element is found then findElement() throws NoSuchElementException whereas findElements() returns a list of 0 elements.

List<WebElement> list = driver.findElements(By.tagName("a"));
Sop(list.size()); ==40

65. How to find whether an element is displayed on the web page?
WebDriver facilitates the user with the following methods to check the visibility of the web elements. These web elements can be buttons, drop boxes, checkboxes, radio buttons, labels etc.
1 isDisplayed()
2 boolean elePresent = driver.findElement(By.xpath("xpath")).isDisplayed();
1 isSelected()
2 boolean eleSelected= driver.findElement(By.xpath("xpath")).isSelected();
1 isEnabled()
2 boolean eleEnabled= driver.findElement(By.xpath("xpath")).isEnabled();

66. How to select a value in a dropdown?
By using Select class
WebElement mySelectElement = driver.findElement(By.name("dropdown"));

Select dropdown = new Select(mySelectElement);
dropdown.selectByVisibleText(Text);
dropdown.selectByIndex(Index);
dropdown.selectByValue(Value);

67. How to capture Screenshot in Selenium WebDriver?

```
//Convert web driver object to TakeScreenshot

TakesScreenshot scrShot =(TakesScreenshot)driver;

//Call getScreenshotAs method to create image file

File SrcFile=scrShot.getScreenshotAs(OutputType.FILE);

//Move image file to new destination
File DestFile=new File(fileWithPath);

//Copy file at destination

FileUtils.copyFile(SrcFile, DestFile);
```

68. How to mouse hover on a web element using WebDriver?
By using Actions class
```
WebElement ele = driver.findElement(By.xpath("xpath"));
//Create object 'action' of an Actions class
Actions action = new Actions(driver);
//Mouseover on an element
action.moveToElement(ele).build().perform();
```

69. How to handle Alerts/Pop-ups in Selenium?
To make object of Alert -
```
Alert alert = driver.switchTo().alert();
```

To dismiss - driver.switchTo().alert().dismiss();
To accept - driver.switchTo().alert().accept();
To get text - driver.switchTo().alert().getText();
To send data to alert - driver.switchTo().alert().sendKeys("Text");

70. How can we handle windows based pop up?
Selenium doesn't support windows based applications. It is an automation testing tool which supports only web application tes ting. We could handle windows based popups in Selenium using some third party tools such as AutoIT, SIKULI, Robot class etc.

71. How to handle hidden elements in Selenium WebDriver?
It is one of the most important selenium interview questions.
We can handle hidden elements by using javaScript executor
```
(JavascriptExecutor(driver)).executeScript("document.getElementsByClassName(ElementLocator).click();");
```

72. How can you find Broken Links in a page using Selenium WebDriver?

73. How to find more than one web element in the list?
```
// To store the list
List <WebElement> eleList = driver.findElements(By.xpath("xpath"));
// To fetch the size of the list
int listSize = eleList.size();
//for loop
for (int i=0; i<listSize; i++)
{
        // Clicking on each link
        links.get(i).click();
        // Navigating back to the previous page that stores the links
        driver.navigate().back();
}
```

74. How to read a JavaScript variable in Selenium WebDriver?
By using JavascriptExecutor
```
// To initialize the JS object.
JavascriptExecutor JS = (JavascriptExecutor)driver;
// To get the site title.
String title = (String)JS.executeScript("return document.title");
System.out.println("Title of the webpage : " + title);
```

75. How do you read test data from excels?
Test data can efficiently be read from excel using JXL or POI API. POI API has many advantages than JXL.

76. Is it possible to automate the captcha using Selenium?
No, It's not possible to automate captcha and bar code reader.

77. List some scenarios which we cannot automate using Selenium WebDriver?
1. Bitmap comparison Is not possible using Selenium WebDriver
2. Automating Captcha is not possible using Selenium WebDriver
3. We cannot read bar code using Selenium WebDriver
4. windows OS based pop ups
5. third party calendars/element
6. Image
7. Word/PDF

78. What is Object Repository in Selenium WebDriver?
Object Repository is used to store element locator values in a centralized location instead of hard coding them within the sc ripts. We do create a property file (.properties) to store all the element locators and these property files act as an object repository in Selenium WebDriver.

79. How can you use the Recovery Scenario in Selenium WebDriver?
By using "Try Catch Block" within Selenium WebDriver Java tests.
```
try {
    driver.get("www.xyz.com");
}catch(Exception e){
    System.out.println(e.getMessage());
}
```

80. How to Upload a file in Selenium WebDriver?
There are two cases which are majorly used to upload a file in Selenium WebDriver such as using SendKeys Method and using AutoIT Script.
Browser Button – type ="file"
SendKeys (c:\test\naveen.jpg);

81. How to Download a file in Selenium WebDriver?
By using AutoIT script, we could download a file in Selenium WebDriver.

82. How to run Selenium WebDriver Test from the command line?
```
Class A{
}
cd c
c: javac A.java
c: java A.java
java org.testng.TestNG C:Users Desktop workspacetestingtestng.xml
```

83. How to switch between frames in Selenium?
By using the following code, we could switch between frames.
```
driver.switchTo().frame();
```

84. How to connect a Database in selenium?
As we all know Selenium WebDriver is a tool to automate User Interface. We could only interact with Browser using Selenium We bDriver.
We use JDBC Driver to connect the Database in Selenium (While using Java Programming Language).

85. How To Resize Browser Window Using Selenium WebDriver?
To resize the browser window to particular dimensions, we use 'Dimension' class to resize the browser window.
```
    //Create object of Dimen.sions class
    Dimension d = new Dimension(480,620);
    //Resize the current window to the given dimension
    driver.manage().window().setSize(d);
```

86. How To Scroll Web Page Down Or UP Using Selenium WebDriver?
JavaScript scrollBy() method scrolls the document by the specified number of pixels.

87. How To Perform Right Click Action (Context Click) In Selenium WebDriver?
We use Actions class in Selenium WebDriver to do Right -Click (Context Click) action.
```
    action.contextClick(driver.findElement(By.xpsjht()).build().perform();
```

88. How To Perform Double Click Action In Selenium WebDriver?
We use Actions class to do Double click action in selenium.

89. How To Perform Drag And Drop Action in Selenium WebDriver?
We use Actions class to do Drag And Drop Action

90. How To Highlight Element Using Selenium WebDriver?
By using JavascriptExecutor interface, we could highlight the specified element

91.How to write a code to read any data from excel file?
```
File src=new File("C:\\Projects\\AutomationFiles\\Results.xlsx");

FileInputStream fis=new FileInputStream(src);

XSSFWorkbook wb = new XSSFWorkbook(fis);

XSSFSheet sheet1 = wb.getSheetAt(0);

String data0=sheet1.getRow(0).getCell(0).getStringCellValue();

System.out.println(data0);
```

Advanceautoparts

Que. What is the return type of WindowHandler?

Que. Syntax of Action class..?

```
Actions action = new Actions(driver);

action.sendKeys(Keys.ESCAPE).perform();
```

Que. Xpath writing for Parent,sibling, ancestor?

```
//Xpath by using starts-with
driver.findElement(By.xpath("//div[starts-with(@data-testid,'round-tri')]"));


//Xpath by using ends-with
driver.findElement(By.xpath("//div[ends-with(@data-testid,'round-tri')]"));


// Xpath by using Parent
driver.findElement(By.xpath("//div[contains(@data-testid,'round-trip-radio-button'//parent::div")).click();



// Xpath by using preceding-sibling
// this gives the node before(above) of current node
driver.findElement(By.xpath("div[contains(text(),'Family & Friends')]//parent::div[@class='css-1dbjc4n']//preceding-sibling::div"));



// Xpath by using following-sibling
// this gives the node after(below) of current node
driver.findElement(By.xpath("div[contains(text(),'Family & Friends')]//parent::div[@class='css-1dbjc4n']//following-sibling::div"));
```

Que - How to write code of Extent Report?

```
String path = "C:/Temp";
ExtentSparkReporter reporter = new ExtentReporter(path);
reporter.config().setReportName("Web Automation Result");
reporter.config().setDocumentTitle("Test Result");

ExtentReports extent = new ExtentReports();
extent.attachReporter(reporter);
extent.setSystemInfo("Tester", "Mayur Khare");

Now calling that method
ExtentReports extent = ExtentReporterNG.getReportObject();
test=extent.createTest(result.getMethod().getMethodName());
```

Que - How to handle SSL Certificate?

```
ChromeOptions options = new ChromeOptions();
Options.setAcceptInsecureCerts(false);
```

Que - How will you sort in hashmap?

Que - Difference between toString and string.Valueof?

Que - How will you fetch numeric values from the excel using apache poi?

Que - Primitive and Non Primitive Data Type?
Ans - Primitve data type are predefined data types in java like int,float,boolean,char
Non Primitive data types are defined by programmers.

Non-primitive types can be used to call methods to perform certain operations, while primitive types cannot.

A primitive type has always a value, while non-primitive types can be null.
That is why we cannot have int i="", it will give error because it is primitve data type and String s = ""/null is fine.

Java

08 August 2022 18:22

Que - What is the difference between Abstract Class and Interface?
Ans -

Que - Difference between check and unchecked exception?
Ans-

Que - How you handle exceptions in java?
Ans-

Que - Can we write try block without catch and finally?
Ans -

Que - What is finally block?
Ans -

Que - What is the difference between classnotfound and class not def?
Ans-

Que - Can we throw exception deliberatley
Ans

Que - Is there any condition if we can ignore finally block?
Ans

Que - Automate the calendar of make my trip?
Ans

Que - Difference between Exception and Errors?
Ans - Exceptions are occur mainly due to our programs and Error are something which occur due to lack of system resources like les s RAM, slow processor, so programmer cannot do anything.

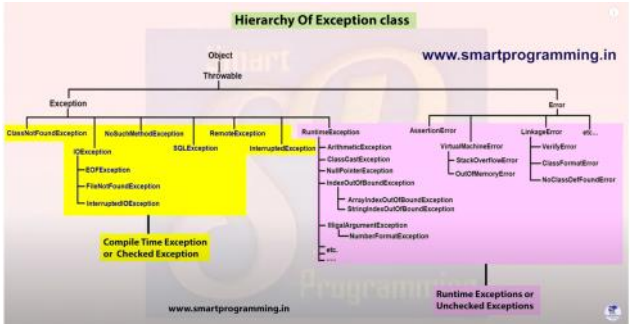
Exceptions can be recoverable(means we can handle them from exception handling) and error is not recoverable(Programmer canno t handle the error, for this programmer need to contact IT person)

Exceptions are two type "Compile time exception"/Checked exception and "Runtime exception/Unchecked exception
Errors are only "Runtime exception/Unchecked exception

CompileTime Exception - Compile time exception are those exception which Java can catch at compile time that this exception can occur in future. (ex ception didn't occur at compile time this was just a warning that this exception can come in future)

RunTime Exception - Runtime exception are those exception which Java cannot catch at compile time and they occur at runtime.

Heriarchy of exception class



Que - How we can handle exception…?
Ans - Exception can be handled by 5 keywords
try, catch, final, throw, throws

Que - How to do exception testing in testng?

```
import org.testng.annotations.Test;

public class ExceptionTest
{
    @Test(expectedExceptions={ArithmeticException.class})
    public void exceptionTesting()
    {
        int i = 1/0;
        System.out.println("Value of i = "+i);
    }
}
```

Que - Can we use finally block with try?
Ans - yes, but exceptional handling will not be done.

