Interview Questions

Basics of JAVA

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

1. What is JAVA?

Java is a simple, class-based, platform independent programming language developed by sun microsystems. Java is used to develop standalone, web, enterprise, and mobile application. Java is based on oops concept and the basic principle of oops concept are Inheritance, Polymorphism, Abstraction, and Encapsulation. Some of the features of java are:

2. Features of JAVA?

Some of the features of java are listed below:

1. Simple

Java is a simple programming language which is very easy to write and simple to understand. The syntax of java is easy as compared to other programming language.

1. Platform Independent

Java is a platform independent language, once the code is compiled the byte code is generated and the generated byte code can be used in any platform/operating system i.e. windows, mac, Linux. So, java is also called as write once and run anywhere programming language.

1. Robust

Java has a strong memory management. Being a programmer, we don't have to worry about allocations and deallocation of memory. Java already has automatic garage collector which runs in JVM, and it get rid of objects that are not being used in java application.

3. What are the Different types of applications you can build with JAVA?

a. Standalone Application

Standalone application is the desktop application or windows-based application that needs to be installed in every machine. Examples are anti-virus, music player.

b. Web application

Those application that runs on a server side and generates a dynamic page is called as we application. JSP, Servlet, Hibernate, Spring technologies are used to develop web application.

c. Enterprise application

Enterprise applications are those application that are distributed in nature and satisfies the need of an organization. We use spring technologies to develop enterprise application.

d. Mobile application

An application which is created for mobile devices is called as mobile application. Android are used for creating mobile devices.

4. What is path and class path?

Path is an environment variable that is used by the operating systems to find the executable files.

Class path is an environment variable that is used by the compiler to load or locate the path.

5. Difference between JDK, JRE and JVM?

JDK is a java development toolkit which is used to develop application. It contains all set of libraries, and other tools like debugger, compiler, document generator to develop application. Without JDK also we can run the program but if we want to develop an application then we should have JDK.

JRE is a runtime environment which provides all the liabraries and files that JVM needs at run time. JRE is platform independent. Operating System cannot directly access JVM, so jre communicates with operating system and create JVM before the execution of the program.

JVM is a virtual machine that converts bytecode into machine language. JVM is a part of JRE. JVM gets created at the time of execution. When it is created it loads, verifies, executes the logic and after executing the logic JVM disappears.

6. Explain JAVA memory model?

**Heap memory**: this memory is allocated for objects and instance variables.

**Stack memory**: is allocated for methods and local variables. (only when method is invoked/called, memory for it is allocated and after it finish executing that method, the memory for those variables and memory gets deallocated)

**Method Area**: Method area is a shared resource where the class level information is stored. The memory for the static variable is allocated in the method area. **Classloader loads the class into the memory. Class loader loads the class in the method area.**

**String constant pool:** is a special memory area used for string objects only.

**Program counter Register** holds the address of currently executing instruction when thread is started, and each thread has its own PC register.

7. Different data types in JAVA?

a. Primitive data types

Primitive Data type are the datatypes that are predefined by java, and it has no additional methods and they are: boolean, char, byte, short, int, long, float, double.

b. Non-primitive data types

Non-primitive data type is created by programmer and is not defined by Java (except for String) and they can be used to call methods to perform certain operations.

8. Explain the design of JAVA class?

While designing a java class, a class should have well defined package name, import statement, meaningful class name, and inside class there should de variable initialization, method declaration and comments wherever necessary.

9. Different types of operators?

They are:

Assignment Operator, Logical Operator, Arithmetic Operator, Relational Operator, Unary Operator.

10. Top 5 operators?

a. Relational Operators

* **==, Equal to:** returns true if left hand side is equal to right hand side.
* **!=, Not Equal to :**returns true if left hand side is not equal to right hand side.
* **<, less than:**returns true if left hand side is less than right hand side.
* **<=, less than or equal to :**returns true if left hand side is less than or equal to right hand side.
* **>, Greater than :**returns true if left hand side is greater than right hand side.
* **>=, Greater than or equal to:**returns true if left hand side is greater than or equal to right hand side.

b. Arithmetic Operators

* **\* :**Multiplication
* **/ :**Division
* **% :**Modulo
* **+ :**Addition
* **– :**Subtraction

c. Assignment Operators

**‘=’** Assignment operator is used to assign a value to any variable.

1. **Logical Operator**

**&&, Logical AND :**returns true when both conditions are true.

**||, Logical OR :**returns true if at least one condition is true.

1. Unary Operator

Unary operators need only one operand. They are used to increment, decrement or negate a value.

