Walmart Interview Questions

Part-1

cheat sheet

Top 25 questions with answers

Prepared by: Abdulhakim Edao

1. Can you explain the difference between an interface and an abstract class in Java?

In Java, an interface is a collection of abstract methods that must be implemented by a class, while an abstract class is a class that can contain both abstract and concrete methods. An interface can be implemented by multiple classes, but an abstract class can only be extended by a single class. This means that an abstract class can provide a common implementation for some of its methods, while an interface cannot.

2. Can you describe the process of object-oriented programming in Java?

Object-oriented programming (OOP) is a programming paradigm that is based on the concept of "objects", which are data structures that contain both data and methods. In Java, objects are created from classes, which define their behavior and state. OOP allows for the creation of modular, reusable code, and makes it easier to organize and maintain complex programs.

3. Can you explain the use of the keyword "static" in Java?

In Java, the keyword "static" can be used to modify a method or a variable. A static method belongs to the class itself, rather than to a specific object, and can be called without creating an instance of the class. A static variable is

also associated with the class, rather than with a specific object, and is shared among all instances of the class.

4. Can you describe the difference between a local variable and an instance variable in Java?

In Java, a local variable is a variable that is declared within the body of a method. Local variables are only visible within the method in which they are declared, and they are only accessible within the block of code in which they are defined. In contrast, an instance variable is a variable that is declared within a class, but outside of any method. Instance variables are associated with an instance of a class, and they are accessible to all methods of the class.

5. Can you explain the concept of polymorphism in Java?

Polymorphism in Java is the ability of a single variable or reference to take on multiple forms. This is achieved by allowing an object to be treated as an instance of a parent class, while also allowing it to be treated as an instance of one or more child classes. This allows for greater flexibility and code reuse.

6. Can you describe the difference between a stack and a queue in Java?

A stack in Java is a data structure that allows for the storage and retrieval of data in a Last-In-First-Out (LIFO) manner. This means that the last element

added to the stack will be the first element to be removed. In contrast, a queue in Java is a data structure that allows for the storage and retrieval of data in a First-In-First-Out (FIFO) manner. This means that the first element added to the queue will be the first element to be removed.

7. Can you explain the concept of exception handling in Java?

Exception handling is a mechanism in Java that allows the program to handle errors or exceptional events that may occur during the execution of the program. When an error or exceptional event occurs, an object representing the error or exceptional event is created and thrown, which can then be caught and handled by a corresponding catch block in the code. This allows the program to continue running and avoid crashing.

8. Can you describe the process of creating and using a thread in Java?

To create a thread in Java, you need to create a class that extends the **Thread** class and overrides the **run()** method. Then, you can create an instance of that class and call the **start()** method to start the thread. To use a thread, you can create a new thread and call the **start()** method, which will cause the **run()** method to be executed in a separate thread.

9. Can you explain the difference between a shallow copy and a deep copy in Java?

A shallow copy in Java creates a new object and copies the values of the object's fields to the new object. However, if the fields of the original object are references to other objects, then only the references are copied, not the objects themselves. This means that the new object and the original object will refer to the same objects, rather than having independent copies. A deep copy, on the other hand, creates a new object and recursively copies all of the objects referenced by the original object, so that the new object has independent copies of all of the objects.

10. Can you describe the use of the "finally" block in Java?

The **finally** block in Java is a block of code that is always executed after the **try** and **catch** blocks, regardless of whether an exception was thrown or not. This is useful for performing cleanup tasks, such as closing open files or releasing resources that were allocated in the **try** block. The **finally** block is guaranteed to be executed, even if an exception is thrown in the **try** or **catch** blocks, or if the thread is interrupted or killed.

11. Can you explain the concept of encapsulation in Java?

Encapsulation in Java is the mechanism that allows the programmer to restrict access to certain components of a class. This is achieved by declaring the fields of the class as private, and providing public methods that allow the fields to be accessed or modified. This allows the programmer to control how

the fields are accessed and modified, and to ensure that the class is used correctly.

12. Can you describe the difference between an array and a linked list in Java?

An array in Java is a data structure that allows the programmer to store a fixed-size sequential collection of elements of the same type. Arrays are indexed, which means that the programmer can access individual elements of the array using an integer index. Arrays are static, which means that their size cannot be changed after they are created.

A linked list in Java, on the other hand, is a data structure that consists of a sequence of nodes, each containing a reference to the next node in the sequence. Linked lists are dynamic, which means that their size can be changed at runtime by adding or removing nodes. Linked lists are not indexed, which means that the programmer must traverse the list to access a specific node.

13. Can you explain the concept of inheritance in Java?

Inheritance in Java is the mechanism that allows a class to inherit properties and behaviors from another class. This is achieved by defining a subclass that extends a superclass. The subclass inherits all of the fields and methods of the superclass, and can also define its own fields and methods. This allows

the programmer to reuse the code defined in the superclass, and to create a hierarchy of classes that share common characteristics.