Que - Can we use finally block with try and without catch?
Ans - yes, but exceptional handling will not be done.

Que - What is finally keyword?
Ans - finally will be executed every time either exception occurs or not.

Que - Can we use finally block without try and catch alone
Ans - No

****

Note-

1. We can use multiple catch blocks with one try block but we can only use single finally block with one try block, not multiple .
2. The statements present in finally block executes even if the try block contains control transfer statements (i.e. jump statem ents) like return, break or continue.
3. There are 4 possibilities where finally block will not execute after try block those are given below

a. Using of system.exit() method.
b. Causing a fatal error that causes the process to abort.
c. Due to an exception arising in finally block itself.
d. The death of the thread.

****

Que - Can we write alone try, catch and finally?
Ans - No

Que - Can we use multiple catch block with one try block?
Ans - yes, but if first catch block contains the parent exception class then we cannot write the second catch block which contains its child class.

```
try
{
    //any code
}
catch( Exception e)
{
    e.stacktrace
```

}
catch (ArithmeticException e)
{

}

This code will not work as its parent class exception is already present in the first catch block so we cannot write a catch block which contains its child exception class (Exception is a parent class of all exception classes)

Que - Can we write nested try-catch block (for ex- can we write try under any try block)?
Ans - yes we can write nested try catch block but for every try block there should be catch block.

Que - Can we write try block under catch block?
Ans - yes (but every try should have its catch block)

Que - Can we write try catch block under finally block?
Ans yes we can write try catch block in a finally block.

Que - Can this sequence is correct (try, finally and then catch)?
Ans - No.

Que - Can we write any statement (any line of code) in between try and catch?
Ans - No

Que - Is this sequence correct try finally then try catch?
Ans yes

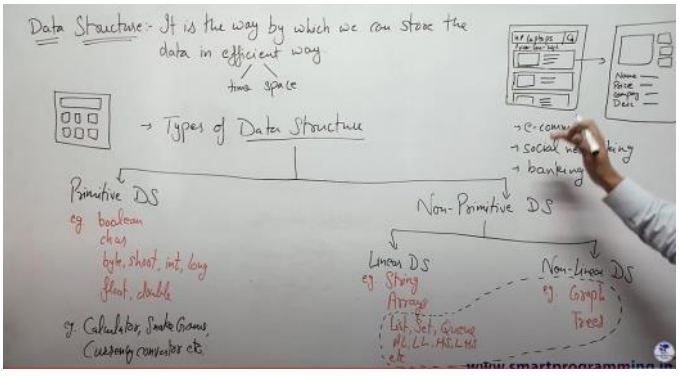
Que - What is the syntax of throw keyword?
Ans - throw new ExceptionclassName(<----any string/message---->);

Que - Difference between throws and throw keyword?
Ans -

Sr. no.	Basis of Differences	throw	throws
1.	Definition	Java throw keyword is used throw an exception explicitly in the code, inside the function or the block of code.	Java throws keyword is used in the method signature to declare an exception which might be thrown by the function while the execution of the code.
2.	Type of exception Using throw keyword, we can only propagate unchecked exception i.e., the checked exception cannot be propagated using throw only.	Using throws keyword, we can declare both checked and unchecked exceptions. However, the throws keyword can be used to propagate checked exceptions only.	
3.	Syntax	The throw keyword is followed by an instance of Exception to be thrown.	The throws keyword is followed by class names of Exceptions to be thrown.
4.	Declaration	throw is used within the method.	throws is used with the method signature.
5.	Internal implementation	We are allowed to throw only one exception at a time i.e. we cannot throw multiple exceptions.	We can declare multiple exceptions using throws keyword that can be thrown by the method. For example, main() throws IOException, SQLException.

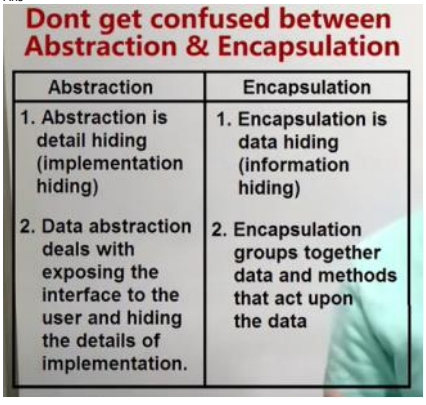
Que : What is the use of throws keyword?
Ans - throws keyword is used to declare the exception i.e. it indicates the caller method that given exception can occur so we have
To handle it with using try and catch block or again declare it by using throws keyword.

Que - What is Data Structure?



Que - What is Abstraction?
Ans - Abstraction is hiding internal implementation & just highlighting the setup services that we are offering.

Que - Is Abstraction & Encapsulation is same.?
Ans



Que - How we can achieve Abstraction?
Ans - There are two ways to achieve abstraction

1. By using Abstract Class (100% or 0% abstraction can be achieved)
2. By Interfaces (100% abstraction can be achieved)

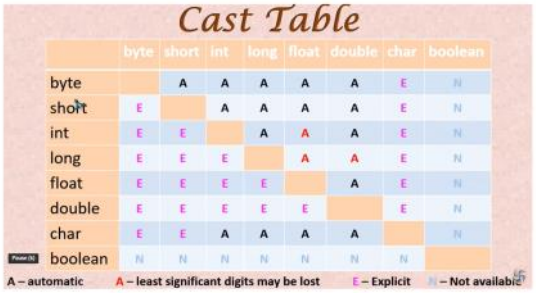
a. A method without a body (no implementation) is known as an abstract method
b. A method must always be declared in an abstract class, or we can say that if a class has an abstract method, class should be declared abstract as well.
c. If a class is mentioned as an abstract class then it's not mandatory that the method should also be an abstract method, non -abstract method will also work.
d. If a regular class extends an abstract class, then the class must have to implement all the abstract methods of the abstract parent class or it has to be declared abstract as well.
e. Abstract classes cannot be instantiated, which means we can't create an object of Abstract class.

Que - What is Selenium Webdriver?
Ans -

1. Selenium webdriver is the main component of selenium.

2. Selenium web driver is nothing but a collection of API's whose purpose is to automate web applications.

Que - Conversion chart one data type to another?



****

HashMap

Que -1 What is HashMap?
Ans - HashMap is an associative array data structure, Stores data in the form of key -value pair

- HashMap works on Hashing

Ans - So, suppose as we can see to use hashmap, we use put method as can be seen in Diagram 1,

- Java uses special method which is coming from object class and called Hash code method.
- // hashing - hashcode() -> hashing

Now as put method gets call, we are sending key and value pair (Naveen,100), so now as it will go inside the put method, now first hash (int hash, Diagram 2) will get calculated with the help of hashcode method

Que -2 How HashMap actually works?

Hash code will be calculated for only the key from hashcode method (hashCode(key), Diagram 2) not for the key value pair.
Now for Naveen one hashcode is generated suppose it's been calculated like [hash("Naveen")= 210678], so now one index will be calculated for this code say 'index 4' [int index = hash & (n -1), Diagram 2], So the [Key, value, HashCode, Next(this will be null)] these all four will get store in index 4

So now for Tom suppose hashcode generated like [hash("Tom")=210780], so now one index will be calculated for this code say 'index 9' [int index = hash & (n -1), Diagram 2], So the [Key, value, HashCode, Next] these all four will get store in index 9.

So now for Lisa suppose hashcode generated like [hash("Naveen")=210678], this is possible that for multiple key same hash code generated like here Lisa and Naveen has same hashcode so now one index will be calculated for this code and that will be obviously 'index 4' [int index = hash & (n -1), Diagram 2], So the [Key, value, HashCode, Next] these all four will get store in index 4, but now these 4 will get store in the 'Null' section of Naveen node, So now this will be called as Linked List and also when we get same index for multiple key then this condition called 'HashMap Collision'.

Diagram 1

```
Map<String, Integer> marks = new
HashMap<String, Integer>();

        marks.put("Naveen", 100);
        marks.put("Tom", 200);
        marks.put("Lisa", 300);
        marks.put("Peter", 400);
        marks.put("Robby", 600);
```

Diagram 2

```
public V put(K key, V value){
    int hash = hashCode(key);
    int index = hash & (n-1);
n = 16 (map size)
}
```

Que -3 What is the requirement for an object to be used as key or value in HashMap?
Ans - Both k and v should implement hashcode() and equals() method.

Que -4 What will happen if we try to store a key which is already present in HashMap?
Ans - Nothing will happen, just it will update the value of the key with the new value of that key, it will not throw any exception.
For ex - marks.put("Mayur",100) and now if we add marks.put("Mayur",400), it will replace the 100 and will store the 400 value, so when you will try to fetch the value of Mayur with the help of get method then 400 will be the outcome.

Que -5 Can we store a null key in Java HashMap?
Ans - Yes, we can store only one Null key in HashMap.

Que -6 Can we store a null value in Java HashMap?
Ans - Yes we can store multiple null values but only one null key.

Que - 7How does HashMap handle collisions in Java?
Ans - This is explained in above question, marked in yellow.

Que -8 Which Data Structure HashMap represents?
Ans - HasHMap represent HasTable Data Structure by using LinkedList and BalancedBinaryTree.

Que -9 Can we store duplicate key in HashMap?
Ans - No.

Que -10 Can we store duplicate value in HashMap?
Ans - Yes but with different key.

Que - 11 Is HashMap thread-safe in Java?
Ans - No this is not thread-safe or not synchronized.

Que - 12 What will happen if you use HashMap in a multithreaded Java application?
Ans - If we use HashMap in a multithreaded java application then hashmap can get corrupted because of working of different object on same map.

Que - 13 What are the different ways to iterate over HashMap in Java?
Ans - By using keyset and iterator
By using entrySet and iterator
By using entrySet and enhanced for Loop
By using keySet and get() method
By using forEach with lambda,
Can see the highlighted below in yellow

Que - 14 How do you remove a mapping while iterating over HashMap in Java?
Ans - use remove method

Que 15 - In which order mappings are stored in HashMap?
Ans - Random order because HashMap doesn't provide any ordering guarantee for keys, values or entries. When you iterate over a HashMap, you may get a different order every time you iterate over it.

Que- 16 Can you sort HashMap in Java?
Ans - No, we cannot sort HashMap in java with the help of key and value, but we can convert this to LinkedHashMap which maintains the insertion order(Sorting)

Que 17- What is the load factor in HashMap?
Ans - A load factor is a number that controls the resizing of HashMap when a number of elements in the HashMap cross the load factor as if the load factor is 0.75 and when becoming more than 75% full then resizing trigger which involves array copy.

Que 18 - How does resizing happens in HashMap?
Ans - same ans above

Que 19 - How many entries you can store in HashMap? What is the maximum limit?
Ans - Until run you out of the bucket, run out of the memory, no limit.

Que - 20 What is the difference between the capacity and size of HashMap in Java?
Ans - The capacity denotes how many entries HashMap can store, and size denotes how many mappings or key/value pair is currently present.

