**Boot Camp Part 1**

**Albert Halder**

**Weekend- Day Batch**

**20612MD**

**DIFFICULT:**

1. **What is the difference between the throw and throws keywords in Java?**

The throw keyword is used to explicitly throw an exception in Java. It is used within the body of a method to indicate that an exception has occurred and should be handled by the calling method or the JVM.

On the other hand, the throws keyword is used in the method signature to indicate that a method may throw one or more exceptions. It is used to delegate the responsibility of handling exceptions to the calling method or the JVM.

Here are some key differences between throw and throws in Java:

* The throw keyword is used to explicitly throw an exception, while the throws keyword is used to declare that a method may throw one or more exceptions.
* The throw keyword is used within the body of a method, while the throws keyword is used in the method signature.
* The throw keyword can only throw one exception at a time, while the throws keyword can declare multiple exceptions at once.
* The throw keyword can be used with both checked and unchecked exceptions, while the throws keyword is only used with checked exceptions.

1. **What is the purpose of the finalize() method in Java? Why is it considered a bad practice to use it?**

The finalize() method is a method of the Object class in Java that is called by the garbage collector before destroying an object from memory. Its purpose is to perform cleanup activities on the object before it is deleted.

However, it is considered bad practice to use the finalize() method in Java for several reasons:

* The finalize() method is not guaranteed to be called by the garbage collector, so you cannot rely on it to release resources or perform other cleanup activities.
* The finalize() method can cause performance issues because it can delay the garbage collection process.
* The finalize() method can make your code more difficult to understand and maintain because it is not clear when it will be called or what it will do.

Instead of using the finalize() method, you should use try-with-resources statements or other mechanisms to release resources and perform cleanup activities.

1. **What is the difference between an abstract class and an interface in Java? When would you use abstract over interface?**

In Java, an abstract class is a class that cannot be instantiated and can contain both abstract and non-abstract methods. An interface, on the other hand, is a contract that specifies a set of methods that a class must implement.

Here are some differences between abstract classes and interfaces:

* A class can implement multiple interfaces, whereas the class can inherit only one abstract class.
* An interface does not have access modifiers. Everything defined inside the interface is assumed to have a public modifier, whereas an abstract class can have an access modifier.
* An abstract class can have final, non-final, static and non-static variables. The interface has only public static final constants.
* An abstract class can provide an implementation for some of its methods, whereas an interface cannot provide any implementation for its methods.

When to use an abstract class over an interface depends on the specific requirements of your program. If you want to provide a default implementation for some of the methods in your base class or if you want to define non-static or non-final variables in your base class, then you should use an abstract class. If you want to define only method signatures without providing any implementation details, then you should use an interface.

1. **What is a Lambda expression in Java? Provide an example to explain its use.**

A lambda expression is a concise way to represent an anonymous function in Java. It allows you to pass a block of code as an argument to a method or constructor. Lambda expressions were introduced in Java 8 and are used extensively in functional programming.

Here’s an example of how you can use a lambda expression in Java:

**A screenshot of a computer program

Description automatically generated with low confidence**

In this example, we’re using a lambda expression to sort the names list and print each element of the list. The lambda expression (a, b) -> a.compareTo(b) is used as an argument to the Collections.sort() method to sort the list. The lambda expression name -> System.out.println(name) is used as an argument to the names.forEach() method to print each element of the list.

1. **What is the difference between the continue and break statements in Java?**

In Java, the break and continue statements are used to control the flow of a loop. The break statement is used to exit a loop early. When a break statement is encountered inside a loop, the loop is immediately terminated, and the program continues with the next statement after the loop. [On the other hand, the continue statement is used to skip the current iteration of a loop and move on to the next iteration](https://www.w3docs.com/snippets/java/difference-between-break-and-continue-statement.html)

A screenshot of a computer program

Description automatically generated with medium confidence

1. **What is the purpose of the volatile keyword in Java? Provide an example to explain its use.**

In Java, the volatile keyword is used to indicate that a variable’s value may be modified by multiple threads. [It guarantees that any changes made to the variable are immediately visible to all other threads that access the variable**1**](https://www.datatrained.com/post/volatile-keyword-in-java/)

The benefits of using the volatile keyword in Java are:

* Ensuring visibility of the latest value of a variable across multiple threads.
* Preventing race conditions and ensuring consistency of data.
* Avoiding the need for explicit locking mechanisms like synchronized blocks or the use of locks.
* Making code simpler and easier to understand by reducing complexity related to synchronization.
* [Enabling safe and efficient communication between threads](https://www.prepbytes.com/blog/java/volatile-keyword-in-java/).

A screenshot of a computer program

Description automatically generated with medium confidence

In this example, the flag variable is marked as volatile. This ensures that any changes made to the variable are immediately visible to all other threads that access it. The doSomething() method will keep running until the flag variable is set to true. Without the volatile keyword, there is no guarantee that changes made to the flag variable by one thread will be immediately visible to another thread.

1. **What is the difference between Array List and LinkedList in Java? When would you use one over the other?**

In Java, the main difference between ArrayList and LinkedList is the way they store and access data. An ArrayList is backed by an array and provides fast random access to elements. [A LinkedList, on the other hand, is backed by a doubly linked list and provides fast insertion and deletion of elements**1**](https://javagoal.com/difference-between-arraylist-and-linkedlist/)[**2**](https://techdifferences.com/difference-between-arraylist-and-linkedlist-in-java.html)[**3**](https://stackoverflow.com/questions/24963620/arraylist-vs-linkedlist-java).

Here are some differences between the two:

**Implementation**: The implementation of both classes is the first difference between ArrayList and LinkedList. The ArrayList class implements the List, Cloneable, RandomAccess, and Serializable interface and extends the AbstractList class.

