**Multithreading-**

**1.What Is Multithreading.**

-Multithreading is a specialized form of multitasking. Multithreaded programs contains two or more parts are running concurrently for maximum utilization of CPU. Each part is known as thread.

**2. What is thread?**

-Thread is a light weight process. Thread can be created by two ways:

(i) By extending Thread class.

(ii) By implementing Runnable Interface

**3. What does join() method.**

-Entry point of the thread called after start().

**4. What is difference between wait() and sleep() method?**

|  |  |
| --- | --- |
| **Wait()** | **Start()** |
| 1. It is used to pause the execution of thread for some period of time. | 1. If a Thread needs to wait for any resource it calls wait method. |
| 1. It is of Thread class. | 1. It is a Object class |
| 1. It has two overloaded forms -   public static void sleep(long milliseconds)  public static void sleep(long milliseconds, int nanoseconds) | 1. It has three overloaded form.   public final void wait()  public final void wait(long milliseconds)  public final void wait(long milliseconds, int nanoseconds) |
| 1. It is class method. | 1. It is a instance method and is final. |
| 1. When time period expires thread comes out of sleep state. | 1. If a thread calls wait(), then it will come out of waiting state when another thread send notification by calling notify() or notifyAll()   If a thread calls wait([param]) , then it will wait for notification only until time get expires, after that it will come out of waiting state. |
| 1. It is not needed for interthread communication.. | 1. It is needed in interthread communication. |

**5. Is it possible to start a thread twice?**

-No , After starting a thread , it can never be start again. If you does so, an IllegalThreadException is thrown. In such a case , thread will run once but second time ,it will thrown exception.

**6. Can we call the run() method instead of start()?**

-No, you cannot call run method to start a thread. You need to call start method to call to create new thread .If you call run method directly ,it won’t create a new thread and it will be in a same stack as main.

Each thread starts in a separate call stack. Invoking a run method from main thread, the run method goes inside current call stack rather than a beginning of new call stack.

**7. What about the daemon threads?**

Daemon thread is a low priority thread that’s run in the background to perform tasks such as garbage collection. Its priority ranges to 10, when garbage collector is required. Its life depends on user thread.

**8. Can we make the user thread as daemon thread if thread is started?**

-No, if we call setDaemon () method after Thread.start() it would throw IllegalThreadException. setDaemon() is called before starting of thread.

**9. What is the purpose of Synchronized block**

-Synchronized block is used to lock an object for any shared resource.

-Scope of synchronized block is smaller than the method.

**10. What is static synchronization?**

-Static synchronized method will lock the class instead of instead of object and it will lock the class because the keyword static means “class instead of object “. The keyword synchronized means that only one thread access the method at a time.

**11. What is the difference between notify() and notifyAll()?**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.No.** | **Key** | **Notify()** | **NotifyAll()** |
| 1. | Notification | In case of multithreading notify() method sends the notification to only one thread among the multiple waiting threads which are waiting for lock. | While notifyAll() methods in the same context sends the notification to all waiting threads instead of single one thread. |
| 2. | Thread Identification | As in case of notify the notification is sent to single thread among the multiple waiting threads so it is sure that which of those waiting thread is going to receive the lock. | On other hand notifyAll sends notification to all waiting threads hence it is not clear which of the thread is going to receive the lock. |
| 3. | Risk factor | In case of notify() method the risk of thread missing is high as notification is sent only single thread and if it misses that than no other thread would get notification and hence the lock. | While in case of notifyAll as notification is to all the waiting threads and hence if any thread misses the notification, there are other threads to do the job. Hence risk is less. |
| 4. | Performance | Memory and CPU drain is less as compare to notifyAll as notification is sent to single one thread so performance is better as compare to notifyAll. | On other hand as the cost of no notification is dropped and notification is sent to all waiting threads the memory and CPU drain is more as compare to notify and hence performance of notifyAll is lesser. |
| 5. | Interchangeable | In case of the notify() method as only single one thread is in picture hence no concept of thread Interchangeable is possible. | While we should go for notifyAll() if all your waiting threads are interchangeable (the order they wake up doesn’t  matter). |

**12. What is deadlock?**

-Deadlock is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process. ... Similar situation occurs in operating systems when there are two or more processes hold some resources and wait for resources held by others.