Que 21 - What will happen if two different keys of HashMap return the same hashcode()?
Ans - if same hashcode appears then collision will occur and they will maintain Linkedlist internally and after jdk 1.8, if threshold value reached to 8 then from 9th node onward it will get converted into Balanced BinaryTree

Iteration in HashMap

```
public static void main(String[] args) {

		HashMap<String, String> capitalmap = new HashMap<String, String>();
		capitalmap.put("India","Delhi");
		capitalmap.put("USA","Washington");
		capitalmap.put("UK","London");
		capitalmap.put("Spain","Madrid");

		System.out.println("********************Iterator1*************************");

		//iterator - 1 : over the keys: by using keySet()

		Iterator<String> it = capitalmap.keySet().iterator();

		while(it.hasNext())
		{
			String key = it.next();// to get the key
			String value = capitalmap.get(key);  //to get the value
			System.out.println(key+" "+value);

		}

		System.out.println("");
		System.out.println("********************Iterator2*************************");

		//iterator - 2 : over the set (pair): by using entrySet

		Iterator<Entry<String, String>> it1 = capitalmap.entrySet().iterator();

		while(it1.hasNext())
		{
			Entry<String, String> keyValue = it1.next();// to get the keyvalue pair
			System.out.println(keyValue.getKey()+ " "+keyValue.getValue());


		}

		System.out.println("");
		System.out.println("********************Iterator3*************************");

		//iterator - 3 : using java 8 for each and lambda

		capitalmap.forEach((k,v)->System.out.println(k+" "+v));

	}
```

Que - How to sort HashMap?

```
public static void main(String[] args) {


		Map<String, Integer> capitalmapUnsorted = new HashMap<String, Integer>();
		capitalmapUnsorted.put("India",100);
		capitalmapUnsorted.put("USA",100);
		capitalmapUnsorted.put("UK",100);
		capitalmapUnsorted.put("Spain",100);

		Iterator<Entry<String, Integer>> it = capitalmapUnsorted.entrySet().iterator();

		while(it.hasNext())
		{
			Entry<String, Integer> entry = it.next();
			System.out.println(entry.getKey()+ " "+entry.getValue());
		}

		System.out.println("----------");
		TreeMap<String, Integer> treemap = new TreeMap<String, Integer>(capitalmapUnsorted);

		Iterator<String> it1 = treemap.keySet().iterator();

		while(it1.hasNext())
		{
			String key = it1.next();
			Integer value = treemap.get(key);
			System.out.println(key+" "+ value);
		}

	}
```

****

Que - What is local, static and Instance variable?

Ans - **Local Variabe** - Local variable is a variable declared inside the body of the method paramenter.

Syntax -

```
public void SetValue()
{
    int i;

}
```

Instance variable - The variable which is declared at class level (in class) but outside the methods are called instance variable

Syntax - the 'i' here is instance variable

```
public class Test {

   int i;

   public void show()

   {
       System.out.println(i);
   }

}
```

Static Variable - A variable which is declared with the help of static keyword is called static variable, this should be declared at the class level and not inside the method.
These variable generates only single copy while local and instance variable generates multiple copies number of time object is been made of the class

Syntax -

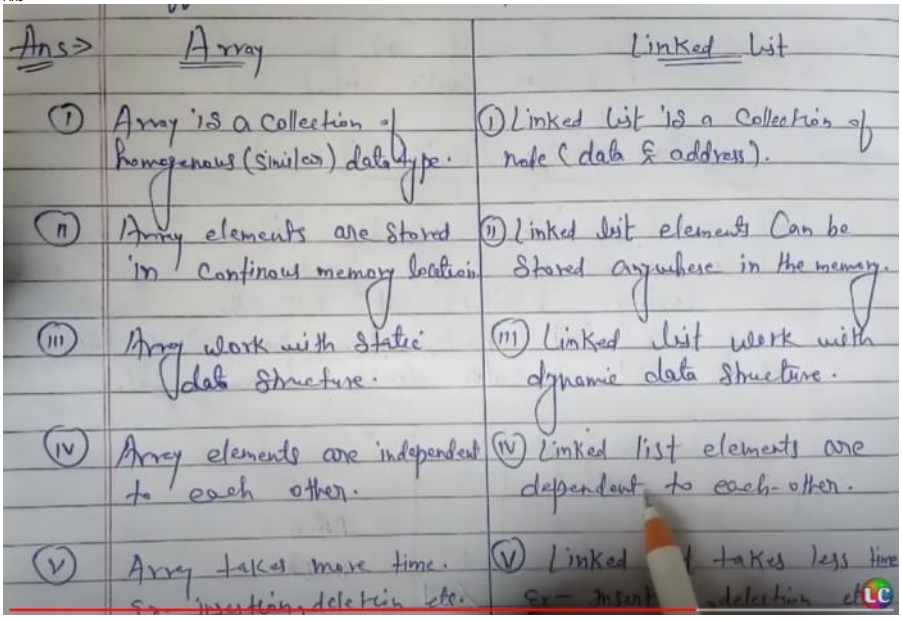
```
public class Test
{

    static int i;

}
```

Que - What is 'this' keyword ?
Ans - This keyword is used to assign the value of the instance variable of same name.

Que - What is super keyword?
Ans - 'Super' keyword is use to differentiate between the method and variable name if the name is same in the child class and super class.

Que - Can we overload and override the main method?
Ans -

Que - Difference between ArrayList and LinkedList?
Ans -



Que - What is the difference between Arrray and ArrayList?

Properties	Array	ArrayList
Static/ Dynamic	Array is **static** in size.	ArrayList is **dynamic** in size. It can be resized itself when needed.
Resizable	Array is a **fixed** length data structure	ArrayList is a **variable** length Collection class.
Primitive/ Generic type	An array can store both **objects and primitives type.**	We **cannot store primitive type** in ArrayList. It automatically converts primitive type to object.
Performance	It performs **fast** in comparison to ArrayList because of fixed size.	**resize() operation :** Automatic resize will **slow** down the performance as it will use temporary array to copy elements from the old array to new array.(by 50%) **add() or get() operation :** almost **same** performance , as for ArrayList object these operations run in constant time.

Que - Difference between ArrayList and LinkedList?
Ans -

Properties	ArrayList	LinkedList
Internal Implementation	ArrayList internally uses a **dynamic array** to store its elements.	LinkedList uses **Doubly Linked List** to store its elements.
Implementation	ArrayList implements only **List**.	LinkedList implements **List as well as Queue**. It can acts as a queue as well.
Manipulation / add / delete operation	Manipulation with ArrayList is **slow** because it internally uses an array. If any element is removed from the array, all the other elements are shifted in memory.	Manipulation with LinkedList is **faster** than ArrayList because it uses a doubly linked list, so no bit shifting is required in memory.
Access/ get operation	ArrayList is **faster** in storing and accessing data as internally Array is used which has random access.	LinkedList is **slower** than ArrayList in storing and accessing data as access requires node by node traversal.
Reverse Iterator	there is no descendingIterator() in ArrayList	LinkedList **can be iterated in reverse** direction using descendingIterator()

Properties	ArrayList	LinkedList
Initial Capacity	If the constructor is not overloaded , then ArrayList creates an empty list of initial **capacity 10**	There is no case of default capacity in a LinkedList. Hence an **empty list** is created when a LinkedList is initialized.
Memory Overhead	In ArrayList Memory overhead is **less** as each index only holds the actual object(data).	Memory overhead in LinkedList is **more** as compared to ArrayList as node in LinkedList needs to maintain the addresses of next and previous node.

Conclusion: LinkedList element deletion and addition is faster compared to ArrayList.

Reason: LinkedList's each element maintains two pointers (addresses) which points to the both neighbor elements in the list. Hence removal only requires change in the pointer location in the two neighbor nodes (elements) of the node which is going to be removed. While In ArrayList all the elements need to be shifted to fill out the space created by removed element.

Hence if there is a requirement of **frequent addition and deletion** in application then **LinkedList** is a best choice and if more **search operations** requirement, **ArrayList** would be your best bet.

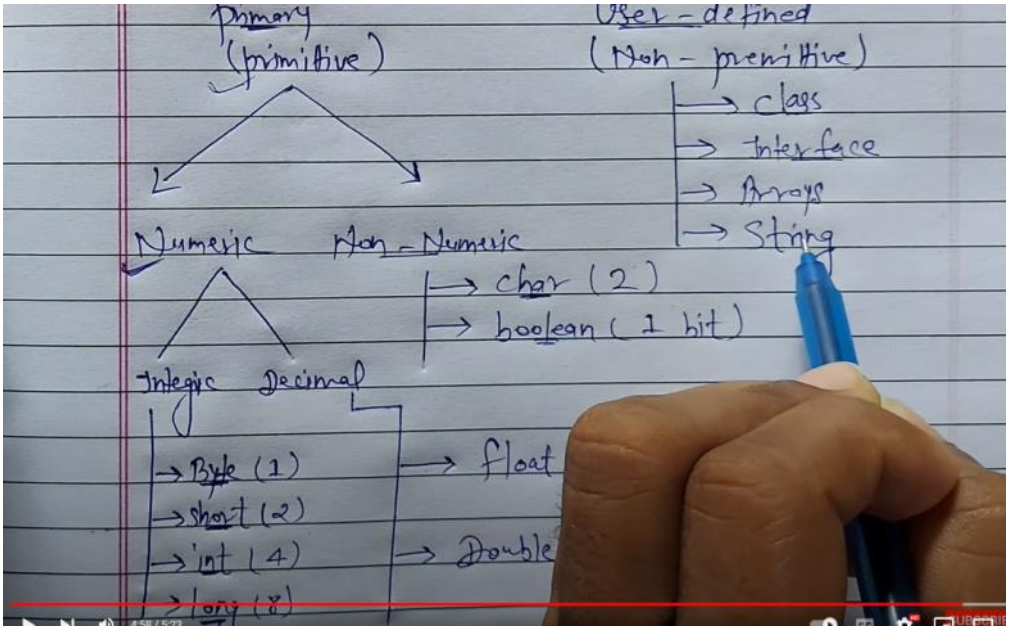
Que - Difference between List and Set?

Ans-

S.No.	List	Set
1	List is an index based data structure.	Set is not an index based data structure.It stores the data according to the hashcode values
2	List can store duplicate elements.	Set does not allow to store duplicate elements
3	List can store any number of null values.	Set can store only one null value
4	List follows the insertion order.	Set does not follows the insertion order.
5	We can iterate(get) the list element by Iterator & ListIterator	We can iterate the set elements by Iterator.

Que - What is Primitve and Non Primitive Data Types?

Ans -



Screen clipping taken: 25-08-2022 15:17

Java Logic Programme

08 August 2022 18:22

Que 1. Reverse the words of the string not individual letter?

String Programs

08 August 2022 18:33

What is String in Java?

Strings, one of the most common objects used in Java programming, are essentially sequences of characters. As an example, the string "Scaler" contains the following characters: "S", "c", "a", "l", "e", and "r". You can either create a string by using String Literal or by using the NEW keyword. Additionally, String supports a variety of methods to operate on Strings, such as the equals method to compare two Strings, the replace method to replace String characters, the substring method to get a substring, the toUpperCase method to convert String to upper case, the split method to split a long String into multiple Strings, and so on.