* **Access**: Accessing elements in an ArrayList is faster than in a LinkedList. This is because an ArrayList is backed by an array, which provides fast random access to elements. [A LinkedList, on the other hand, is backed by a doubly linked list, which requires traversing the list to access elements**1**](https://javagoal.com/difference-between-arraylist-and-linkedlist/).
* **Insertion/Deletion**: Inserting or deleting elements in a LinkedList is faster than in an ArrayList. [This is because a LinkedList only needs to update pointers when inserting or deleting elements, while an ArrayList needs to shift all subsequent elements over by one position**1**](https://javagoal.com/difference-between-arraylist-and-linkedlist/).
* **Memory**: An ArrayList uses more memory than a LinkedList. This is because an ArrayList needs to allocate space for its internal array, even if it’s not full. [A LinkedList, on the other hand, only needs to allocate memory for its nodes**1**](https://javagoal.com/difference-between-arraylist-and-linkedlist/).

When deciding which one to use, you should consider what operations you’ll be performing most frequently. If you need fast random access to elements, use an ArrayList. [If you need fast insertion or deletion of elements, use a LinkedList](https://javagoal.com/difference-between-arraylist-and-linkedlist/)

1. **What is the difference between public, protected, and private access modifiers in Java?**

In Java, access modifiers are used to set the accessibility of classes, methods, and variables. There are four access modifiers in Java: public, protected, private, and default (no modifier). Here are the differences between them:

* **Public**: A public class, method, or variable can be accessed from anywhere in the program. [This means that any other class can access the public member of another class**1**](https://iq.opengenus.org/public-private-protected-in-java/)[**2**](https://stackoverflow.com/questions/215497/what-is-the-difference-between-public-protected-package-private-and-private-in).
* **Protected**: A protected class, method, or variable can be accessed from within the same package or from a subclass in a different package. [This means that only subclasses and classes in the same package can access the protected member**1**](https://iq.opengenus.org/public-private-protected-in-java/)[**2**](https://stackoverflow.com/questions/215497/what-is-the-difference-between-public-protected-package-private-and-private-in).
* **Private**: A private class, method, or variable can only be accessed from within the same class. [This means that no other class can access the private member**1**](https://iq.opengenus.org/public-private-protected-in-java/)[**2**](https://stackoverflow.com/questions/215497/what-is-the-difference-between-public-protected-package-private-and-private-in).
* **Default**: A default class, method, or variable (no modifier) can only be accessed from within the same package. [This means that no other class outside the package can access the default member**1**](https://iq.opengenus.org/public-private-protected-in-java/)[**2**](https://stackoverflow.com/questions/215497/what-is-the-difference-between-public-protected-package-private-and-private-in).

When deciding which access modifier to use, you should consider what level of accessibility you want for your members. If you want a member to be accessible from anywhere in the program, use public. If you want a member to be accessible only within its own class, use private. If you want a member to be accessible within its own package and its subclasses, use protected. [If you don’t specify an access modifier (default), then the member is only accessible within its own package](https://iq.opengenus.org/public-private-protected-in-java/)

1. **What is an exception in Java? Provide an example to explain the concept.**

In Java, an exception is an event that occurs during the execution of a program and disrupts the normal flow of the program’s instructions. It is an object that represents an error or exceptional condition that has occurred in your code. Exceptions are used to handle runtime errors such as ClassNotFoundException, IOException, SQLException, RemoteException, etc

A screenshot of a computer code

Description automatically generated with medium confidence

In this example, we create an array of integers with a length of 5. We then try to assign a value to the 10th element of the array, which doesn’t exist. This will throw an ArrayIndexOutOfBoundsException. We catch this exception using a try-catch block and print out an error message.

* 1. **What is the purpose of the static keyword in Java? Provide an example to explain its use.**

the static keyword is used for memory management mainly. [It can be used with variables, methods, blocks and nested classes](https://www.geeksforgeeks.org/static-keyword-java/)

**A screenshot of a computer program

Description automatically generated with medium confidence**

In this example, the count variable is marked as static. This means that there will only be one instance of the variable shared across all instances of the class. The getCount() method returns the value of the count variable.

* 1. **What is a constructor in Java? Provide an example to explain its use.**

[In Java, a constructor is a special method that is used to initialize objects](https://www.w3schools.com/java/java_constructors.asp).[The constructor is called when an object of a class is created](https://www.w3schools.com/java/java_constructors.asp)

A screenshot of a computer code

Description automatically generated with low confidence

In this example, the ConstructorExample class has a constructor that takes a String parameter called name. When an instance of the class is created, the constructor is called with the value of name. The getName() method returns the value of name.

* 1. **What is the purpose of the interface keyword in Java? Provide an example to explain its use.**

In Java, an interface is a fully abstract class that includes a group of abstract methods (methods without a body). We use the interface keyword to create an interface in Java. Here’s an example:

**A screenshot of a computer program

Description automatically generated with medium confidence**

Here, Language is an interface. It includes abstract methods: getType() and getVersion(). [To use an interface in your class, append the keyword “implements” after your class name followed by the interface name](https://www.programiz.com/java-programming/interfaces).

* 1. **What is a thread in Java? Provide an example to explain the concept.**

In Java, a thread is the path followed when executing a program. A single-threaded application has only one thread and can handle only one task at a time. To handle multiple tasks in parallel, multi-threading is used. [For example, multiple threads are created and each performs a different task](https://www.golinuxcloud.com/thread-in-java-examples/).

**A screen shot of a computer code

Description automatically generated with low confidence**

Here, Multi is a class that extends the Thread class. The run() method is overridden to define the task that the thread will perform. [The start() method is called to start the thread](https://www.javatpoint.com/how-to-create-a-thread-in-java).

* 1. **What is the purpose of the super keyword in Java? Provide an example to explain its use**.

In Java, the super keyword is used to refer to the immediate parent class of a class. [It can be used to access the data members or fields of the parent class if the parent class and child class have the same fields](https://www.javatpoint.com/super-keyword)

A screen shot of a computer code

Description automatically generated with low confidence

In this example, Dog is a subclass of Animal. The Dog class has a field called color, which is different from the field with the same name in its parent class. The printColor() method is used to print the value of both fields. [The super.color statement is used to access the value of the color field in the parent class](https://www.geeksforgeeks.org/super-keyword/).