**13.Difference between process and thread?**

|  |  |  |
| --- | --- | --- |
| **Comparison Basis** | **Process** | **Thread** |
| Definition | A process is a program under execution i.e.an active program. | A thread is a lightweight process that can be managed independently by a scheduler. |
| Context switching time | Processes require more time for context switching as they are more heavy. | Threads require less time for context switching as they are lighter than processes. |
| Memory Sharing | Processes are totally independent and don’t share memory. | A thread may share some memory with its peer threads. |
| Communication | Communication between processes requires more time than between threads. | Communication between threads requires less time than between processes . |
| Blocked | If a process gets blocked, remaining processes can continue execution. | If a user level thread gets blocked, all of its peer threads also get blocked. |
| Resource Consumption | Processes require more resources than threads. | Threads generally need less resources than processes. |
| Dependency | Individual processes are independent of each other. | Threads are parts of a process and so are dependent. |
| Data and Code sharing | Processes have independent data and code segments. | A thread shares the data segment, code segment, files etc. with its peer threads. |
| Treatment by OS | All the different processes are treated separately by the operating system. | All user level peer threads are treated as a single task by the operating system. |
| Time for creation | Processes require more time for creation. | Threads require less time for creation. |
| Time for termination | Processes require more time for termination. | Threads require less time for termination. |

**14. Thread class methods:**

|  |  |
| --- | --- |
| **Method** | **Meaning** |
| getName | Obtain thread’s name |
| getPriority | Obtain thread’s priority |
| isAlive | Determine if a thread is still running |
| Join | Wait for a thread to terminate |
| Run | Entry point for the thread |
| Sleep | Suspend a thread for a period of time |
| Start | Start a thread by calling its run method |

**Basic Questions:**

1. **What is a transient variable?**

Transient variable 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.

1. **Why do threads block on I/O?**

Thread gets block just because when two thread try to access the single resource simultaneously the thread get blocked and when one thread is using the resource other should be put to sleep state because when the thread which is processing produces the result only then the resource is allowed to another thread.

1. **What is synchronization and why is it important?**

Synchronization is to achieve the lock on resource. When two or more threads to access same shared resource at a time, then race condition will arise and resource may go to inconsistent state. Thus synchronization is to achieve the lock on resource. The thread having lock on the resource only use that resource at a time. All threads others thread should wait for that resource until the thread holding the resources releases the lock.

1. **Is null a keyword?**

In java, null is reserved word (keyword) for literals values. It seems like a keyword, but actually, it is a literal similar to true and false.

-It is case sensitive.

-It is a value of reference variable.

-The access to null references generates a “NullPointerException”.

-It is not allowed to pass null as a value to call the methods that contains any primitive data type.

1. **What is the Vector class?**

-Vector class is a legacy class.

-Implements a growable array of objects.

-Elements of Vector can be accessed using an integer index. However, the size of a Vector class can grow or shrink as needed to accommodate adding and removing items after the Vector has been created.

-It is synchronized.

Way to create Vector class object:

Vector vec = new Vector(); it creates empty Vector with default capacity of 10

Vector object = new Vector(int initialCapacity);

Vector vec = new Vector(int initialCapacity, int capacityIncrement)

1. **What is the difference between the >> and >>> operators.**

>> is a signed right shift operator and >>> is a unsigned right shift operator.

>>> right shifts and fill with 0 at the left end , while >> filled with the sign bit.

1. **Can a double value be cast to a byte?**

- You cannot cast double value to byte because byte range is smaller than the double and it does not contain decimals like a double does.

1. **What is the difference between a break statement and a continue statement?**

Both the statements break and continue works with in loop. The main difference is continue statement escapes the remaining statements inside the loop and move to next iteration while break statement causes the flow to exit the loop.

1. **What value does read() return when it has reached the end of a file?**

-The read() method returns -1 when it reached the end of a file.

1. **Which class is extended by all other classes?**

-The Object class extended by all other classes.