Now let's look at the most common asked String Interview questions:

- Java String Interview Questions for Freshers
- Java String Interview Questions for Experienced
- String Programming Questions

Java String Interview Questions for Freshers

1. How to declare a string in Java?

String declaration in Java can be done in two ways:

- **By string literal**: Double quotes are used to create Java String literals.
- Example: String str= "Scaler";
- **By new keyword**: Keyword "new" is used to create a Java string.
- Example: String str=new String ("Scaler");

2. Is String a primitive or derived type in Java?

Strings are derived data types. Strings are Java objects that represent sequences of characters. String objects are created using the java.lang.String class. There are many functions that need to be called upon when processing a string, such as substring(), indexof(), equals(), toUppercase(), etc, which primitives types do not have.

3. State the difference between String in C and String in Java.

- **String in C**: In C, strings are just arrays of characters, and they are terminated with a /0, which is why we commonly refer to them as "null-terminated". Strings in C, like "abc$%", actually consist of 6 characters 'a' 'b' 'c' '$' '%' and '/0', but these can be easily manipulated.
- **String in Java**: Java treats Strings as objects, not arrays. String objects are created using the java.lang.String class. String objects in Java are immutable; you cannot modify their contents. This means whenever we manipulate a String object, the new String is created rather than the original string being modified.

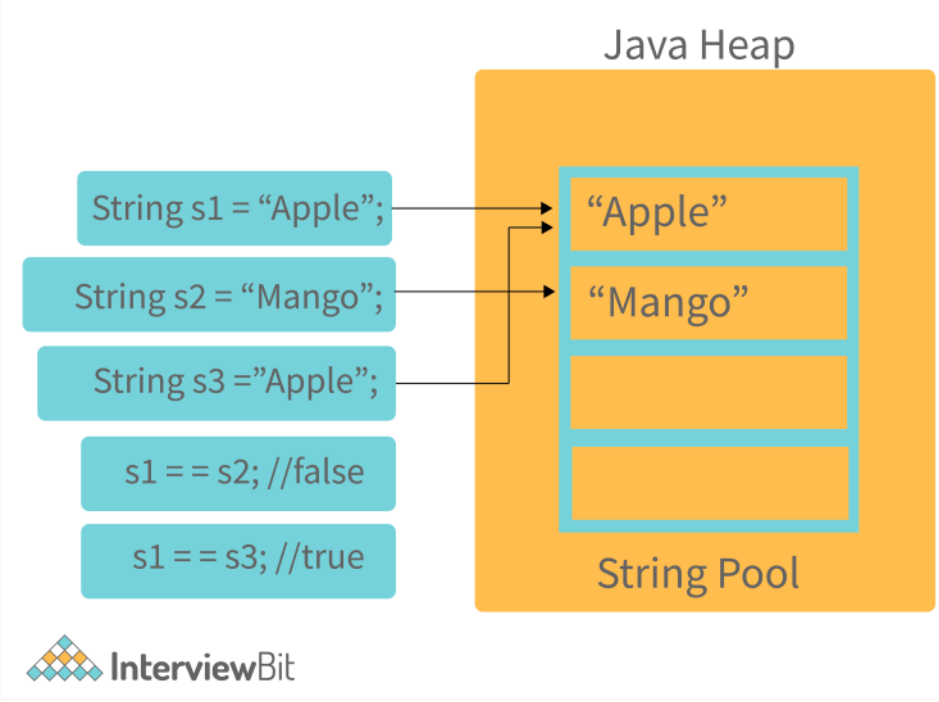
You can download a PDF version of Java String Interview Questions.

Download PDF

4. Explain String pool in Java.

String Pool, also known as SCP (String Constant Pool), is a special storage space in Java heap memory that is used to store unique string objects. Whenever a string object is created, it first checks whether the String object with the same string value is already present in the String pool or not, and if it is available, then the reference to the string object from the string pool is returned. Otherwise,

the new string object is added to the string pool, and the respective reference will be returned.



As shown in the above image, two Strings s1 and s2 are created with the values "Apple" and "Mango". Therefore, when the third String s3 containing the value "Apple" is created, instead of creating a new object, the existing object reference will be returned. Here, s1==s2 is false both strings s1 and s2 refer to different string values from the string pool i.e. apple and mango. We can see that s1==s3 is true because both strings s1 and s3 refer to a single string value from a string pool i.e., apple.

5. Is String immutable or final in Java? If so, then what are the benefits of Strings being Immutable?

Yes, Strings are immutable in Java. Immutable objects mean they can't be changed or altered once they've been created. However, we can only modify the reference to the string object. The String is immutable in Java because of many reasons like security, caching, synchronization and concurrency, and class loading.

6. What does the string intern() method do in Java?

If you apply the intern() method to a few strings, you will ensure that all strings having the same content share the same memory. As soon as a String object is invoked with intern(), it first checks if the string value of the String object is already present in the string pool and if it is available, then the reference to that string from the string constant pool is returned. If not, a new string object is added to the string pool, and a reference to it is returned.

Example:

```
String str1 = new String("Scaler by InterviewBit").intern();  //Line1
String str2 = new String("Scaler by InterviewBit").intern(); //Line2
System.out.println(str1 == str2); //prints true
```

As you can see, the intern() method is invoked on the String objects. When Line1 is executed, memory is allocated within the SCP. In line 2, no new string objects are created in the SCP because str1 and str2 have the same content. As a result, the reference to the object created in line1 is returned. This means that str1 and str2 both point to the same memory. Therefore, the print statement prints true.

7. State the difference between String and StringBuffer.

String objects in Java are immutable and final, so we can't change their value after they are created. Since strings are commonly used in applications, we need to perform several operations on them such as substring(), equals(), indexof(), toUppercase(), etc. Each time we manipulate a string, a new String object is created, and all previous objects will be garbage, placing a strain on the garbage collector. This is why The Java team developed StringBuffer. A StringBuffer is a mutable object, meaning it can be changed, but the string is an immutable object, so it cannot be changed once it has been created.

- **String**

Syntax:

```
String str1="InterviewBit";
String str2=new String("Scaler");
Scanner str3=new Scanner(System.in);
String str4=str3.nextLine();
```

Example: Concatenation Example of String. A string class takes longer to perform a concatenation operation than a string buffer class.

```
publicclassScanner{
   publicstaticvoidmain(String []args){
     StringBuilder stbu=newStringBuilder();
     //Initial object sizeSystem.out.println(stbu.capacity());
     String str="Scaler";
     System.out.println(str);
     String str1 = newString("InterviewBit");
     System.out.println(str1);
     str1 += " Articles";        //string updateSystem.out.println(str1);
   }
}
```

Output:

```
16Scaler
InterviewBit
InterviewBit Articles
```

- **StringBuffer**

Syntax:

```
StringBuffer var = new StringBuffer(str);
```

Example: Concatenation Example of StringBuffer. String buffer class perform concatenation operations more quickly than string classes.

```
publicclassStringBuffer{
   publicstaticvoidmain(String []args){
     StringBuilder stbu=newStringBuilder();
     //Initial object sizeSystem.out.println(stbu.capacity());
     StringBuffer stbr= newStringBuffer("InterviewBit");
     System.out.println(stbr);
     stbr.append(" Articles");      //string updateSystem.out.println(stbr);
     stbr=newStringBuffer("Scaler");
     System.out.println(stbr);
   }
}
```

Output:

```
16InterviewBit
InterviewBit Articles
Scaler
```

8. State the difference between StringBuffer and StringBuilder in Java.

StringBuffer and StringBuilder are two Java classes for manipulating strings. These are mutable objects, i.e., they can be modified, and provide various methods such as insert(), substring(), delete(), and append(), for String manipulation.

- **StringBuffer**: The StringBuffer class was created by the Java Team when they realized the need for an editable string object. Nevertheless, StringBuffer has all methods synchronized, meaning they are thread-safe. Therefore, StringBuffer allows only one thread to access a method at once, so it is not possible to call StringBuffer methods from two threads simultaneously, which means it takes more time to access. The StringBuffer class has synchronized methods, making it thread-safe, slower, and less efficient than StringBuilder. The StringBuffer class was introduced in Java 1.0.
- Syntax:

```
StringBuffer var = new StringBuffer(str);
```

- **StringBuilder**: It was at that point that the Java Team realized that making all methods of StringBuffer synchronized wasn't the best idea, which led them to introduce StringBuilder. The StringBuilder class has no synchronized methods. Unlike StringBuffer, StringBuilder does not offer synchronized methods, which makes it less thread-safe, faster, and more efficient. StringBuilder was introduced in Java 1.5 in response to StringBuffer's shortcomings.
- Syntax:
  `StringBuilder var = new StringBuilder(str);`

9. In Java, how can two strings be compared?

In Java, there are several ways for comparing two strings. The following are a few of them:

- **String Equals Method**: In this method, the strings are compared based on the values within them. If the values of the two strings are the same, it returns true; otherwise, it returns false. This method is case-sensitive.
  Syntax:
  `str1.equals(str2);`
  For example:
  Input 1= Scaler
  Input 2= InterviewBit
  Output= falseInput 1= Scaler
  Input 2= Scaler
  Output= trueInput 1= Scaler
  Input 2= scaler
  Output= false
- **String Equals Ignore Case**: By using this method, the two strings are compared without taking into account the case (upper or lower). It returns true if the two values are the same and not null.

  Syntax:
  `str1.equalsIgnoreCase(str2);`
  For Example:
  Input 1= Scaler
  Input 2= InterviewBit
  Output= falseInput 1= Scaler
  Input 2= Scaler
  Output= trueInput 1= Scaler
  Input 2= scaler
  Output= true
- **Object Equals Method**: The method returns true if its arguments are equal, otherwise, it returns false. Accordingly, if both arguments are null, the result is true, and if just one argument is null, the result is false.
  Syntax:
  `Object.equals(str1, str2)`
  For example:
  Input 1= Scaler
  Input 2= InterviewBit
  Output= falseInput 1= Scaler
  Input 2= Scaler
  Output= trueInput 1= Scaler
  Input 2= nullOutput= falseInput 1= nullInput 2= nullOutput= True
- **String Compare To Method**: This method compares input strings with each other. Upon comparison, the following value is returned:
  1. If (str1>str2), a positive value is returned.
  2. If (str1==str2), 0 is returned.
  3. If (str1<str2), a negative value is returned.

  Syntax:
  `str1.compareTo(str2)`
  Example:
  Input 1= InterviewBit

```
Input 2= Scaler
Output= -10Input 1= Scaler
Input 2= Scaler
Output= 0Input 1= Scaler
Input 2= InterviewBit
Output= 10
```

10. What is the difference between str1 == str2 and str1.equals(str2)?

Java offers both the equals() method and the "==" operator for comparing objects. However, here are some differences between the two:

- Essentially, equals() is a method, while == is an operator.
- The == operator can be used for comparing references (addresses) and the .equals() method can be used to compare content. To put it simply, == checks if the objects point to the same memory location, whereas .equals() compares the values of the objects.

Example:

```
publicclassStringComparison{
  publicstaticvoidmain(String[] args){
     String str1=newString("Scaler");
     String str2=newString("Scaler");
     System.out.println(str1 == str2);
     System.out.println(str1.equals(str2));
  }
}
```

Output:

```
false
true
```

In this example, two different String objects are being created, str1 and str2.