11. Explain how each control statement works?

(if-else, for, while, do-while, ternary, switch, break and continue)

Control statements can be a decision-making statement or looping statements or branching statements.

Decision making statements includes if else, switch statements that determines which statement to execute and when.

When a statement executes a block of repeated code until a specific condition is met are known as looping statements. They include while, for, do-while loops.

Branching statements jump from a statement to another statements. Break and continue are called branching statements.

OOPS

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

12. What is Class and Object?

A class is a user defined template that represents state and behavior where state means properties and behavior means methods. Class is designed to perform a part of functionality of program and a class has well defined package name, import statement, meaningful class name, and inside class there is variable declaration, method declaration and comments wherever necessary. Class is used to create an object.

An object is an instance of class that has state and behavior. Object is used to call any method or to perform operation.

Every time and object is created by using new keyword.

13. What are different OOPS concepts?

Different types of OOPS concepts are:

* 1. Abstraction
  2. Polymorphism
  3. Inheritance
  4. Encapsulation

14. What is Abstraction?

Abstraction is one of the important concepts of Object-Oriented Programming language. Abstraction is defined as the process of hiding of implementation details and showing only the functionality to the user. Abstraction can be achieved using abstract classes or interfaces.

1. Abstract Class

A class that is declared with abstract keyword is known as abstract class. It can have abstract as well as non-abstract method. When the abstract class is extended by sub class all the method of abstract class must be defined and implemented. The abstract class cannot be instantiated.

1. Interface

Interface is an abstract type which is used to provide specification to the class so that class can implement those methods. An interface is declared by using the interface keyword. Interface is used to achieve abstraction and multiple inheritance in java. The methods declared in interface are by default abstract it means that in interface there should only be method signature without the body. Interface methods are by default abstract and public. Attributes are by default public, static and final.

15. What is Encapsulation?

Encapsulation is one of the core concepts of OOP in java. It is defined as the act of wrapping up of properties and behaviors under one single unit. In Encapsulation properties and behaviors are bound and cannot be accessed directly from other classes. Encapsulation is achieved by declaring a variable private and creating public getter and setter methods to set and get the values from other classes.

16. What is the difference between abstraction and encapsulation?

|  |  |
| --- | --- |
| Abstraction | Encapsulation |
| 1. Abstraction is achieved by abstract class and interface. | 1. Encapsulation is achieved by declaring a variable private and creating public getter and setter. |
| 2. Abstraction is defined as the process of hiding all the implementation details and showing only the functionality to the user. | 2. Encapsulation is the act of wrapping up all the behavior and properties into a one single unit. |
| 3. It solves an issue at the design level. | 3. Encapsulation solves an issue at implementation level. |
|  |  |
|  |  |

17. What is Inheritance?

Inheritance is one of the important concepts of OOP in java. Inheritance represents IS-A relationship which is also called as parent child relationship in which child class can inherit the properties and behaviors of parent class. Inheritance allows us to create classes that are built in existing classes and in that class, we can specify our own implementation which helps in code reusability. In Inheritance we use extends keyword to extend the functionality of parent class to child class.

18. What is Polymorphism?

Polymorphism is an important concept of OOP in java. It refers to the ability of an object to take multiple forms or performing a single action in a different way. Polymorphism is achieved in Java with the help of method overloading and overriding.

19. What is method overloading and method overriding?

Method overloading is a concept in polymorphism where a method has different return types and different method signature. It is also called as compile time polymorphism. Method overloading is used to add more to the behavior of methods and there is no need of more than one class for method overloading. We can achieve overloading in single class also.

Method overriding is a concept in polymorphism where a method has same or co-variant return type and same method signature. It is also called as run time polymorphism. Method overriding is used to change the behavior of existing methods and there is a need for at least two classes for method overriding where a child class will override parent class method and provide its own implementation.

20. What is static in JAVA?

Static is a keyword in java which is applicable for variable, methods, nested class, and initialization block.

When a variable is declared as static it is called as static variable. For static variables memory is allocated only once during class loading and the value will be common for all the instances.

When a method is static, we do not need to create an object.