**15)What is a package in Java? Provide an example to explain the concept.**

[In Java, a package is a mechanism to encapsulate a group of classes, sub-packages, and interfaces](https://www.geeksforgeeks.org/packages-in-java/). [Packages are used for preventing naming conflicts and organizing related classes into a single unit](https://www.geeksforgeeks.org/packages-in-java/)

A screenshot of a computer

Description automatically generated with medium confidence

In this example, we create a package called com.example.mypackage. The MyClass class is part of this package. [The main() method simply prints out the message "This is my package!"](https://www.w3schools.com/java/java_packages.asp)

* 1. **What is a method in Java? Provide an example to explain the concept.**

In Java, a method is a collection of statements that are grouped together to perform an operation[**1**](https://www.tutorialspoint.com/java/java_methods.htm). [When you call a method, the system executes the statements in the method](https://www.tutorialspoint.com/java/java_methods.htm)

A screenshot of a computer

Description automatically generated with medium confidence

[In this example, we create a method called myMethod() which simply prints out the message "Hello World!"](https://www.w3schools.com/java/java_methods.asp). The main() method calls myMethod()

**17) What is the difference between a HashMap and a TreeMap in Java? When would you use one over the other?**

[The main difference between HashMap and TreeMap is that HashMap does not maintain insertion order while the TreeMap does](https://techdifferences.com/difference-between-hashmap-and-treemap-in-java.html).

[HashMap is implemented by Hash Table while TreeMap is implemented by Red-Black tree](https://stackoverflow.com/questions/2444359/what-is-the-difference-between-a-hashmap-and-a-treemap).

[When iterating, TreeMap guarantees the key order which is determined by either element’s compareTo() method or a comparator set in the TreeMap’s constructor](https://stackoverflow.com/questions/2444359/what-is-the-difference-between-a-hashmap-and-a-treemap).

[In terms of time and space efficiency, HashMap is more time-efficient while TreeMap is more space-efficient](https://stackoverflow.com/questions/2444359/what-is-the-difference-between-a-hashmap-and-a-treemap)

[You would use a HashMap when you don’t care about the order of the elements and you want to perform basic operations like get, put and remove in constant time**3**](https://www.geeksforgeeks.org/hashmap-treemap-java/). [You would use a TreeMap when you need to maintain the order of the elements or you need to perform operations like finding the smallest or largest key in logarithmic time](https://www.geeksforgeeks.org/hashmap-treemap-java/)

**18) What is the purpose of the assert keyword in Java? Provide an example to explain its use.**

The assert keyword in Java is used to test assumptions about your program. It is used to check whether a given condition is true or false. If the condition is true, then the program continues to execute normally. If the condition is false, then an error is thrown.

A screenshot of a computer program

Description automatically generated with medium confidence

In this example, we declare an integer variable x and assign it a value of 5. We then use the assert keyword to check whether x is equal to 10. Since x is not equal to 10, an error is thrown with the message “x must be 10” and the program stops executing.

**19)What is the difference between final, finally, and finalize keywords in Java?**

[In Java, final, finally, and finalize are three distinct keywords that serve different purposes](https://www.prepbytes.com/blog/java/final-finally-and-finalize-in-java/).

* [final is used to define a constant variable, a method that cannot be overridden or a class that cannot be extended](https://www.prepbytes.com/blog/java/final-finally-and-finalize-in-java/).
* [finally is used in try-catch blocks to specify a block of code that will be executed regardless of whether an exception is thrown or not](https://www.prepbytes.com/blog/java/final-finally-and-finalize-in-java/).
* [finalize is a method of the Object class that is called by the garbage collector before an object is destroyed**1**](https://www.prepbytes.com/blog/java/final-finally-and-finalize-in-java/).

**20)What is polymorphism in Java? Explain with an example.**

Polymorphism is an important concept of object-oriented programming. It simply means more than one form. [That is, the same entity (method or operator or object) can perform different operations in different scenarios**1**](https://www.programiz.com/java-programming/polymorphism).

[In Java, polymorphism can be achieved in two ways: compile-time polymorphism (static binding) and run-time polymorphism (dynamic binding)**2**](https://www.janbasktraining.com/blog/what-is-polymorphism-in-java/). [Method Overloading is an example of static polymorphism, and Method Overriding is an example of dynamic polymorphism**2**](https://www.janbasktraining.com/blog/what-is-polymorphism-in-java/).

For example, let us consider “Animal” as the parent class and “Dog” is a child class of Parent class. So, any Dog is an animal. Here, the same method “animalSound()” can be used for both the parent and child class. [However, the implementation of this method will be different for both classes](https://www.janbasktraining.com/blog/what-is-polymorphism-in-java/)

**21) Write a program to implement a binary search tree in Java.**

A binary search tree is a special kind of binary tree in which nodes are arranged in a specific order. [In a binary search tree (BST), each node contains only smaller values in its left subtree and only larger values in its right subtree1](https://www.gatevidyalay.com/binary-search-trees-data-structures/).

[Here is an example of constructing a binary search tree using the following sequence of numbers: 50, 70, 60, 20, 90, 10, 40, 100](https://www.gatevidyalay.com/binary-search-trees-data-structures/)

**22) Write a program to implement a doubly-linked list in Java.**

A screenshot of a computer program

Description automatically generated with medium confidence

**23)Write a program to sort an array of objects in Java using the quicksort algorithm.**

Quick sort works recursively in order to sort a given array. These are the three basic steps of the Quicksort algorithm:

1. Partition the array into left and right sub-arrays, in which the items in the left sub-array are smaller than the specified item and the items in the right sub-array are greater than the specified item.  
2. Recursively call the Quicksort to sort the left sub-array.  
3. Recursively call the Quicksort to sort the right sub-array.

The partitioning step is the key when sorting an array with Quicksort. Quicksort itself uses a Partition algorithm to partition the given array.