- If str1 and str2 are compared using the == operator, then the result will be false, because both have different addresses in the memory. Both must have the same address in the memory for the result to be true.
- If you use the equals method, the result is true since it's only comparing the values given to str1 and str2, even though they are different objects.

11. Is it possible to compare Strings using the == operator? If so, what is the risk involved?

Yes, you can compare strings using the == operator. One can use == operators for reference comparison (address comparison). The majority of the time, developers compare strings with the == operator, instead of using the equals() method, resulting in an error.

Example:

```
publicclassStringComparison{
   publicstaticvoidmain(String args[]){
     String str1="Scaler";
     String str2="Scaler";
     String str3=newString("Scaler");
     System.out.println(str1==str2);
     //true because both points to same memory allocation System.out.println(str1==str3);
    //false because str3 refers to instance created in heapSystem.out.println(str1.equals(str3));
    //true because both share same content //even if both are different string objects }
}
```

Output:

```
true
false
true
```

12. What is the use of the substring() method in Java?

The substring method is used to return substring from a specified string. This method takes two parameters i.e., beginIndex (the starting index) and endIndex (the ending index). In the case of substring(), method startIndex is inclusive and endIndex is exclusive.

Syntax:

```
substring(int beginIndex, int endIndex)
```

Or

```
substring(int beginIndex)
```

Here,

- **beginIndex**: Index that marks the starting of subsequence and it is inclusive.
- **endIndex**: Index that marks the ending of subsequence and it is exclusive.

Example:

```
importjava.lang.Math;
publicclassInterviewBit{
  // driver codepublicstaticvoidmain(String args[]){
      String str = "Scaler by InterviewBit";

     //prints substring from 7th indexSystem.out.print("Returns: ");
      System.out.println(str.substring(7));
      // prints substring from 0-6, exclusive 6th indexSystem.out.print("Returns: ");
      System.out.println(str.substring(0, 6));
     // prints the substring from 10-22, exclusive 22th indexSystem.out.print("Returns: ");
      System.out.println(str.substring(10, 22));
  }
  }
```

Output:

```
Returns: by InterviewBit
Returns: Scaler
Returns: InterviewBit
```

13. Can we use a string in the switch case in java?

Yes, Java allows you to use strings in switch case conditions. Below is a Java program that shows the use of string in switch case.

Example:

```
publicclassStringinSwitchCase{
  publicstaticvoidmain(String[] args){
     String fruit = "Apple";
     switch(fruit)
     {
        case"Mango":
           System.out.println("Sweet");
           break;
        case"Apple":
           System.out.println("Delicious");
           break;
        case"Orange":
           System.out.println("Luscious");
           break;
        default:
           System.out.println("Not a fruit");
     }
  }
}
```

Output:

```
Delicious
```

Java String Interview Questions for Experienced

14. What are the different string methods in Java?

There are various string operations in Java that allow us to work with strings. These methods or operations can be used for string handling in Java as well as string manipulation in Java. Some of such methods are as follows:

- **split()**: Split/divide the string at the specified regex.

- **compareTo()**: Compares two strings on the basis of the Unicode value of each string character.
- **compareToIgnoreCase()**: Similar to compareTo, but it also ignores case differences.
- **length()**: Returns the length of the specified string.
- **substring()**: Returns the substring from the specified string.
- **equalsIgnoreCase()**: Compares two strings ignoring case differences.
- **contains()**: Checks if a string contains a substring.
- **trim()**: Returns the substring after removing any leading and trailing whitespace from the specified string.
- **charAt()**: Returns the character at specified index.
- **toLowerCase()**: Converts string characters to lower case.
- **toUpperCase()**: Converts string characters to upper case.
- **concat()**: Concatenates two strings.

15. Is String thread-safe in Java?

Strings are immutable objects, which means they can't be changed or altered once they've been created. As a result, whenever we manipulate a String object, it creates a new String rather than modifying the original string object. In Java, every immutable object is thread-safe, which means String is also thread-safe. As a result, multiple threads can access a string. For instance, if a thread modifies the value of a string, instead of modifying the existing one, a new String is created, and therefore, the original string object that was shared among the threads remains unchanged.

16. Why is a string used as a HashMap key in Java?

Basically, the HashMap object can store key-value pairs. When creating a HashMap object and storing a key-value pair in that object, you will notice that while storing, the hash code of the key will be calculated, and its calculated value will be placed as the resultant hash code of the key. Now, when the key is passed to fetch its value, then the hash code of the key is calculated again, and if it's equal to the value of the hash code initially calculated, the initial value placed as the resultant hash code of the key is retrieved or fetched.

Let's say we utilized a variable as a key to store data and then changed the value of that variable. In this case, since we have altered the key, the hash code calculated of the current key will not match the hash code at which its value was originally stored. This makes retrieval impossible. String values are immutable, so once they've been created, they can't be changed. As a result, it is recommended to use Strings as HashMap keys.

17. What is the best way to split a string in Java?

Split() is a Java method for breaking a string based on a Java string delimiter (specified regex). For example, a space or a comma(,) will usually be used as the Java string split attribute to break or split the string.

Syntax:

```
string.split(String regex, int limit)
```

Here,

- **regex**: String is divided at this specified regex.
- **limit (optional parameter)**: Controls or limits the number of resulting substrings. Split() returns all potential substrings if the limit parameter is not specified or is 0.

Example:

```
publicclassSplitString{
 publicstaticvoidmain(String[] args){
  String str = "Scaler by InterviewBit";
  // split string from spaceString[] result = str.split(" ");
  for(inti=0; i < result.length; i++)
  {
    System.out.println(result[i]);
  }
 }
}
```

Output:

```
Scaler
by
```

```
InterviewBit
```

18. Why char array is preferred over a String in storing passwords?

There are various reasons why a char array rather than a string should be used to store passwords. The following are a few of them:

- **Strings are immutable**: The content of Strings cannot be modified/overwritten because any modification will result in the creation of a new String. As a result, we should always save sensitive data like passwords, Social Security numbers, and so on in a char[] array rather than a String.
- **Security**: Because String is immutable, storing the password as plain text keeps it in memory until it is cleaned up by the garbage collector. As string uses SCP (String Constant Pool) for re-usability of a string, it's possible that it'll remain in memory for a long time, and anyone with access to the SCP or memory dump can simply identify or retrieve the password in plain text. That's another reason why we should use an encrypted password instead of plain text.
- **Logfile safety**: With an array, the data can be erased or wiped up, overwritten and the password will not be present anywhere in the system. Whereas, when using plain String, the chances of mistakenly printing the password to monitors, logs, or other insecure locations are substantially higher.

19. Explain the string subSequence method.

The Java String subSequence() method is a built-in function that returns a charSequence (a subsequence) from a string.

20. What do you mean by StringJoiner?

StringJoiner is a Java class that allows you to construct or create a sequence of strings (characters) that are separated by delimiters like a hyphen(-), comma(,), etc. Optionally, you can also pass suffix and prefix to the char sequence.

Example:

```
// importing StringJoiner class  importjava.util.StringJoiner;
publicclassExampleofStringJoiner{
  publicstaticvoidmain(String[] args){
     StringJoiner joinStrings = newStringJoiner(",", "[", "]");
     // passing comma(,) and square-brackets as delimiter   // Adding values to StringJoiner
joinStrings.add("Scaler");
     joinStrings.add("By");
     joinStrings.add("InterviewBit");
     System.out.println(joinStrings);
  }
}
```

Output:

```
[Scaler,By,InterviewBit]
```

21. How can a Java string be converted into a byte array?

The getBytes() method allows you to convert a string to a byte array by encoding or converting the specified string into a sequence of bytes using the default charset of the platform. Below is a Java program to convert a Java String to a byte array.

Example:

```
importjava.util.Arrays;
publicclassStringToByteArray
{
  publicstaticvoidmain(String[] args){
    String str = "Scaler";
 byte[] byteArray = str.getBytes();
 // print the byte[] elementsSystem.out.println("String to byte array: "+ Arrays.toString(byteArray));
  }
}
```

Output:

```
String to bytearray: [83, 99, 97, 108, 101, 114]
```

22. In Java, how do you convert a string to an integer and vice versa?

There is an Integer class in the Java lang package that provides different methods for converting strings to integers and vice versa. The parseInt() method allows you to convert a String into an integer and the toString() method allows you to convert an Integer into a String. Below is a Java program to convert a string to an integer and vice versa.

Example:

```
publicclassStringtoInteger{
 publicstaticvoidmain(String args[]){
   String str1 = "1296";
   inti= Integer.parseInt(str1);
   System.out.println(i);
   String str2 = Integer.toString(i);
   System.out.println(str2);
 }
}
```

Output:

```
12961296
```

23. How can we convert string to StringBuilder?

The append() method can be used to convert String to StringBuilder, and the toString() method can be used to convert StringBuilder to String. Below is a Java program to convert a string array to one StringBuilder object using the append method.

Example:

```
publicclassStringToStringBuilder{
 publicstaticvoidmain(String args[]){
   String strs[] = {"Scaler", "by", "InterviewBit!"};
   StringBuilder sb = newStringBuilder();
   sb.append(strs[0]);
   sb.append(" "+strs[1]);
   sb.append(" "+strs[2]);
   System.out.println(sb.toString());
 }
}
```

Output:

```
Scaler by InterviewBit!
```

24. How do you check whether a String is empty in Java?

The Java String class contains a particular method for determining whether or not a string is empty. The isEmpty() method determines whether or not a string has zero length. In the case where the length of the string is zero, it returns true, or else it returns false.

Example:

```
publicclassStringEmpty{
  // Function to determine if String is emptypublicstaticbooleanisStringEmpty(String str){
     //Use the isEmpty() method //to determine if the string is empty.if(str.isEmpty())
        returntrue;
     elsereturnfalse;
  }
  publicstaticvoidmain(String args[]){
     String str1="InterviewBit";    //non-empty stringString str2="";             //empty
stringSystem.out.println("Str1 \""+ str1 + "\" is empty? "+ isStringEmpty(str1));
     System.out.println("Str2 \""+ str2 + "\" is empty? "+ isStringEmpty(str2));
 }
}
```

Output:

```
Str1 "InterviewBit"is empty? falseStr2 ""is empty? true
```

25. How many objects will be created for the following codes:

A.

```
String str1 = "abc";                    //Line1String str2 = newString("abc");          //Line2
```
B.
```
String str1 = "abc";                    //Line1String str2 = "abc";                    //Line2
```
C.
```
String str1 = newString("abc");         //Line1String str2 = newString("abc");         //Line2
```

- **For A:** In this case, two objects will be created. We know that whenever a Java string is created using a new keyword, then two objects will be created i.e. one in the Heap Area and another one in the String constant pool. When the line1 is executed, the new string object str1 gets created and stored in the string constant pool. However, when line2 is executed, only one object is created using a new operator that gets stored in the heap memory (str2). This is because String constant pool already has a String object with the same string value (abc), and therefore, the reference of the string str1 from the string constant pool is returned.
- **For B:** In this case, one object will be created. Here, for line1 (str1), one new object will get created in String constant pool, whereas for line 2, string str2 will create a reference to the String str1 because the string constant pool already has a String object str1 with the same string value (abc).
- **For C:** In this case, three objects will be created. In the case of line1 (str1), two objects are created, one in the string constant pool and one in the heap memory. As for line 2 (str2), one new object is created and stored in heap memory, but not in the string constant pool because a String constant pool object str1 already has the string object str1 with the same string value (abc).