Static block is a block of a code in java that will be executed when the class is first loaded in JVM. Static block is mostly used for the configuration purpose (database connection) to make everything ready before the start of the program.

21. What is final?

Final is a keyword in java which can be used in variables, methods, and class. When a variable is declared as final, once the value is initialized it cannot be modified. When a method is final, it cannot be overridden, when a class is declared as final it cannot be extended to sub class.

22. What is final, finally, finalize?

Final is a keyword in java which can be used in variables, methods, and class. When a variable is declared as final, once the value is initialized it cannot be modified. When a method is final, we cannot override that method in the child class, when a class is declared as final, we can’t extend this class functionality by creating child class.

Finally, is a block in java that goes along with try catch or try block. Inside finally block we write resource deallocation code or clean up code like connection close. The specialty of finally, block is it is executed whether the exception has occurred or not, handled or not.

Finalize is a method associated with garbage collector concept. Garbage collector will call Finalize method to perform clean up activities just before destroying an object. Once finalize method completed cleanup process, automatically garbage collector will destroy the object.

23. What is static and dynamic binding?

Binding is called as connecting the method call with method body. Basically it is of two types: static binding and dynamic binding.

The binding that takes place at compile time is called as static binding. Java compiler binds method calls with method definition/body during compilation. Method overloading is an example of static binding.

The binding that takes place at run time is called as dynamic binding. JVM binds the method call with method definition at runtime and invokes the relevant method during

runtime when the method is called. Method overriding is an example of dynamic binding.

24. Abstract class vs interface?

|  |  |
| --- | --- |
| Abstract class | Interface |
| 1. Abstract class can have abstract and non-abstract methods. | 1.Inteface can have abstract methods and default methods. |
| 2. Abstract class does not support multiple inheritance. | 2.Interface supports multiple inheritance. |
| 3.The abstract keyword is used to declare abstract class. | 3.The interface keyword is used to declare interface. |
| 4.Abstract class can provide the implementation of interface. | 4.Interface can’t provide the implementation of abstract class. |
| 5.Abstract class can have final, non-final, static, non-static variables. | 5.Interface can have only final and static variables. |
|  |  |

25. Which one you choose between abstract and interface?

26. Why do create abstract classes in application development?

Abstract class are the class that can have abstract as well as non-abstract methods. Abstract method is method which has method declaration but not implementation and is implemented by the extended class. So, abstract class provides the flexibility of the code in the future, whenever developer is not sure of the implementation, they can create an abstract class and later whenever necessary they can make implementation of those methods.

27. why do create interfaces in application development?

28. What are different types of access modifiers?

There are four types of access modifiers in java i.e. default, private, protected and public.

When there is no access modifier it is called default access modifier. If data members are declared as default access modifier, then they are accessible only within packages.

If a data member is declared as private its, they can be accessed only within class.

If a data member is declared as public, then its data member can access from anywhere.

If a data member is declared as protected, then they can be accessed anywhere within the same package and outside the package they can be accessed through child package.

29. What is call by value?

Call by value means calling a method with parameter as value and call by references means calling a method with parameter as reference.

30. What is HAS-A relation?

Has a relation is known as composition in java. It basically means that an instance of a one class has a reference to an instance of another class.

String

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

31. What is String in JAVA?

String is an object that represents a sequence of characters. Java String provides a lot of methods to perform operations and some of them are equals (), compare (), length(), split(), index(), to Lowercase(), to Uppercase(). String is immutable which means that once the object is created, we are not allowed to make any changes, if we try to modify or make any changes instead new object will be created. If the content is not fixed, keep on changing it is never recommended to use String because every time a new object will be created internally for any changes and as a result performance will be down and memory problem will be created. And if the content is never going to change at run-time Like: city, college name, state throughout the content and application then we should go for String.

32. What is String pool?

String pool is a separate memory area in java heap memory. Whenever we create a string literal, JVM first checks if the literal object is already present in the pool. If the literal is present, then it returns reference of that literal object. Otherwise JVM will create new string object and return the reference of that string object.

33. Why String is immutable?

String is immutable because once the object is created it cannot be changed or modified, if we try to change or modify the object then new object will be created. Majority of data in java are represented in the form of string, there are very high chance for string to have more than one object reference. If we try to change the value of string object using one of the object references, the value will be changed for all the references pointing towards the same object. That is why java made string immutable. So, in this case if we try to change the value of a string object using one of the object references, it will create a new object and that object reference is pointed towards the new object which we created, and others will remain unaffected.