14. Can you describe the process of overloading and overriding methods in Java?

Method overloading in Java is the mechanism that allows a class to have multiple methods with the same name, but with different parameter lists. This allows the programmer to provide multiple versions of a method that can be called with different sets of arguments. Method overriding, on the other hand, is the mechanism that allows a subclass to provide its own implementation of a method that is defined in the superclass. This allows the subclass to override the behavior of the method and provide its own implementation.

15. Can you explain the use of the "this" keyword in Java?

In Java, the "this" keyword is used to refer to the current object or instance of a class. It is typically used to access member variables or methods of the current object from within the class, as in "this.variable" or "this.method()".

16. Can you describe the difference between a TreeSet and a HashSet in Java?

TreeSet and HashSet are both implementations of the Set interface in Java. A Set is a collection that cannot contain duplicate elements.

The main difference between TreeSet and HashSet is that TreeSet is an ordered set, whereas HashSet is unordered. This means that the elements in a TreeSet are sorted according to their natural ordering, or according to a custom Comparator that you can specify when you create the set. In contrast, the elements in a HashSet are not sorted at all.

Additionally, TreeSet provides some additional methods for working with ordered sets, such as first(), last(), and subSet(). HashSet, on the other hand, is generally faster than TreeSet because it uses a hashing algorithm to store and retrieve elements, while TreeSet uses a binary search tree.

17. Can you explain the concept of generics in Java?

Generics in Java is a mechanism for creating a type-safe class or method that can operate on a variety of different types, while still providing compile-time type checking to ensure that the types used are correct. This allows developers to create code that is more reusable and flexible, while still maintaining the benefits of strong type checking.

18. Can you describe the process of creating and using a JavaFX GUI?

To create a JavaFX GUI, you will need to use the JavaFX library and associated tools. This typically involves creating a new JavaFX project in your preferred development environment, and then designing the GUI using a combination of Java code and XML-based markup. Once the GUI is designed,

you can use the JavaFX library to add event handling and other functionality to make the GUI interactive.

19. Can you explain the difference between a "break" statement and a "continue" statement in Java?

In Java, a "break" statement is used to immediately exit a loop, switch statement, or labeled statement. This can be useful for breaking out of a loop early, or for skipping to the next iteration of the loop. In contrast, a "continue" statement is used to skip the current iteration of a loop and move on to the next iteration. This can be useful for skipping over certain elements or conditions in a loop.

20. Can you describe the use of the "try-with-resources" statement in Java?

The "try-with-resources" statement is a new feature in Java that allows you to automatically manage the lifecycle of a resource, such as a file or database connection, within a try block. This eliminates the need to manually close the resource in a finally block, which can improve the readability and reliability of your code. To use the try-with-resources statement, you simply declare the resource within the parentheses following the try keyword, and the resource will be automatically closed at the end of the try block.

21. Can you explain the concept of Lambda expressions in Java?

Lambda expressions are a new feature in Java 8 that allow you to create anonymous functions, or functions that are not bound to an identifier. This allows you to write more concise, flexible, and functional code, by using lambda expressions to pass behavior as an argument to a method, or to return a result from a method. Lambda expressions can be used in conjunction with functional interfaces, which are interfaces that have a single abstract method, to create function objects that can be used in a variety of contexts.

22. Can you describe the difference between a HashMap and a TreeMap in Java?

A HashMap is a data structure that stores elements in a map, using a hashing function to determine the location of each element based on its key. This allows for fast insertion and lookup times, but does not provide any guarantees about the order of the elements in the map. In contrast, a TreeMap is a data structure that stores elements in a map, sorted according to their natural ordering or a comparator that is provided at the time the map is created. This allows for fast insertion and lookup times, as well as guarantees about the order of the elements in the map.

23. Can you explain the concept of functional interfaces in Java?

A functional interface in Java is an interface that has a single abstract method, known as a functional method. This allows the interface to be used

as a lambda expression, or as the target type of a method reference. Functional interfaces are often used in conjunction with lambda expressions and method references to create function objects that can be passed as arguments to methods, or returned as results from methods.

24. Can you describe the process of creating and using a Stream in Java?

To create a Stream in Java, you can use the **Stream** class or one of its subclasses, such as **IntStream** or **LongStream**. The Stream class provides a range of methods for generating streams, such as **of()**, **iterate()**, and **generate()**, as well as methods for transforming and operating on streams, such as **map()**, **filter()**, and **reduce()**. To use a Stream, you can create a stream from a source, such as a collection or an array, and then apply one or more operations to the stream to transform or process the elements in the stream.

25. Can you explain the difference between a parallel stream and a sequential stream in Java?

A parallel stream is a type of stream that is capable of executing operations on the elements of the stream in parallel, using multiple threads. This can improve the performance of certain types of operations, particularly those that are CPU-intensive, by allowing the operation to be divided across multiple threads and processed simultaneously. In contrast, a sequential stream is a type of stream that executes operations on the elements of the

stream in a single thread, in the order in which the elements appear in the stream. This can be useful for operations that must be performed in a specific order, or that have dependencies between elements in the stream.