String Programming Questions

26. How to print all permutations of string in Java?

The term "permutation of the string" refers to all of the conceivable new strings that can be created by swapping the positions of the given string's characters. For example, the string CAT has a total of 6 permutations i.e., [CAT, CTA, ACT, ATC, TCA, TAC]. Below is a Java program to print all permutations of a given string.

```
publicclassInterviewBit{
// Function to display all permutations of the string strstaticvoidprintallPermutns(String str, String
str2){
    // check if string is empty or nullif(str.length() == 0)
     {
      System.out.print(str2 + " ");
      return;
     }

    for(inti = 0; i < str.length(); i++)
     {
      // ith character of strcharch = str.charAt(i);
      // Rest of the string after excluding// the ith characterString str3 = str.substring(0, i) +
str.substring(i + 1);
      // Recursive callprintallPermutns(str3, str2 + ch);
     }
  }
  // Driver codepublicstaticvoidmain(String[] args){
    String s = "cat";
    printallPermutns(s, "");
  }
}
```

Output:

```
cat cta act atc tca tac
```

27. Write a program to calculate the total number of characters in the String?

To find the total count of characters in a specified string, we can use the for loop, while loop, or do while loop. Below is a Java program to calculate the total number of characters in a string using for loop.

```
publicclassTotatCharacters{
```

```
  publicstaticvoidmain(String[] args){
    String str = "Scaler by InterviewBit";
    intcount = 0;
    System.out.println("Input String: "+str);

    //Count total characters in the given string except space    for(inti = 0; i < str.length(); i++)
      {
        if(str.charAt(i) != ' ')
        count++;
      }

    //Display total number of characters in the given string    System.out.println("The total number
of characters in the given string: "+ count);
  }
}
```

Output:

```
Input String: Scaler by InterviewBit
The total number of characters in the given string: 20
```

28. How to reverse a string in Java?

There are different ways to reverse a string in Java-like using CharAt() method, StringBuilder/StringBuffer Class, Reverse Iteration, etc. The reverse() method is available in both the StringBuilder and StringBuffer classes and is commonly used to reverse a string. The reverse () method simply reverses the order of the characters. Below is a Java program to reverse a string using the StringBuilder class.

```
publicclassReverseString{
   // function to reverse a string using StringBuilderpublicstaticString revstr(String str){
       returnnewStringBuilder(str).reverse().toString();
   }
   publicstaticvoidmain(String[] args){
      String str= "Scaler by InterviewBit";
      str= revstr(str);
      System.out.println("Result after reversing a string is: "+ str);
  }
}
```

Output:

```
Result after reversing a string is: tiBweivretnI yb relacS
```

29. How to convert an Array to String in Java?

An array can be converted to a string in four different ways such as Arrays.toString() method, String.Join() method, StringBuilder.append() method, and Collectors.joining() method. Here, we will see an example of the Array.toString() method. Arrays.toString() returns a string representation of the array contents. The string represents the array's elements as a list, enclosed in square brackets ("[]"). The characters ", " (a comma) followed by a space are used to separate adjacent elements. It returns “null” if the array is null.

```
importjava.util.Arrays;
publicclassArrayToString{
   publicstaticvoidmain(String[] args){
       String[] strArray = { "Scaler", "by", "InterviewBit"};
       String str1 = ConvertArraytoString(strArray);
     System.out.println("An array converted to a string: "+ str1);
   }
   // Using the Arrays.toString() methodpublicstaticString ConvertArraytoString(String[] strArray){
    returnArrays.toString(strArray);
   }
}
```

Output:

```
An array converted to a string: [Scaler, by, InterviewBit]
```

30. Is it possible to count the number of times a given character appears in a String?

The charAt() method of the String class can be used to find out the number of times a specified character appears in a string. Below is a Java program to check the occurrences of a specified character in a string.

```
publicclassCkeckforOccurences{
    publicstaticvoidmain(String[] args){
       String str= "InterviewBit";
       charch = 'e';  //character to look for occurrences is eintcount = 0;
       for(inti = 0; i < str.length(); i++)
        {
        if(str.charAt(i) == ch)
           {
           count++;
           }
        }
    System.out.println("The character '"+ ch + "' appears "+ count + " times in the given string '"+ str
+ "'.");
    }
}
```

Output:

```
The character 'e'appears 2times in the given string 'InterviewBit'.
```

As you can see, the above program checks how many times the character ch occurs in the string str. Whenever we encounter the character ch in the string, we increase the count by one. Finally, we print the total character count at the end.

31. In what way should two strings be compared to determine whether they are anagrams?

If two strings contain the same characters but in a different order, they can be said to be anagrams. Consider dusty and study. In this case, dusty's characters can be formed into a study, or study's characters can be formed into dusty. Below is a java program to check if two strings are anagrams or not.

```
importjava.util.Arrays;
publicclassCheckAngagram{
  publicstaticvoidmain(String[] args){
      String str1 = "Bored";
      String str2 = "Robed";

      //Convert strings to lowercasestr1 = str1.toLowerCase();
      str2 = str2.toLowerCase();
      // Check to see if the lengths are the sameif(str1.length() == str2.length())
      {
          // convert strings into char arraychar[] str1charArray = str1.toCharArray();
          char[] str2charArray = str2.toCharArray();
          // sort the char arrayArrays.sort(str1charArray);
          Arrays.sort(str2charArray);
          // if the sorted char arrays are same or identical// then the strings are anagrambooleanresult
= Arrays.equals(str1charArray, str2charArray);
          if(result)
          {
            System.out.println(str1 + " and "+ str2 + " are anagrams of each other.");
          }
          else{
            System.out.println(str1 + " and "+ str2 + " are not anagrams of each other.");
          }
      }
      else{
```

```
        System.out.println(str1 + " and "+ str2 + " are not anagrams of each other.");
      }
    }
}
```

Output:

```
bored and robed are anagrams of each other.
```

In the above program, there are two strings i.e., str1 and str2. Here, we are comparing str1 and str2 to determine if they are anagrams. The strings are first converted to lowercase since Java is case-sensitive and B and b are two different characters in Java. Here,

- **str1.toCharArray()**: Convert or transform the string into a char array.
- **Arrays.sort()**: It sorts the char arrays.
- **Arrays.equals()**: Checks or verifies if sorted char arrays are equal.

In the case where sorted arrays are equal, the strings are anagrams.

32. How can we remove a specific character from a String?

There are several ways to remove a character from a string, such as removing the character at the beginning of the string, the end of the string, or at a specific position. It is possible to remove a specific character from a string using the replace() method. You can also use remove() with different variations like replaceFirst(), replaceAll(), etc. Below is a Java program that uses replace(), replaceFirst(), and replaceAll() methods to remove characters from a String.

```
publicclassRemoveCharacter{
 publicstaticvoidmain(String args[]){
    String str = "Scaler by InterviewBit";

    //remove the specified character.System.out.println("String after removing 'e' =
"+str.replace("e", ""));
   //remove the first occurrence of the specified character.System.out.println("String after removing
First 'e' = "+str.replaceFirst("e", ""));
//remove all occurrences of the specified character.System.out.println("String after replacing all
small letters = "+str.replaceAll("([A-Z])", ""));
 }
}
```

Output:

```
String after removing 'e'= Scalr by IntrviwBit
String after removing First 'e'= Scalr by InterviewBit
String after replacing all small letters = caler by nterviewit
```

33. Write a program to check whether the given input string is a palindrome.

When a string is the same when read right to left or left to right, it is called a palindrome. To assess whether a string is a palindrome or not, we first reverse the string and then compare the reversed string with the original one. Below is a Java program that will check if a string is a palindrome.

```
publicclassPalindromeChecker
{
  publicstaticvoidmain(String[] args)
  {
    String str1 = "rotator";
    String revstr = reverseString(str1); //revstr=reverse stringif(str1.equals(revstr))
      {
        System.out.println("The string"+ str1 + " is a Palindrome String.");
      }
    else
      {
       System.out.println("The string"+ str1 + " is not a Palindrome String.");
      }
  }
  // a method for reversing a stringpublicstaticString reverseString(String str2)
  {
```

```
    String revstr = "";
    for(inti = str2.length() - 1; i >= 0; i--)
     {
       revstr += str2.charAt(i);
     }
       returnrevstr;
   }
}
```

Output:

```
The string rotator is a Palindrome String.
```

As shown in the above example, we have a string "Rotator" stored in string object "str1" and another string object "revstr" to store the reverse of str1. To check whether two strings are equal or not, we have used the equals() method.

34. What will be the output of the below program?

```
String str1 = "scaler";                  //Line1String str2 = newString("scaler");     //Line2str2.intern();
//Line 3System.out.println(str1 == str2);
```

The output of the above program is false. We know that when the intern() method is executed or invoked on a string object, then it checks whether the String pool already has a same string value (scaler) or not, and if it is available, then the reference to the that string from the string constant pool is returned. In the above example, the intern method is invoked on str2. However, since we didn't assign it back to str2, str2 remains unchanged and therefore, both str1 and str2 have different references. If we change the code in line 3 to str2 = str2.intern(), then the output will be true.

35. What is the output of the below program?

```
publicclassStringTest{
publicstaticvoidmain(String[] args){
 String str1 = newString("interviewbit");
 String str2 = newString("INTERVIEWBIT");
 System.out.println(str1 = str2);
}
}
```

This program prints "INTERVIEWBIT" since str2 String is assigned to str1 String. The comparison operator “==” should not be confused with the assignment operator “=”.

Conclusion

Here's everything you need to know about Java String interview questions and answers. To summarize, there are many specifics related to String that every Java programmer needs to be familiar with and these String questions will not just help you prepare better for Java interviews, but will also open a new door to learning more about String.

The more familiar you are with these frequently asked interview questions, the greater your chances of getting hired.

Hopefully, we were able to answer any questions or concerns you had. Wishing you luck in your future endeavours.

Recommended Interview Preparation Resources

- Java Projects With Source Code
- Java MCQ With Answers
- Java Interview Questions for 5 years Experience
- Technical Interview Questions
- Coding Interview Questions
- InterviewBit Blog
- DSA

String MCQs

1.