34. How do you make a class immutable?

We can make a class immutable by declaring a class as final, variables as final and not using setter method.

35. StringBuilder vs StringBuffer?

|  |  |
| --- | --- |
| String Buffer | String Builder |
| 1.) Most of the methods present inside StringBuffer are synchronized. | 1.) Most of the methods present in StringBuilder are non-synchronized. |
| 2.) At a time only one thread is allowed to operate on StringBuffer object and hence it is thread safe. | 2.) At a time, multiple thread is allowed to operate on StringBuilder object and hence it is not thread safe. |
| 3.) Threads are required to wait to operate on StringBuffer object and hence relatively performance is slow. So, StringBuffer is less efficient than StringBuilder. | 3.) Threads are not required to wait to operate on StringBuilder object and hence relatively performance is high. So, StringBuilder is less efficient than StringBuffer. |
| 4.) StringBuffer was introduced in 1.0 Version. | 4.) StringBuilder was introduced in 1.5 Version. |
| 5.) If the content is not fixed and keep on changing but thread safety is required and at a time only one thread is allowed, then we should go for String Buffer. | 5.) If the content is not fixed and keep on changing and thread safety is not required, multiple thread is allowed to operate simultaneously then we should go for StringBuilder. |

36. == vs equals?

|  |  |
| --- | --- |
| == | .equals |
| 1. It is an operator. | 1. It is a method. |
| 2. == operator is used for reference comparison. | 2. For object class equals method is used for reference comparison, but for string class equals method is used for content comparison. |
|  |  |
|  |  |

37. Top 5 methods in String class?

Length(), charAt(), equals(),substring(), split().

Exception Handling

38. What is Exception Handling and How do you achieve it?

Exception is an event that disrupts the normal flow of the program. The process of handling these exceptions so that the flow of the program does not break is called Exception Handling.

Exception Handling helps us to prevent failure of the program. If a program has a bunch of statements and exception occurred in midway after executing certain statement, then the statements after the exception will not be executed unless we handle that exception.

One way of achieving exception handling is to use try-catch block. In try-catch block, we can put the code that we want to run in try block. Any exception that the code might caught can be kept in one or multiple catch block. We cannot use try block alone. Try block should be followed by either catch or finally block.

39. How do you make sure a code must be executed even if exception happens?

By placing the code in the finally block.

40. What code you normally write in finally block?

Finally, block is associated with try or catch block. In finally block we usually write most important code of the program like closing connections and statements. The main purpose of the finally block is to maintain clean up code. The specialty of finally block is that the code written in finally block are always executed whether the exception rise or not, or exception is handled or not.

41. What are checked vs unchecked exceptions?

The exceptions which are checked by compiler for smooth execution of the program at the runtime are called as checked exceptions.

In the case of checked exceptions compiler will check whether we are handling exception if the programmer is not handling then we will get the compile time error. If a checked exception is thrown, we must handle the exception with try-catch keyword or specify the exception using throws keyword. CE: unreported exception java.io.FileNotFoundExceptiom, must be caught or declared to be thrown.

The exceptions which are not checked by the compiler is called as unchecked exception. Unchecked exceptions are not forced by the compiler to either handle or specify the exception. It is up to the programmer whether they want to specify or catch the exception. All the exceptions under RuntimeException and Error are unchecked exceptions.

Whether exception is checked or unchecked compulsory it will occur only at runtime, there is no chance of occurring any exception at compile time.

Runtime exception and its child classes, error and its child classes are unchecked except this remaining are checked exceptions.

|  |  |
| --- | --- |
| Checked Exception | Unchecked Exception |
| 1. The exception which is checked by the compiler for the smooth execution of program is called as checked exception. | 1. The exception which are not checked by the compiler is called as unchecked exception. |
| 2.We will get compile time error if we didn’t handle checked exception. | 2. Compiler does not produce any error if we handle the exception or not. |
| Example: fileNotFound, IOException. | Exemple: ArrayIndexOutOfBound, NullPointerException. |

42. How do you create custom Exception?

Custom Exception is a user defined exceptions in java which is required to develop meaningful exceptions based on the application requirements. These exceptions are related to business logic that is useful for the application user or developers to understand the exact problem and a string. We can create our own custom exception by extending the Exception class in java and we must have a constructor that takes string as error message. After creating a custom exception, we can simply throw a custom exception wherever it is needed. Some of the examples are InsufficientFundsExeception, InvalidAgeException, LowScoreException.