Here is an example of how to implement the quicksort algorithm in Java for an array of objects:

A screenshot of a computer program

Description automatically generated with medium confidence

A picture containing text, screenshot, font

Description automatically generated

A screenshot of a computer program

Description automatically generated with medium confidence

**24)Write a program to implement a stack using a linked list in Java.**

A screenshot of a computer screen

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

**25)Write a program to implement a queue using two stacks in Java.**

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

**26)Write a program to find the longest common subsequence of two strings in Java.**

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

**27) Write a program to find the maximum subarray sum in an array of integers in Java.**

A screenshot of a computer program

Description automatically generated with medium confidence

**28)Write a program to implement a merge sort algorithm in Java.**

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with medium confidence

**29)Write a program to implement a binary search algorithm for a rotated sorted array in Java.**

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with medium confidence

**30)Write a program to find the shortest path in a weighted graph using Dijkstra's algorithm in Java.**

**Attached:**

**A screenshot of a computer

Description automatically generated**

**31) Create a Page Object Model for the login page of a website. Use Page Factory to initialize the elements. Write a TestNG test to verify successful login with valid credentials.**

**Attached:**

**A screenshot of a computer

Description automatically generated**

The Page Object Model (POM) is a design pattern that is used to create an object repository for web UI elements. It provides an abstraction layer between the test script and the web UI. The Page Factory is a class provided by Selenium WebDriver that is used to initialize the elements of a page object.

Here’s an example of how you can create a POM for the login page of a website and use Page Factory to initialize the elements:

In this example, the LoginPage class has three elements: usernameField, passwordField, and loginButton. The constructor initializes these elements using the PageFactory.initElements() method. The class also has three methods: enterUsername(), enterPassword(), and clickLoginButton(), which are used to interact with the elements.

Here’s an example of how you can write a TestNG test to verify successful login with valid credentials:

A screenshot of a computer program

Description automatically generated with medium confidence

In this example, the setUp() method initializes the WebDriver and LoginPage objects and navigates to the login page. The testLoginWithValidCredentials() method enters valid credentials and clicks the login button. The test then verifies that the user is logged in by checking that the URL contains “dashboard”. Finally, the tearDown() method closes the browser.

32) Create a Page Object Model for the shopping cart page of a website. Use Page Factory to initialize the elements. Write a TestNG test to add a product to the cart, navigate to the cart page, and verify that the product is added to the cart.

Attached:

A screenshot of a computer

Description automatically generated

To create a Page Object Model for the shopping cart page of a website using Page Factory in TestNG, you can follow these steps:

1. Create a new Java class for the shopping cart page and define all the web elements on that page using @FindBy annotations.
2. Create a new Java class for the test case and define all the test steps.
3. In the test case class, create an instance of the shopping cart page class using PageFactory.initElements() method.

A screenshot of a computer

Description automatically generated with medium confidence

1. Use the instance of the shopping cart page class to perform all the actions on that page.

Once you have created an instance of the shopping cart page class using PageFactory.initElements() method, you can use that instance to perform all the actions on that page. For example, you can use the instance to click on a button, enter text in a text box, select an option from a drop-down list, etc.

1. Use TestNG annotations like @Test, @BeforeTest, @AfterTest, etc., to define the test case.

Yes, that’s correct. You can use TestNG annotations like @Test, @BeforeTest, @AfterTest, etc., to define the test case. Here is what each of these annotations does:

* @Test: This annotation is used to mark a method as a test method.
* @BeforeTest: This annotation is used to mark a method that should be run before any test method.
* @AfterTest: This annotation is used to mark a method that should be run after all the test methods have been executed.

**33) Create a Page Object Model for the search page of a website. Use Page Factory to initialize the elements. Write a TestNG test to search for a product, verify that the search results are displayed, and click on a specific product to navigate to its details page.**

**Attached:**

**A screenshot of a computer

Description automatically generated**

 Here’s how you can create a Page Object Model for the search page of a website using Page Factory in TestNG:

A screenshot of a computer

Description automatically generated with medium confidence

1. First, create a new Java class for the search page.
2. In this class, define all the web elements on the search page using the @FindBy annotation provided by Page Factory.
3. Create a constructor method that initializes these web elements using PageFactory.initElements() method.
4. Create methods that perform actions on these web elements like entering text into the search box and clicking on the search button.

Here’s how you can write a TestNG test to search for a product, verify that the search results are displayed, and click on a specific product to navigate to its details page:

1. First, create a new Java class for your test case.
2. In this class, create an instance of your search page class.
3. Use the methods defined in your search page class to enter text into the search box and click on the search button.
4. Use TestNG’s Assert class to verify that the search results are displayed.
5. Use the methods defined in your search page class to click on a specific product and navigate to its details page.

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**MEDIUM**

* 1. **What is the difference between an abstract class and an interface in Java?**

an abstract class is a class that cannot be instantiated and can contain both abstract and non-abstract methods. An interface, on the other hand, is a contract that specifies a set of methods that a class must implement. Here are some differences between the two:

* An interface can only contain method signatures and constants, while an abstract class can contain fields, constructors, and non-abstract methods.
* A class can implement multiple interfaces but can only inherit from one abstract class.
* All methods in an interface are public by default, while an abstract class can have any access modifier for its methods.
* Variables declared in an interface are by default final and static, while an abstract class can have final, non-final, static and non-static variables.
* An interface is used to achieve abstraction and polymorphism in Java, while an abstract class is used to achieve abstraction.

**2) What is the purpose of the static keyword in Java?**

The static keyword in Java is mainly used for memory management. It is used to share the same variable or method of a given class. The users can apply static keywords with variables, methods, blocks, and nested classes. The static keyword belongs to the class than an instance of the class. Here are some examples of how it can be used:

1. A static variable is shared by all instances of the class.
2. A static method belongs to the class and can be called without creating an instance of the class.
3. A static block is used to initialize static variables.
   1. **What is the purpose of the final keyword in Java?**

The final keyword in Java is used to indicate that a variable, method, or class cannot be modified or extended. Here are some of its characteristics:

1. Final variables: When a variable is declared as final, its value cannot be changed once it has been initialized.
2. Final methods: When a method is declared as final, it cannot be overridden by subclasses.
3. Final classes: When a class is declared as final, it cannot be subclassed.

The final keyword is useful when you want a variable to always store the same value or when you want to prevent a method or class from being modified or extended.

* 1. **What is the difference between a checked and an unchecked exception in Java?**

Checked exceptions are checked at compile-time. This means that the compiler will check if the code that might throw a checked exception is handled properly. If it is not handled properly, the code will not compile. Examples of checked exceptions include IOException, SQLException, and ClassNotFoundException.