Which of these methods of the String class retrieves characters at a specific index?

char()

concat()

charat()

charAt()
2.
Can one of these methods be used to check the equality of strings?
equal()
equals()
isequal()
isequals()
3.
What will be the output of the following code?

```
publicclassStringClass{
   publicstaticvoidmain(String args[]){
      String str = "Scaler";
      String str1 = "InterviewBit";
      String str2 = "Scaler";
      System.out.println(str.equals(str1) + " "+ str.equals(str2));
   }
}
```

true false
true true
false true
false false
4.
What String method is used to determine the length of a String object?
length()
get()
Sizeof()
lengthof()
5.
If the invoking string is less than the string compared, what value will be returned by the function compareTo?
Zero
Negative value
Positive value
None of the above
6.
Which class should be used when creating mutable objects?
StringBuffer
StringBuilder
Both A and B
None of the above
7.
What will be the output of the following code?

```
publicclassCompare{
   publicstaticvoidmain(String[] args){
   String s1 = "Scaler";
   String s2=newString("Scaler");
   System.out.println(s1==s2);
   }
}
```

true
false
Compilation Error
None of the above
8.
What will be the output of the following code?

```
publicclassMain{
 publicstaticvoidmain(String args[]){
   String str = "InterviewBit".replace("e", "");
   System.out.println(str);
```

```
 }
}
```
Int rvi wBitsdsd
IntrviewBit
IntrviwBit
InterviwBit
9.
String objects are mutable. True or False.
True
False
10.
Java String SCP stands for ___.
String Collection Pool
String Common Pool
String Constant Pool
String Count Pool

From <https://www.interviewbit.com/java-string-interview-questions/#what-is-the-best-way-to-split-a-string-in-java>

Que - What is string constant pool?

Functional Testing Question

24 August 2022 00:32

Que -1) What is Test Plan?
Ans - A Test Plan is a detailed document that describes the test strategy, objectives, schedule, estimation, deliverables, and resources requir ed to perform testing for a software product. Test Plan helps us determine the effort needed to validate the quality of the application under test. The test plan serves as a blueprint to conduct software testing activities as a defined process, whic h is minutely monitored and controlled by the test manager.

You already know that making a **Test Plan** is the most important task of Test Management Process. Follow the seven steps below to create a test plan as per IEEE 829

- Analyze the product
- Design the Test Strategy
- Define the Test Objectives
- Define Test Criteria
- Resource Planning
- Plan Test Environment
- Schedule & Estimation
- Determine Test Deliverables

Step 1) Analyze the product

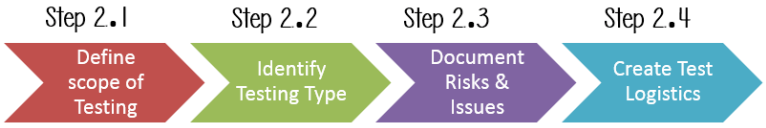
How can you test a product **without** any information about it? The answer is **Impossible.** You must learn a product **thoroughly** before testing it.
The product under test is Guru99 banking website. You should research clients and the end users to know their needs and expec tations from the application

- Who will use the website?
- What is it used for?
- How will it work?
- What are software/ hardware the product uses?

Step 2) Develop Test Strategy

Test Strategy is a **critical step** in making a Test Plan in Software Testing. A Test Strategy document, is a high -level document, which is usually developed by Test Manager. This document defines:

- The project's **testing objectives** and the means to achieve them
- Determines testing **effort** and **costs**
- To develop Test Strategy for testing, you should follow steps below



Step 2.1) Define Scope of Testing

Before the start of any test activity, scope of the testing should be known. You must think hard about it.

- The components of the system to be tested (hardware, software, middleware, etc.) are defined as "**in scope**"
- The components of the system that will not be tested also need to be clearly defined as being "**out of scope**."

Defining the scope of your testing project is very important for all stakeholders. A precise scope helps you

- Give everyone a **confidence & accurate information** of the testing you are doing
- All project members will have a **clear** understanding about what is tested and what is not

How do you determine scope your project?
To determine scope, you must –

- Precise customer requirement
- Project Budget
- Product Specification
- Skills & talent of your test team
- https://www.guru99.com/what-everybody-ought-to-know-about-test-planing.html

Que-2) What is Test Strategies?
Ans -

Testing Concept Automation and Functional

21 September 2022 15:36

Que - 1 What will be your approach to start automation ?

Ans:-

- Define your high business value tests.
- Identify your risk
- Understand your technology, tools and resources
- Make sure your data is good
- Define your DevSecOps
- Considering your test environment
- Tag your tests
- Look for testing efficiencies
- Embrace agile tools

Step 1: Define your high business value tests

The first step is to define what's most important to test by defining what we call the "high business value tests" – ie, the flows that could potentially cause the business to fail if they stopped working. Consider Knight Capital Group, the trading company that went from fully functional to belly up in just 45 minutes and lost $485 million due to the failure of a single software book (ie, a high business value test that wasn't considered). Be sure to work with your business to understand what your high business value tests are because it will allow you to understand if the solution you're proposing fits your critical scenarios. It will also help you demonstrate true business value with your automation framework, which will dovetail nicely into the ROI conversations.

Step 2: Identify your risk

A key part of any test automation strategy is knowing what to test first and what to test last. You should use a risk-based approach to determine this testing automation priority. You can determine the risk, or priority, of each thing you want to test by figuring out its business impact and adding that to the probability of it failing. Obviously, the things with the highest business impact and highest probability of failure should be highest on your priority list, while the things with the lowest business impact and lowest probability of failure should be at the bottom. This will also help quite a lot with the aforementioned testing squeeze and knowing what you want to cut first.

Step 3: Understand your technology, tools, and resources

You need to understand how your overall testing automation solution will affect your overall environment. Do you have the proper accounts to run this? Do you have the proper environmental access? Do you have the right libraries and APIs and other pieces you may need to have your testing automation solution talk to your applications? You need to have a robust working solution that you can easily build into your overall framework without bogging anything down or creating broken or fragile tests.

Recommended: How to select Automation Testing Tools?

Step 4: Make sure your data is good

A lot of test automation projects fail due to data failures. What if you could ensure the data is correct right at the start using another script in your automation framework or by running pre-scripts to validate or load the data, thereby saving you hours and hours of rewriting or redoing your tests? For each release and big framework iteration, you should deeply examine how you're handling your data, how you're storing your data, where your data is coming from, what your retry logic is, and if you have to worry about masking or de-identifying data.

Step 5: Define your DevSecOps

A lot of testers can now work directly with Jenkins servers or other build-and-deploy tools, so you need to define that in your testing automation strategy.

You need to ask:

- Where is the code stored?
- How are you deploying it?
- What environments are you running it on and are they safe?
- Are the libraries and open source code you're using secure?

You need to be doing some level of security scanning and have a process for how that scanning is done for your test automation framework.

Step 6: Consider your testing environment

There are a lot of things to document around environmental conditions. Do you need certain tokens or VPNs? Do you need a launch box? How does that work and where does that live and who is responsible for it? How is the patching done on those systems? You need to document all of this, as it will also help greatly with onboarding new testers to your organization and getting the logins set up. In short, you need to understand where your code is running at all times and have it fully documented.

Step 7: Tag your tests

Having the ability to tag your tests and group them logically will allow you to say, "It doesn't matter whether we have 20,000 or 50,000 test automation scripts because I know these are for my check-out, these are for my login, these are for smoke tests, etc…" If you don't have those tags in place, you may be left doing a lot of groundwork trying to figure out the purpose of certain tests and figuring out which tests to run. Set a tagging agreement right up front to ensure consistent tagging and regular updates of the most commonly used tags.

Step 8: Look for testing efficiencies

As you do more and more testing, you can start to apply the same testing logic in different areas to create efficiencies and shave off testing time and resources. For example, if you're doing the same thing with the unit testing, manual testing, and automation testing, you probably don't need three people running the exact same test. You can save a lot of time by

fully understanding your overall testing automation strategy, all the way from the unit test down to the UI test.

Step 9: Embrace agile tools

Embrace your agile and DevOps tools and really work on your documentation. Your testing automation strategy should become a living document that's updated and reviewed each sprint to make sure you're sticking to your visions and goals. Embrace this process and use cloud-based tools such as GitHub to make it a success. That said, you don't need to document *everything*. You should, however, have regular check-ins and micro-strategy sessions.

[Video] Katalon x PNSQC: Test Automation Strategy: Insider Practices & Secrets here

Tying it all Together

Remember: the main thing you want to accomplish with your test automation strategy is to focus on your goals and communication and not your format. How will you deliver value? What is your overall technology goal? How is it affecting the people in your organization and your environments?

Also—be sure to win the support of your business partners, product owners, and project managers, because they're the ones who are going to support you when the testing squeeze comes. As long as they understand your strategy, they'll trust you to make the decisions on what needs to be cut and what doesn't.

Finally, use your testing automation strategy as a basis for investing in the right technologies and growing your business through innovation—because, after all, that's what this is all about, right? Growth, innovation, and not having to go in reverse.

Step #3. Using the correct tool for automation

This point deserves its own article (and I will write one on that). This is another difficult step in the process of starting automation. **There are various tools in the market, but you have to select those which are best for your application.**
To make it short, I will write the most important considerations while selecting the tool. I will explain the tool selection process in detail in a separate article.
The most important things to consider while selecting the right tools are:

1. The tool must be within your **budget**. The automation tools are really expensive. So the company should have the budget to purchase the tool.
2. The tool must **support technologies** used in your application. If your application is using flash or Silverlight, the tool must support it. If your application is running on mobile, the tool must be able to execute scripts on mobile. You can purchase a single tool that supports all technologies used in your application or you can purchase separate tools for each technology. **For example**, you can use selenium for your web applications, Robotium for your Android applications, and MS Coded UI for desktop applications. Whatever the decision, this should be in your budget.
3. You must have the necessary **skilled resources** who can use this tool or learn that tool in less time. **For example**, if you have hired an automation architect who has only experience in QTP, and you are purchasing a license for MS Coded UI, the resource might not be comfortable using it. Tools are like good cars, but you must have good drivers too to drive these good cars.
4. The tool must have a **good reporting mechanism** to show the results to stakeholders after each execution.

Step #4. Analyzing various applications to determine those which are best suited for automation

If your organization is working on 5 applications, it is not necessary that each should be automated. We have to see the various factors while selecting any application to automate.
The application which should be automated must have these factors:

5. The application should not be in the early stages of its development. (The application should have all or some modules which are stable and tested by manual testers)
6. The UI of the application must be stable. (The UI must not change frequently)
7. The manual test cases of this application should be in written form.

The main goal of automation is to make sure that if the application is bug-free in one build, it should remain bug-free in the next build. The manual tester should not waste their time in finding regression issues, these issues should be identified in automation.
So to find a regression, we must have an application that is already stable and has some test cases written for it. The automation team will convert these test cases into scripts and will run these scripts on every build to make sure no regression appears.
Also, read => How to Select Correct Test Cases for Automation Testing

Step #5. Training the Team

After tool selection and resource hiring, the next step is logically the training of the resources.
If manual testers are converted into automation engineers, they have to be trained on automation terminologies and concepts. If an automation architect is hired from outside, he must get knowledge about the product to test, the manual testing process, and what management is expecting.
Give resources some time to try different things until they finally come up with a winning automation strategy. Train them on the tools which the organization is already using bug tracking software and requirements management software.
Good training and strong communication between manual testers, developers, and the automation team is really necessary.