43. How does exception propagation works?

Exception propagation occurs when an exception is thrown from the top of the stack. When it is not caught, it drops down the call stack to the previous method. If it is not caught there, it drops down again to the previous method. This continues unless the method reaches the bottom of the call stack or is caught somewhere in between. If the exception is caught the program will execute normally and if exception is not caught, then the program will disrupt abnormally.

44. throw vs throws

The differences between throw and throws keyword are:

Throws keyword is used to declare an exception whereas throw keyword is used to explicitly throw an exception.

Throw keyword is used inside a function whereas throws keyword is in the method signature.

Throw keyword can throw one exception at a time whereas throws keyword can be used to declare multiple exception.

Throw keyword cannot propagate checked exceptions. It is only used to propagate the unchecked exception that are not checked using throws keyword. Throws keyword is used to propagate the checked exceptions only.

45. Exception vs Error?

|  |  |
| --- | --- |
| Exception | Error |
| Exceptions in java are of type java.lang.Exception. | Errors in java are of type of java.lang.Error. |
| Exceptions include both checked as well as unchecked type. | All errors in java are unchecked type. |
| Checked exceptions are known to compiler whereas unchecked exceptions are not known to compiler because they occur at run time. | Errors happens at run time. They will not be known to compiler. |
| You can recover from exceptions by handling them through try-catch blocks. | It is impossible to recover from errors. |
| Exceptions are caused by the application itself. | Errors are mostly caused by the environment where the application is running. |
| Examples:  Checked Exceptions : SQLException, IOException  Unchecked Exceptions: ArrayIndexOutOfBoundException, ClassCastException, NullPointerException. | Examples:  java.lang.StackOverflowError,  java.lang.OutOfMemoryError. |

46. What are inner classes?

The class that can be written within the class is called as inner class.

Text, application

Description automatically generated

47. What is Anonymous class?

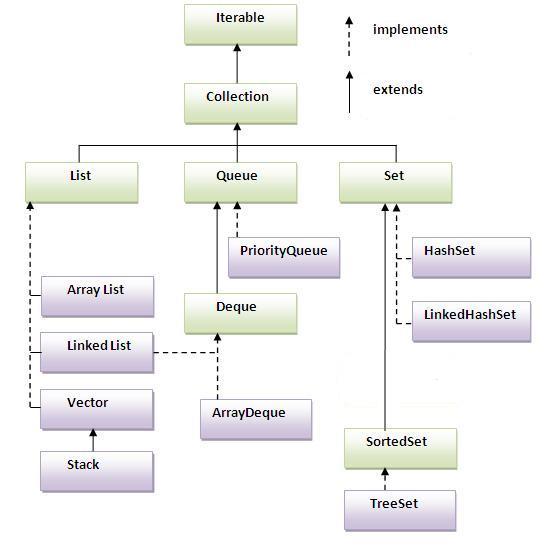
An anonymous class in Java is a class not given a name and is both declared and instantiated in a single statement. You should consider using an anonymous class whenever you need to create a class that will be instantiated only once.

1. New
2. Active
3. Blocked / Waiting
4. Timed Waiting
5. Terminated

Collections

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

48. JAVA Collection framework tree?



49. ArrayList vs LinkedList?

|  |  |
| --- | --- |
| ArrayList | LinkedList |
| 1.The underlying data structure is dynamic array. | 1. The underlying data structure is doubly linked list. |
| 2. ArrayList is best choice if our frequent operation is storing and retrieval because it implements Random Access Interface so we can access random element at the same speed. | 2. LinkedList is not the good choice for retrieval because to retrieve an element we must traverse from beginning or end to reach that element so, it will be time consuming, and performance will be low.  2. LinkedList is the best choice if our frequent operation is insertion and deletion from the middle. |
| 3. ArrayList is better for storing and accessing data. | 3. LinkedList is better for manipulating data. |
|  |  |
|  |  |
| 2. Elements are stored on index basis in ArrayList. | 2. Elements are stored in node based in ArrayList. |