Unchecked exceptions are not checked at compile-time. This means that the compiler will not check if the code that might throw an unchecked exception is handled properly. Examples of unchecked exceptions include NullPointerException, ArrayIndexOutOfBoundsException, and IllegalArgumentException.

* 1. **What is the difference between a stack and a queue data structure in Java?**

In Java, a stack is a data structure that stores a collection of elements with operations to push (add) and pop (remove) elements from the top of the stack. A stack follows the LIFO (Last In First Out) principle, which means that the last element added to the stack will be the first one to be removed.

On the other hand, a queue is a data structure that stores a collection of elements with operations to enqueue (add) elements at the back of the queue and dequeue (remove) elements from the front of the queue. A queue follows the FIFO (First In First Out) principle, which means that the first element added to the queue will be the first one to be removed.

In summary, the main difference between a stack and a queue is their order of operation. A stack follows LIFO while a queue follows FIFO.

* 1. **What is the purpose of the synchronized keyword in Java?**

In Java, the synchronized keyword is used to control access to shared resources by multiple threads. When a method or block is declared as synchronized, only one thread can execute it at a time. This is useful when multiple threads are accessing the same resource and you want to ensure that only one thread can access it at any given time.

The synchronized keyword works by creating a lock on the object that the method or block is associated with. When a thread enters a synchronized block or method, it acquires the lock and no other thread can enter the block or method until the lock is released.

This helps prevent race conditions and other concurrency issues that can occur when multiple threads are accessing shared resources. It also ensures that changes made by one thread are visible to all other threads.

* 1. **What is the purpose of the volatile keyword in Java?**

In Java, the volatile keyword is used to indicate that a variable’s value may be modified by multiple threads. When a variable is declared as volatile, it ensures that all threads see the most up-to-date value of the variable.

The volatile keyword is used to prevent race conditions that can occur when multiple threads access the same variable. Without the volatile keyword, one thread may not see changes made by another thread, which can lead to unexpected behavior.

When a variable is declared as volatile, it is stored in main memory instead of in a thread’s local cache. This ensures that all threads see the same value of the variable.

* 1. **What is the purpose of the transient keyword in Java?**

In Java, the transient keyword is used to indicate that a variable should not be serialized when an object is written to a file or sent over a network. When an object is serialized, all of its member variables are also serialized. However, sometimes you may not want certain variables to be serialized.

When a variable is declared as transient, it is not included in the serialization process. This can be useful if you have sensitive data that you don’t want to be saved or transmitted.

For example, if you have an object that contains a password field, you may not want that field to be serialized. By declaring the password field as transient, you can ensure that it is not included in the serialization process.

* 1. **What is the difference between the equals() and hashCode() methods in Java?**

In Java, the equals() method is used to compare two objects for equality. The hashCode() method is used to generate a hash code for an object.

The equals() method compares two objects based on their contents. If two objects have the same contents, they are considered equal. The hashCode() method generates a hash code for an object based on its contents. If two objects have the same contents, they will have the same hash code.

The hashCode() method is used by certain data structures, such as HashMaps and HashSets, to quickly look up objects. When you add an object to a HashMap or HashSet, it uses the object’s hash code to determine where to store it. When you look up an object in a HashMap or HashSet, it uses the object’s hash code to quickly find it.

It’s important that if two objects are equal according to their contents (as determined by the equals() method), they must have the same hash code (as determined by the hashCode() method). If two objects have different hash codes but are equal according to their contents, it can cause problems when using data structures that rely on hash codes.

* 1. **What is the purpose of the finalize() method in Java?**

In Java, the finalize() method is used to perform any necessary cleanup when an object is no longer needed. When an object is no longer referenced by any other objects, it becomes eligible for garbage collection. Before it is garbage collected, the JVM calls the object’s finalize() method.

The finalize() method can be used to perform any necessary cleanup, such as closing files or releasing other resources. However, it’s important to note that there are no guarantees as to when or even if the finalize() method will be called. The JVM may not call the finalize() method at all if it determines that there is enough memory available without garbage collecting the object.

Because of this uncertainty, it’s generally not a good idea to rely on the finalize() method for critical cleanup tasks. Instead, you should use try-finally blocks or other mechanisms to ensure that resources are properly released.

* 1. **What is the purpose of the this keyword in Java?**

In Java, the this keyword is used to refer to the current object. It can be used to access instance variables and methods of the current object.

For example, if you have a class with an instance variable called name, you can use the this keyword to refer to that variable:

A screenshot of a computer program

Description automatically generated with medium confidence

In this example, the constructor takes a parameter called name and assigns it to the instance variable with the same name. However, because the parameter has the same name as the instance variable, we need to use the this keyword to refer to the instance variable.

* 1. **What is the difference between a superclass and a subclass in Java?**

In Java, a superclass is a class that another class (the subclass) inherits from. The subclass inherits all of the fields and methods of the superclass.

A picture containing text, screenshot, document, font

Description automatically generated

In this example, Animal is the superclass and Dog is the subclass. Dog inherits all of the fields and methods of Animal, including the constructor that takes a name parameter. Dog also has its own method called bark().

* 1. **What is the purpose of the package keyword in Java?**

In Java, a package is a way to organize related classes and interfaces. The package keyword is used at the beginning of a source file to specify which package the class or interface belongs to.

For example:

A screenshot of a computer

Description automatically generated with medium confidence

In this example, MyClass belongs to the com.example package.

Packages are used to avoid naming conflicts between classes and interfaces. By organizing related classes and interfaces into packages, you can avoid naming conflicts with classes and interfaces from other packages.

Packages are also used to control access to classes and interfaces. By default, classes and interfaces are only accessible within their own package. However, you can use access modifiers (such as public, private, protected) to control access to classes and interfaces within a package.

* 1. **What is the difference between an instance variable and a class variable in Java?**

In Java, an instance variable is a variable that belongs to an instance of a class. Each instance of the class has its own copy of the instance variable.

A picture containing text, screenshot, font

Description automatically generated

In this example, Person has two instance variables: name and age. Each instance of Person has its own copy of these variables.