Step #6. Creating the test automation Framework

The biggest task for the automation architect is to come up with an automation framework that should support automated testing in the long run.
The automation framework is basically the set of rules and careful planning to write the scripts in a manner that results in the least amount of maintenance. If anything changes in the application, the scripts need little or no updating to cope with that change. That is the beauty of an automation framework.
Also, read =>> Top websites to learn Automation Testing
There are five kinds of automation frameworks, namely linear, modular, data-driven, keyword-driven, and hybrid. All of these frameworks will be covered in detail with examples in a separate article in this series.
You can also start reading more on automation frameworks in the following tutorials:
=> Why Do We Need Framework for Test Automation?
=> QTP Framework examples
=> Selenium Framework examples

Step #7. Developing an Execution Plan

The execution plan includes selecting which environments the scripts will be executed. The environment includes OS, Browser, and different hardware configurations.
For example, if the test case demands that it should check the website in 3 browsers, namely, Chrome, Firefox, and IE, then the automation team will write the script in such a manner that it will be able to execute in each browser. This should always be told before writing the scripts because it will be taken care of in scripts if the automation team knows it beforehand. The execution plan should also state who will execute the scripts. Normally the automation team executes the scripts on every build, but it varies from company to company. Some managers ask developers to execute these scripts on their build before release and some companies hire a dedicated resource just for the execution. Even some companies run scripts in unattended mode, which of course requires no additional resource.

Step #8. Writing Scripts

When the framework is designed, the execution plan is known and resources are trained on the new tool, now it's the right time to start writing scripts.
Scripts should be written in an organized manner with the proper naming conventions. The source code should be maintained in source control to avoid code loss. Version control and history should be maintained. Test automation is just like software development. All best programming practices should be taken care of while writing the scripts.
Also, read => How to Translate Manual Test Cases into Automation Scripts

Step #9. Reporting

The reporting feature is usually provided by the tool. But we can create custom reporting mechanisms like auto-emailing the results to management.
We can create reports at the end of each execution in the form of charts and tables if management needs them. The management should always be informed about the test case coverage, that means which manual test cases are covered in automation and which of them are remaining.

Step #10. Maintenance of Scripts

If best programming practices are followed and the framework is good, then maintenance will not be a problem.
Maintenance usually occurs when there is a change request in an application. The scripts should immediately be updated to cope with that change to ensure flawless execution.
For example, if you are writing some text in the textbox through the script and now this text box becomes the drop-down list, we should immediately update the script.
Some other kinds of changes include that your scripts were running on the English version of the application. Now there is a change request that the application should support Chinese. Your framework should allow you to update your scripts with little effort to support execution in Chinese too! That is why Automation architects are expensive. :)
If the framework is not good and best practices are not followed, then maintenance will become a nightmare. Most automation projects fail due to poor maintenance of scripts.

From <https://www.softwaretestinghelp.com/automation-testing-tutorial-3/>

Que - How to select best test case candidate to automate?

Ans -

- Repetitive tests that run for multiple builds
- Tests that tend to cause human error
- Tests that require multiple data sets
- Frequently used functionality that introduces high risk conditions
- Tests that are impossible to perform manually
- Tests that run on several different hardware or software platforms and configurations
- Tests that take a lot of effort and time when manual testing

Que - How to select best Automation Tool for your project?

Ans -

Depending on your projects and your QA team's nature, your choice of automation testing tool must be compatible with many different aspects, such as project scope and requirements, in addition to its reputation. The best tool available does not guarantee the best testing outcome. It must be the right one.

The Significance of Test Automation

A lot of controversy about whether manual testing has fallen behind as automation testing has risen in popularity. Regardless of contrary opinions, the importance of test automation nowadays is no doubt undeniable.

The software market now demands "Quality at Speed" from industry players. As the name implies, "Quality at Speed" means higher quality products must reach end-users but in a shorter time span than before. This difficult demand fueled test automation's impressive growth and turned it into a game-changer by allowing QA teams to execute faster and more accurate test cases.

Test types that require repetitive actions, like regression testing, are what needed to be automated the most. Frequent changes in software will significantly escalate the total cost in terms of time and human resources used to run tests manually. Therefore, automating tests is the smarter and more efficient choice in such cases.

You can dig a bit deeper into test automation with this complete introduction of automation testing.

Choose the Right Automation Tool, Not the Best One

Regardless of these advantages, test automation does not work the same for all projects. While many QA teams have benefited from automation, other companies have wasted time, efforts, and financial resources in implementing automation tools.

Automation testing success primarily lies in identifying the right tool for different needs and demands. This process takes time and effort at first, but it is a must for your team to automate tests efficiently in the long run.

Types of Automation Testing Tools

All available test automation tools can be divided into three types in general as below.

1. Open-source automation tools
   These tools are free platforms that allow users to access and use their source code. Users can opt to fully adopt the code or modify it to suit their testing needs. This type of tool is free of charge and developed by the community. Open-source tools are the top choice for many automation testers with a programming background due to its free access and the ability to customize advanced test cases.
2. Commercial automation tools
   Commercial tools are produced to serve commercial purposes and usually distributed via subscription plans. Users must purchase a paid license to use the software. Compared with open source software, this kind of tool usually has more premium features and thorough customer service that completes the whole testing process for companies or enterprises.
3. Custom framework
   There are niche projects that a single open-source software or fixed commercial testing tool cannot fulfill the requirements. They are mainly due to differences in their testing processes and testing environments. In such cases, teams need to develop customized software on their own. The custom framework is much more complicated than the two other solutions and can be deployed by technical experts.

How to Select the Right Automation Tool for Your Project?

Get to know your needs and testing demands first

Before proceeding with the tool selection process, considering whether Test Automation is the right direction for your team at the moment is critical. Not all QA teams need automation to accelerate their testing process. Manual testing still plays an essential part in this field for specific needs and project requirements.

So when is Test Automation necessary?

- When there are many repetitive test cases to be done
- When there are frequent regression tests
- When the team has to simulate a vast number of users for performance testing
- When the UI is apparently stable
- When critical functionalities cannot solely rely on manual testing

These are among the most demanding requirements for the application of test automation. QA practitioners need a deep understanding of their projects to accurately identify them.

Criteria for Automation Testing Tool Evaluation

1. Does your team possess the necessary skills to best utilize the tool?
   Automation testing is much more technical than manual testing. In many automation tools, especially open-source software, testers must possess a sufficient level of programming knowledge to write and execute test scripts. This technical barrier appears to be the most challenging obstacle in adopting test automation for QA teams with limited

IT background.
Testing tools requiring no coding in execution have proven to be a promising solution to this bottleneck. You can find out more about top codeless automation testing tools for 2022 here.

2. What is your team budget?
   Here's a fact: Test automation is not affordable in many cases. However, it brings out a positive ROI for your team and business in the long run as long as the budget is calculated thoroughly. Depending on the budget, it will be easier for you to pick the appropriate software, an open-source or commercial tool. We will talk about these types of tools further in the following part of this article.
3. What features to look for?
   While the requirements vary from team to team, there are some key factors that you should always take in consideration when choosing a suitable automation tool. They include:

- Supported platforms
- Applied application under tests
- Programming languages
- CI/CD integration capabilities
- Reporting functionality

4. How difficult is script maintenance and reusability?
   A significant factor that escalates the total cost for test automation is script maintenance. Pre-written scripts in automation testing are fragile by nature. The ideal automation tool should come with capabilities to reduce such effort, such as eliminating object locator flakiness. On the other hand, script reusability saves you and the team a great deal of time for similar test cases as you can reuse test scripts.
5. How are the integration capabilities?
   The selected automation tool must be able to integrate to CI/CD pipelines and external platforms to ensure the testing continuity. Robust and comprehensive integration also allows you to better your test management and team collaboration.
6. How and where can you get technical support?
   Another key point to be noticed is the support for your tool. For commercial tools, they should provide users with prompt customer support for all technical issues. Remember to check out their official documentation and website to see what means of support you can get. In terms of open-source software, a large and active user community is what you can lay on whenever encountering problems.

Top Suggestions for Test Automation Tools

For your reference, we gathered here the list of some trusted automation tools that fulfill all of the criteria above, including both open-source and commercial ones.

1. Selenium
   Selenium is an open-source framework to perform web testing across browsers and platforms. The tool is widely used and modified by developers and testers worldwide, which means you can easily find support from the community via its official website and forums. Selenium has become the standard framework for many other tools to be built on top of it.
2. Katalon Studio
   Katalon Studio is a freemium automation tool with both free and paid versions. The tool can be used to automate Web, API, Mobile, and Desktop applications testing— with smart analytics and CI/CD integrations. Katalon Studio is built on top of Selenium framework and inherits some of the most outstand ing features.
3. SoapUI
   SoapUI is an open-source web service testing application for Simple Object Access Protocol and representational state transfers. Its functionality covers web service inspection, invoking, development, simulation and mocking, functional testing, load testing, and compliance testing.
4. JMeter
   As a part of the Apache Software Foundation, JMeter is an open-source Java web testing platform. JMeter is primarily used for functional and performance tests thanks to its stimulation mechanism, making extensive scale tests feasible.

From <https://katalon.com/resources-center/blog/automation-testing-tool-strategy>

embrace

28 September 2022 11:17

SQL_Questions

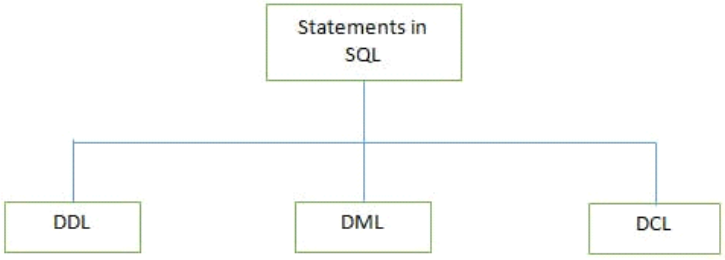
30 November 2022 11:33

Q #1) What is SQL?
Answer: Structured Query Language SQL is a database tool that is used to create and access the database to support software applicati ons.

Q #2) What are tables in SQL?
Answer: The table is a collection of records and information in a single view.

Q #3) What are the different types of statements supported by SQL?

Answer:



From <https://www.softwaretestinghelp.com/50-popular-sql-interview-questions-for-testers/>

Cucumber_Question

30 November 2022 11:33

Rest_API/Rest_Assured

01 December 2022 16:16

1.) What is EndPoint and Base URI ?
Ans - Addresses where API is hosted on the server.

2.) What is GET in API Testing?
Ans - The GET method is used to extract information from the given server using a given URI. While using GET request, it should only extract data and should have no other effect on the data. No PayLoad/Body required.

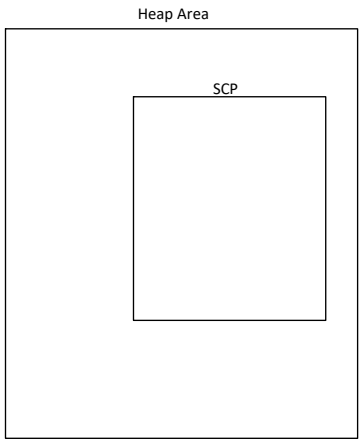
String Interview Questions

30 July 2022 13:09

String Constant Pool (SCP) -

SCP was situated in Method Area (memory type) in the 1.6 version but from 1.7 its been transferred to Heap Area, this is beca use in Method Area the size of SCP was constant but in Heap Area, it can increase and decrease its size.
SCP is used for storing the string object.

Let's create one string object:-



xz

23 August 2022 18:02

mark to margin
order types
interest rate swap
middle office
test plan
fixed protocol, pipco messages
self join, inner join