47. ArrayList vs Set?

|  |  |
| --- | --- |
| Arraylist | Set |
| 1.Arraylist is the child class that implements List interface. | 1. Set is the interface in the collection. |
| 2. In ArrayList insertion order is preserved. | 2. In Set insertion order is not preserved. |
| 3. Duplications are allowed in ArrayList. | 3. Duplications are not allowed in Set. |
|  |  |

50. Which one you prefer between ArrayList vs LinkedList ?

The selection between ArrayList and LinkedList depends on the requirement. If our frequent operation is retrieval then we should go for ArrayList because arraylist implements Random Access Interface so, we can access random element at the same speed, but it is not better choice if our frequent operation is insertion and deletion in the middle because arraylist internally uses an array so, if any elements are added or removed from the array, all the bits are shifted in memory.

Talking about LinkedList, it is the best choice if our frequent operation is insertion and deletion in the middle, because LinkedList is based on node-based concept and it internally uses doubly linkedlist to store data so, if we want to perform delete operation, the node will be deleted and then the address will point towards the previous node so, bit shifting is not required.

51. How do you sort collection of elements in JAVA ?

To sort collection of elements in Java we can use Collections.sort() method provided by Collection class. The elements must implement comparable interface and put the comparison logic of two elements in the compareTo() method to sort them.

52. What is Collections class in JAVA?

Collections are the utility class in java that is found in java.util package. It defines several methods like: sorting and searching which is used to operate on collection. It has a static method.

Collection is a framework in java, if we want to represent group of individual objects as a single entity then we should go for collection. Collection contains the collection of classes and interfaces which can be used to perform various operations like: insertion, deletion, searching, sorting and manipulation of data. Collection provide various interface List, Set, Queue, dequeue. This interfaces had there own implementation classes like arraylist, LinkedList, HashSet, LinkedHash Set, PriortyQueue and so on.

Some of the features of collection are:

1. Both homogeneous and Heterogeneous element are accepted.
2. Collection are resizable in nature, based on our requirement we can increase and decrease the size.
3. If we want to represent a group of individual object as single entity then we should go for collection.
4. In collection each class has their own implementation and underlying data structure. So, being a programmer, we can just make use of those classes and methods.

53. HashTable vs HashMap?

|  |  |
| --- | --- |
| HashTable | HashMap |
| 1. HashTable does not allow any null key or value. | 1. Hashmap allows one null key and multiple null values. |
| 2. Hashtable is a legacy class. | 2. Hashmap is a new class introduced in JDK 1.2. |
| 3.Hashtable is slow. | 3. HashMap is fast. |
| 4. Hashtable is synchronized. It is thread-safe and can be shared with many threads. | 4. HashMap is non synchronized. It is not thread safe and can’t be shared between many threads without proper synchronization code. |
| 5. Hashtable is traversed by enumerator and iterator. | 5. HashMap is traversed by Iterator. |
|  |  |

54. What is HashMap?

Hashmap is one of the implementation of map interface that is based on key value pair principle. The important operation that we perform in hashmap are : put and get. Hashmap allows multiple null values but almost one null key.

55. How does HashMap Works?

HashMap is one of the implementations of map interface that works on the key value-based concept. The two important operations that we do in HashMap are: put and get. When we are using put method to insert value based on key in the hashmap. At first it checks whether the key is null or not, if the key is null the key value bucket is placed at the 0th index of map entry table. If the key is not null hashcode of the key is generated by calling hashCode() method and with that hashcode, index is calculated and key value bucket is placed in a map entry table. If there is already existed key-value bucket in that index, hashCode collision will occur, meaning that two key generated the same hashcode. In that case equals method will be used to check the key, if the key is different than the key value bucket will be attached as a linked list in the same index, but if the key is same then the old value will be replaced by the new value.

 Similarly, when we want to get a value for a given key from hashmap, with the help of key, hashcode is generated and index is calculated and using equals method key is checked, if the key is same then the value will be retrieved.

56. What is HashMap collison?

Whenever we are storing key value pair using put method in hashmap, if two keys produce same hashcode and point towards the same index position hashmap collision will occur. To resolve this problem equals method will be used to check if the key for those hashcode is same or not, if the key is same the old value will be replaced by new one and if the key is different the key value bucket will be attached as a linked list in the same index.