A class variable (also called a static variable) is a variable that belongs to the class itself, rather than to any particular instance of the class. There is only one copy of a class variable, regardless of how many instances of the class exist.

For example:

A picture containing text, screenshot, font

Description automatically generated

In this example, Person has one class variable: count. Each time a new instance of Person is created, count is incremented. The getCount() method returns the current value of count.

* 1. **What is the purpose of the instance of operator in Java?**

The instanceof operator in Java is used to test whether an object is an instance of a particular class or not. Its syntax is:

objectName instanceof className;

Copy

Here, if objectName is an instance of className, the operator returns true. Otherwise, it returns false.

The instanceof operator is useful when you want to check whether an object is of a particular type before performing some operation on it. For example, you might want to check whether an object is an instance of a particular subclass before casting it to that subclass.

* 1. **What is the purpose of the super keyword in Java?**

The super keyword in Java is used to refer to the parent class of a subclass. Here are some of its characteristics:

* super is used to call a superclass constructor: When a subclass is created, its constructor must call the constructor of its parent class. This is done using the super() keyword, which calls the constructor of the parent class.
* super can be used to call a method in the superclass: If a subclass has a method with the same name as a method in its parent class, you can use the super keyword to call the method in the parent class.
* super can be used to access a member variable in the superclass: If a subclass has a member variable with the same name as a member variable in its parent class, you can use the super keyword to access the member variable in the parent class.
  1. **What is the purpose of the abstract keyword in Java?**

In Java, the abstract keyword is used to define abstract classes and methods. Here are some of its key characteristics:

* Abstract classes cannot be instantiated: An abstract class is a class that cannot be instantiated directly. Instead, it is meant to be extended by other classes, which can provide concrete implementations of its abstract methods.
* Abstract methods do not have a body: An abstract method is a method that does not have a body. Instead, it is meant to be implemented by subclasses of the abstract class.
* Abstract classes can have non-abstract methods: An abstract class can have both abstract and non-abstract methods. Non-abstract methods are implemented in the same way as they are in a regular class.

Overall, the abstract keyword is a powerful tool for defining abstract classes and methods in Java. By declaring a class or method as abstract, developers can provide a structure for subclassing and ensure that certain methods are implemented in a consistent way across all subclasses.

* 1. **What is the purpose of the interface keyword in Java?**

In Java, the interface keyword is used to define an interface. An interface is a collection of abstract methods that are defined but not implemented. Here are some of its key characteristics:

* Interfaces cannot be instantiated: An interface is a reference type that cannot be instantiated directly. Instead, it is meant to be implemented by classes.
* Interfaces can have constants: An interface can have constants that are implicitly public, static, and final.
* Interfaces can have default methods: A default method is a method that has a body in the interface. It is meant to provide a default implementation for the method.
* Interfaces can have static methods: A static method is a method that has the static modifier in the interface. It is meant to provide a utility method that can be called without creating an instance of the class.
* A class can implement multiple interfaces: A class can implement multiple interfaces by separating them with commas.

Overall, the interface keyword is a powerful tool for defining interfaces in Java. By declaring an interface, developers can provide a structure for implementing classes and ensure that certain methods are implemented in a consistent way across all classes.

* 1. **What is the difference between a private and a protected access modifier in Java?**

In Java, the private and protected access modifiers are used to control access to class members. Here are some of their key characteristics:

* private: A private member can only be accessed within the same class. It cannot be accessed from outside the class, even by subclasses.
* protected: A protected member can be accessed within the same class, within subclasses in the same package, and within subclasses in different packages.

Overall, the difference between private and protected is that private members are only accessible within the same class, while protected members are accessible within subclasses as well.

* 1. **What is the purpose of the throws keyword in Java?**

In Java, the throws keyword is used to declare that a method may throw an exception. Here are some of its key characteristics:

* throws: The throws keyword is used in the method signature to declare that a method may throw one or more exceptions. If a method throws an exception that is not declared in its signature, it will result in a compile-time error.
* Checked exceptions: Exceptions that are checked at compile time are called checked exceptions. These exceptions must be declared in the method signature using the throws keyword.
* Unchecked exceptions: Exceptions that are not checked at compile time are called unchecked exceptions. These exceptions do not need to be declared in the method signature using the throws keyword.

Overall, the throws keyword is a powerful tool for handling exceptions in Java. By declaring that a method may throw an exception, developers can provide a structure for handling errors and ensure that certain methods are implemented in a consistent way across all classes.

**10 Java Programming Questions:**

* 1. **Write a Java program to find the factorial of a number using recursion.**

Here’s an example Java program that uses recursion to find the factorial of a number.

A screen shot of a computer program

Description automatically generated with low confidence

In this program, the factorial method is called recursively until the base case is reached (when n == 0). The base case is when the recursion stops and the method returns a value.

* 1. **Write a Java program to check if a given number is a palindrome or not.**

Ans: Here’s an example Java program that checks if a given number is a palindrome or not:

A picture containing text, screenshot, font, number

Description automatically generated

In this program, the isPalindrome method checks if the given number is a palindrome or not. It does this by reversing the number and comparing it to the original number. If they are equal, then the number is a palindrome.

* 1. **Write a Java program to check if a given string is a palindrome or not.**

Here’s an example Java program that checks if a given string is a palindrome or not:

A screenshot of a computer code

Description automatically generated with low confidence

In this program, the isPalindrome method checks if the given string is a palindrome or not. It does this by comparing the characters at the beginning and end of the string. If they are equal, then it moves on to the next pair of characters until it reaches the middle of the string.

* 1. **Write a Java program to reverse a given string without using any built-in functions.**

Here’s an example Java program that reverses a given string without using any built-in functions:

A screen shot of a computer program

Description automatically generated with low confidence

In this program, the reverseString method reverses the given string by swapping the characters at the beginning and end of the string. It does this by converting the string to a character array, then using two pointers to swap the characters until they meet in the middle.

* 1. **Write a Java program to find the second highest number in an array.**

Here’s an example Java program that finds the second highest number in an array:

A picture containing text, screenshot, font

Description automatically generated

In this program, the findSecondHighest method finds the second highest number in the given array by iterating through the array and keeping track of the two highest numbers. It does this by comparing each number to the current highest and second highest numbers, and updating them accordingly.