57. What is ConcurrentHashMap?

· The underlined data structure for ConcurrentHashMap is Hashtable. It implements Serializable, ConcurrentMap<K, V>, Map<K, V> interfaces and extends AbstractMap<K, ​V> class.

· ConcurrentHashMap class is thread-safe i.e. multiple threads can operate on a single object without any complications.

· In ConcurrentHashMap, the Object is divided into a number of segments according to the concurrency level.

· The default concurrency-level of ConcurrentHashMap is 16.

· In ConcurrentHashMap, at a time any number of threads can perform retrieval operation but for updated in the object, the thread must lock the particular segment in which the thread wants to operate. This type of locking mechanism is known as Segment locking or bucket locking. Hence at a time, 16 update operations can be performed by threads.

· Inserting null objects is not possible in ConcurrentHashMap as a key or value.

58. Why do you think Strings makes approriage keys for HashMap?

In the case if we used a certain variable as a key and to store key-value pair. In case we modify the key, at the time of retrieval since as we have changed the key the hash code of the current key will not match with the hash code where the value has been stored which makes impossible to retrieve the value.

Since string object are immutable. Immutability allows to get the same hash code every time as we cannot modify the value of the string once it is created. So, it is the appropriate keys for hashmap.

 59. What kind of classes are good for HashMap keys?

Immutable classes like String and wrapper class are good for hashmap keys. As immutability allows us to get same hash code every time, for a key object and it will be easy to retrieve the value. So, it actually solves most of the problems.

60. What is Iterator vs ListIterator?

|  |  |
| --- | --- |
| Iterator | Listiterator |
| 1. Iterator is called as unidirectional because we can traverse only in forward direction. | 1.ListIterator is called as bidirectional because using listiterator we can move either in forward direction or in backward direction. |
| 2. Using iterator we cannot add any element to the collection. | 2. Using Listiterator we can add element to the collection. |
| 3. We cannot obtain indexes while using iterator. | 3. We can obtain indexes at any point of time while traversing a list using listiterator. The methods nextIndex() and previousIndex() are used for this purpose. |
| 4. We cannot replace the existing element value while using Iterator. | 4. By using set method of ListIterator we can replace the last element returned by next() or previous () methods. |
| 5. Iterator is used to transverse both list and set. | 5. ListIterator is used to transverse only list. |

61. Comparable vs Comparator interfaces?

|  |  |
| --- | --- |
| Comparable | Comparator |
| 1. Comparable interface is meant for default natural sorting. | 1. Comparator interface is meant for customized sorting order. |
| 2. Comparable provides single sorting sequence.  In other words, we can sort the collection on the single element such as id or price. | 2. Comparator provides multiple sorting sequence.  In other words, we can sort the collection in the basis of multiple elements such as id, name and price etc. |
| 3. Compare is found in java.lang package. | 3. Comparator is found in java.util package. |
| 4. Comparable provides compareTo() method to sort elements. | 4. Comparator provides compare() and equal q method to sort elements. |
| A |  |

62. Treeset and TreeMap?

|  |  |
| --- | --- |
| Treeset | TreeMap |
| 1.Treeset is an implementation of set interface. | 1. Treeset is an implementation of set interface. |
| 2. Treeset stores one object. | 2. TreeMap stores two object key value. |
| 3. Treeset does not allow null values. | 3. TreeMap can have multiple null values and only one null key. |
| 4.TreeSet is sorted based on object. | 4.TreeMap is sorted based on key. |
|  |  |

63. what is contract between hashcode and equals?

If two objects are equal, then they must have the same hash code. But if two objects have the same hash code, they may or may not be equal. If you override one method you must override the other method too.

63. Why do you need override Hashcode and equals?,

Multi-Threading

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

64. What is multithreading?

Multithreading is a feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU. Each part of such program is called a thread.

Thread:

65. How do you create Threads?

Thread can be created by two ways.

1.) By Extending the thread class.

2.) Implementing the runnable interface.

67. How do you synchronize your code?

Synchronization is defined as the capability to control the access of multiple threads to any shared resource. We can use synchronized block to synchronize the code. Basically, synchronized block allows thread executing at a time. All other threads attempting to enter the synchronized block are blocked until the thread inside the synchronized block exists the block.