* 1. **Write a Java program to implement bubble sort algorithm.**

 Here is a Java program to implement bubble sort algorithm:

A picture containing text, screenshot, font, number

Description automatically generated

* 1. **Write a Java program to implement selection sort algorithm.**

 Here is a Java program to implement selection sort algorithm:

A picture containing text, screenshot, font

Description automatically generated

This program sorts an array of integers using the selection sort algorithm. In selection sort algorithm, we search for the lowest element and arrange it to the proper location. [We swap the current element with the next lowest number](https://www.javatpoint.com/selection-sort-in-java).

* 1. **Write a Java program to implement insertion sort algorithm.**

Here is an example Java program to implement insertion sort algorithm:

A screenshot of a computer program

Description automatically generated

**­**

* 1. **Write a Java program to implement binary search algorithm.**

Binary search is a search algorithm that finds the position of a target value within a sorted array. It works by repeatedly dividing in half the portion of the list that could contain the item, until you’ve narrowed down the possible locations to just one. The algorithm is efficient for large lists because its time complexity is O(log n).

Here’s an example of how it works in Java:

A white screen with black text

Description automatically generated

In this example, we have an array of integers and we want to find the index of a specific integer. We start by setting two pointers: one at the beginning of the array and one at the end. We then calculate the middle index and compare its value to our target integer. If it’s equal, we return that index. If it’s less than our target integer, we move our low pointer to the middle index + 1. If it’s greater than our target integer, we move our high pointer to the middle index - 1. We repeat this process until we find our target integer or determine that it doesn’t exist in the array.

* 1. **Write a Java program to count the number of words in a given string.**

 Here’s an example of how to count the number of words in a string in Java:

A screenshot of a computer code

Description automatically generated

In this example, we have a string and we want to count the number of words in it. We start by splitting the string into an array of words using the split() method and passing in a regular expression that matches one or more whitespace characters. We then return the length of the resulting array.

**3 Selenium Framework Questions:**

* 1. Create a Page Object Model for **the registration page** of a website. Use Page Factory to initialize the elements. Write a TestNG test to verify successful registration with valid credentials.
* Attached:
* A screenshot of a computer

  Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

* 1. Create a Page Object Model for the **checkout page of a website**. Use Page Factory to initialize the elements. Write a TestNG test to add a product to the cart, navigate to the checkout page, and verify that the correct product is being purchased.

Attached:

A screenshot of a computer

Description automatically generated

* 1. Create a Page Object Model for **the home page of a website**. Use Page Factory to initialize the elements. Write a TestNG test to navigate to the home page, verify the presence of certain elements, and click on a link to navigate to a different page.

Attached:

A screenshot of a computer

Description automatically generated

----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**EASY**

* 1. **What is the purpose of the public keyword in Java?**

The public keyword is an access modifier used for classes, attributes, methods and constructors in Java. [It makes them accessible by any other class **1**](https://www.w3schools.com/java/ref_keyword_public.asp).

When a class is declared as public, it can be accessed from anywhere in the program. [If a method is declared as public, it can be accessed from anywhere in the program](https://www.javatpoint.com/public-keyword-in-java)

* 1. **What is the purpose of the private keyword in Java?**

The private keyword is an access modifier used for classes, attributes, methods and constructors in Java. It makes them accessible only within the same class .

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

* 1. **What is the purpose of the protected keyword in Java?**

The protected keyword is an access modifier used for classes, attributes, methods and constructors in Java. It makes them accessible within the same package or subclasses in different packages .

When a class is declared as protected, it can be accessed within the same package or subclasses in different packages. If a method is declared as protected, it can be accessed within the same package or subclasses in different packages .

* 1. **What is the purpose of the static keyword in Java?**

The static keyword is used to create variables and methods that will exist independently of any instances created for the class.

When a variable is declared as static, it can be accessed without creating an instance of the class. If a method is declared as static, it can be called without creating an instance of the class .

* 1. **What is the purpose of the final keyword in Java?**

The final keyword is used to create constants.

When a variable is declared as final, it cannot be changed after it has been initialized. If a method is declared as final, it cannot be overridden by subclasses .

* 1. **What is the purpose of the abstract keyword in Java?**

The abstract keyword is used to create abstract classes and methods.

When a class is declared as abstract, it cannot be instantiated. If a method is declared as abstract, it does not have a body and must be implemented by subclasses .

* 1. **What is the purpose of the interface keyword in Java?**

The interface keyword is used to create interfaces.

An interface is a collection of abstract methods and constants that can be implemented by classes.

* 1. **What is the purpose of the new keyword in Java?**

The new keyword is used to create an instance of a class.

When an object is created using the new keyword, memory is allocated for the object and the constructor of the class is called to initialize the object .

* 1. **What is the difference between a class and an object in Java?**

A class is a blueprint or template for creating objects. It defines the properties and methods that an object will have.

An object is an instance of a class. It is created using the new keyword and has its own set of properties and methods .

* 1. **What is the purpose of the main() method in Java?**

The main() method is the entry point of a Java program.

When a Java program is executed, the JVM (Java Virtual Machine) looks for the main() method in the class that is specified as the starting point of the program.

The main() method must be declared as public, static, and void.

* 1. **What is the difference between a float and a double in Java?**

In Java, float and double are both used to represent floating-point numbers.

The main difference between them is their precision. A float is a single-precision 32-bit floating-point number, while a double is a double-precision 64-bit floating-point number.

A double can represent more decimal places than a float and is therefore more precise.

* 1. **What is the difference between an int and a long in Java?**

In Java, int and long are both used to represent integer values.

The main difference between them is their size. An int is a 32-bit signed integer, while a long is a 64-bit signed integer.

A long can represent larger values than an int and is therefore useful when dealing with large numbers.

* 1. **What is the purpose of the System.out.println() method in Java?**

The System.out.println() method is used to print a string to the console.

The string that is passed as an argument to the method will be printed to the console followed by a newline character.

This method is commonly used for debugging purposes and for displaying output to the user.

* 1. **What is the purpose of the Scanner class in Java?**

The Scanner class is used to read input from the user.

It provides methods for reading different types of input, such as integers, floating-point numbers, and strings.

The Scanner class is commonly used for console input and for reading data from files.

* 1. **What is the purpose of the Math class in Java?**

The Math class provides methods for performing mathematical operations in Java.

It includes methods for performing basic arithmetic operations, such as addition, subtraction, multiplication, and division.

It also includes methods for performing more complex operations, such as calculating the square root of a number or generating a random number.

* 1. **What is the purpose of the String class in Java?**

The String class is used to represent strings of characters in Java.

It provides methods for manipulating strings, such as concatenating two strings together or finding the length of a string.

Strings are used extensively in Java programs for storing and manipulating text data.

* 1. **What is the purpose of the StringBuilder class in Java?**

The StringBuilder class is used to create and manipulate strings in Java.

It provides methods for appending, inserting, and deleting characters from a string.

The StringBuilder class is similar to the String class, but it is more efficient when making multiple modifications to a string.

* 1. **What is the purpose of the equals() method in Java?**

The equals() method is used to compare two objects for equality in Java.

It is defined in the Object class and can be overridden by subclasses to provide custom equality comparisons.

The equals() method returns true if the two objects being compared are equal, and false otherwise.

* 1. **What is the purpose of the compareTo() method in Java?**

The compareTo() method is used to compare two objects in Java.

It is defined in the Comparable interface and is used to provide a natural ordering for objects.

The compareTo() method returns a negative integer if the object being compared is less than the other object, zero if they are equal, and a positive integer if the object being compared is greater than the other object.

* 1. **What is the purpose of the toString() method in Java?**

The toString() method is used to convert an object to a string in Java.

It is defined in the Object class and can be overridden by subclasses to provide custom string representations of objects.

The toString() method is commonly used for debugging purposes and for displaying output to the user.

* 1. **10 Java Programming Questions:**
  2. **Write a Java program to print "Hello, World!" to the console.**

Here’s a simple Java program that prints “Hello, World!” to the console:

A screenshot of a computer program

Description automatically generated

* 1. **Write a Java program to find the sum of two numbers.**

**A screenshot of a computer program

Description automatically generated**

This program prompts the user to enter two numbers and then calculates their sum.

* 1. **Write a Java program to find the average of three numbers.**

Here’s a simple Java program that finds the average of three numbers:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

This program prompts the user to enter three numbers and then calculates their average.

* 1. **Write a Java program to check if a given number is even or odd.**

Here’s a simple Java program that checks if a given number is even or odd:

A screenshot of a computer program

Description automatically generated

This program prompts the user to enter a number and then checks if it is even or odd.

* 1. **Write a Java program to check if a given number is prime or not.**

**A screenshot of a computer program

Description automatically generated**

Here’s a simple Java program that checks if a given number is prime or not:

This program prompts the user to enter a number and then checks if it is prime or not.

* 1. **Write a Java program to check if a given string is a palindrome or not.**

Here’s a simple Java program that checks if a given string is a palindrome or not:

A screenshot of a computer program

Description automatically generated

This program prompts the user to enter a string and then checks if it is a palindrome or not.

* 1. **Write a Java program to reverse a given string.**

Here’s a simple Java program that reverses a given string:

A screenshot of a computer program

Description automatically generated

This program prompts the user to enter a string and then reverses it.

* 1. **Write a Java program to implement a simple calculator.**

Here’s a simple Java program that reverses a given string:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

This program prompts the user to enter a string and then reverses it.

* 1. **Write a Java program to convert Fahrenheit to Celsius.**

Here is a Java program that converts Fahrenheit to Celsius:

A screenshot of a computer

Description automatically generated

This program prompts the user to enter temperature in Fahrenheit and then converts it into Celsius using the formula (fahrenheit - 32) \* 5 / 9. The result is then displayed on the console.

* 1. **Write a Java program to generate a random number between 1 and 100.**

Here is a Java program that generates a random number between 1 and 100:

A screenshot of a computer program

Description automatically generated

This program uses the Random class from the java.util package to generate a random number between 1 and 100. The nextInt() method of the Random class returns a random integer between 0 (inclusive) and the specified value (exclusive). In this case, we add 1 to the result of nextInt(100) to get a random integer between 1 and 100.

* 1. **3 Selenium Framework Questions:**

1. **Write a TestNG test to navigate to a website and verify the title of the page.**

**Attached:**

**A screenshot of a computer

Description automatically generated**

Here is a TestNG test that navigates to a website and verifies the title of the page:

A screenshot of a computer program

Description automatically generated

This test uses the Selenium WebDriver to navigate to the website https://www.example.com and then verifies that the title of the page is “Example Domain”. The @BeforeTest method sets up the WebDriver instance, the @Test method navigates to the website and verifies the title, and the @AfterTest method tears down the WebDriver instance.

Note that you will need to download the appropriate version of the ChromeDriver executable and set its path in the System.setProperty() method.

1. **Write a TestNG test to fill out a login form on a website and verify successful login with valid credentials.**

**Attached:**

**A screenshot of a computer

Description automatically generated**

Here is a TestNG test that fills out a login form on a website and verifies successful login with valid credentials:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

This test uses the Selenium WebDriver to navigate to the login page of the website https://www.example.com/login and then fills out the username and password fields with valid credentials. It then clicks the login button and verifies that the URL of the page has changed to the dashboard URL.

Note that you will need to download the appropriate version of the ChromeDriver executable and set its path in the System.setProperty() method.

1. **Write a TestNG test to navigate to a website, click on a link to navigate to a different page, and verify the presence of certain elements on the new page.**

**Attached:**

**A screenshot of a computer

Description automatically generated**

Here is a TestNG test that navigates to a website, clicks on a link to navigate to a different page, and verifies the presence of certain elements on the new page:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

This test uses the Selenium WebDriver to navigate to the website https://www.example.com and then clicks on a link with the text “Click here”. It then verifies that the title of the new page is “New Page Title” and that certain elements with IDs “element1”, “element2”, and “element3” are displayed on the page.

Note that we will need to download the appropriate version of the ChromeDriver executable and set its path in the System.setProperty() method.