68. What is volatile?

Volatile keyword is used to modify the value of a variable by different threads. It is also used to make classes thread safe. It means that multiple threads can use a method and instance of the classes at the same time without any problem. It uses value from the main memory instead of local memory.

69. What is race condition?

A race condition is a situation when two or more threads can access shared data and they try to change it at the same time. Suppose we have two threads Sometime first thread executes first and sometime other get execute first and we get different result every time. This condition is called race condition. We can use synchronization to restrict race condition as it control the access of multiple threads to any shared resource.

1. Read-modify-act

a. Two or more threads first read a given variable, then modify its value and write it back to the variable.

2. Check-then-act

1. Two or more threads check a given condition and go to act based on that information.

70. What is dead lock?

Deadlock is a situation where two threads are blocked forever, waiting for each other. Deadlock can occur in a situation when a thread is waiting for an object lock, that is acquired by another thread and second thread is waiting for an object lock that is acquired by the first thread. Since both threads are waiting for each other to release the lock, the condition is called deadlock.

71. What is Thread Local?

The ThreadLocal class in Java enables you to create variables that can only be read and written by the same thread. In thread local the value changed by using one thread cannot be seen by other thread.

ThreadLocal variables are special kinds of variables created and provided by the Java ThreadLocal class. These variables are only allowed to be read and written by the same thread. Two threads cannot be able to see each other’s ThreadLocal variable, so even if they will execute the same code, then there won't be any race condition and the code will be thread-safe.

72. What is Thread Pool?

A thread pool is a group of threads initially created that waits for jobs and executes them. The idea is to have the threads always existing, so that we won't have to pay overhead time for creating them every time. They are appropriate when we know there's a stream of jobs to process, even though there could be some time when there are no jobs.

73. How do you use Executor framework?

Executors framework (java.util.concurrent.Executor), released with the JDK 5 in package java.util.concurrent is used to run the Runnable objects without creating new threads every time and mostly re-using the already created threads.

The Executor framework allows us to execute tasks asynchronously. The ExecutorService helps in maintaining a pool of threads and assigns them tasks.

74. What is starvation?

Starvation is a situation where a thread is unable to gain regular access to shared resources and is unable to make progress.

**Starvation** occurs when a thread is continually denied access to resources and as a result it is unable to make progress. This usually happens when greedy threads consume shared resources for long periods of time.

75. Synchronized methods vs Synchronized blocks?

Synchronized Method: In this method, the thread acquires a lock on the object when they enter the synchronized method and releases the lock either normally or by throwing an exception when they leave the method.

Synchronized Block: In this method, the thread acquires a lock on the object between parentheses after the synchronized keyword and releases the lock when they leave the block.

Synchronized blocks should be preferred more as it boosts the performance of a particular program. It only locks a certain part of the program (critical section) rather than the entire method and therefore leads to less contention.

The key difference is this: if you declare a method to be synchronized, then the entire body of the method becomes synchronized; if you use the synchronized block, however, then you can surround just the "critical section" of the method in the synchronized block, while leaving the rest of the method out of the block.

If the entire method is part of the critical section, then there effectively is no difference. If that is not the case, then you should use a synchronized block around just the critical section. The more statements you have in a synchronized block, the less overall parallelism you get, so you want to keep those to the minimum.

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

Miscellaneous

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

76. What is serialization and externalization?

Serialization in Java is a mechanism of writing the state of an object into byte stream which helps to transport the object. It is mainly used in technologies like Hibernate, JPA, EJB

The reverse operation of serialization is called deserialization where byte-stream is converted into an object.

Externalization provides custom serialization, where we can manage our object stream and decide what to store in it. Externalization is useful if we want to serialize a part of an object. We can serialize only the required fieds of an object.

We can externalize an object using writeExternal() method.

77. How do you avoid a variable participating in serialization?

We can avoid a variable participating in serialization by marking that variable as either static or transient.

transient variable in Java is a variable whose value is not serialized during Serialization and which is initialized by its default value during de-serialization, for example for object transient variable it would be null.

78. What is marker interface?

A marker interface is an interface that has no methods or constants inside it. It provides run-time information about objects, so the compiler and JVM have additional information about the object.

Serializable and Cloneable are marker interface.

79. What is Enum in JAVA?

A enum is a special “class” that represents a group of constants (unchangeable variables like final variables). To create enum, we use the enum keyword and separate the constants with a comma.