1. **Does a class inherit the constructors of its super class?**

* No, a subclass cannot inherit the constructor of its super class. Constructor is special function member of a class in that they are not inherited by its sub class. Constructors are used to give valid state for an object at creation.

1. **What is the purpose of the System class?**

-Facilities provided by System class are standard input, standard output and error output stream, access to extremely defined properties and environment variables, a means of loading files and libraries , and a utility methods for copying a portion of an array.

1. **What modifiers may be used with an interface declaration?**

-The keywords public and abstract can appear as a modifiers at the beginning of an interface declaration. In this situation ,these modifiers have the following meaning :

public- if an interface declared as public , it can be referenced by any class and interface.

1. **What is the difference between a while statement and a do statement?**

|  |  |  |
| --- | --- | --- |
| **Basis for comparisons** | **While** | **Do while** |
| General form | While(condition){  Statements;//body of the loop  } | do{  -  Statements;//body of the loop  -  }while(condition); |
| Controlling condition | In while loop controlling condition appear at the start of the loop | In do while loop controlling condition appear at the end of the loop |
| Iterations | The iterations do not occur if, the condition at the first iteration appear false | The iteration occur at least once even if the condition is false at the first iteration |
| Alternate name | Entry – controlled loop | Exit-controlled loop |
| Semi-colon | Not used | Used at the end of the loop |

1. **What modifiers can be used with a local inner class?**

-You could use all modifier when you define local inner class.

**Basic Questions1:**

1. **If I don't provide any arguments on the command line, then the String array of Main method will be empty or null?**

-It is empty not null.

1. **What if I write static public void instead of public static void?**

-Programs compiles and run properly.

**3. What is the default value of the local variables?**

-Local variable are visible within the declared method, constructor or block. Local variables are implemented at stack internally. There is no default value for local variables , so local variables should be declared and initial value should be assigned before the first use.

**4. What will be the initial value of an object reference which is defined as an instance variable?**

-The object references are all initialized to null in java.

**5. What is the purpose of default constructor?**

-Default constructor is something that is useful to initialize the data members(variables) to their default value when you create any object.(i.e. null for objects, 0.0 for float and double, false for boolean ,0 for byte, short, int and, long).

**6. Does constructor return any value?**

- No, constructordoes not return any value. While declaring a constructor you will not have anything like return type. In general, constructor  is implicitly called at the time of instantiation. And it is not a method, its sole purpose is to initialize the instance variables.

**7. Is constructor inherited?**

-No, a subclass inherits all the members from its super class. Constructors are not the members, so they are not inherited by subclasses, but the constructor of the super class can be invoked from the sub class. A constructor may only be called with new. It cannot be called as a method.

**8. Can you make a constructor final?**

-No, a constructor can’t be made final. A final method cannot be overridden by any subclasses.

**9. Why main method is static?**

–Java main () method is always static, so compiler can call it without creation of an object or before the creation of an object of the class. In any java program main () method is the starting point where compiler starts program execution.

**10. What is static block?**

Java – Static block is used to initializing the static variables. This block gets executed when the class is loaded in the memory. A class can have multiple static blocks, which will execute in the same sequence in which they have been written into the program

**11. Can we execute a program without main() method?**

-Yes , you can compile and run execute program without main method by using static block. But after static block executed you will get an an error saying no main method found. But this will not worked with JAVA 7 version.

**12. What if the static modifier is removed from the signature of the main method?**

-Programs throws “NoSuchMethodError” error at runtime.

**13. Which class is the superclass for every class**

-Object class is the super class for every class

**14. Why multiple**[**inheritance**](http://gem.squadinfotech.in/mod/assign/view.php?id=703)**is not supported in java?**

-The reason behind this to prevent ambiguity. Consider a case where class B extends class A and class C and both class A and C have the same method display(). Now java compiler cannot decide , which display method it should be inherit. To prevent such situation, multiple inheritances is not allowed in java.

**15. What is composition?**

–Composition is design technique to implement has-a relationship in class. We can use java inheritance or object composition for code reuse. Java composition is achieved by using instance variables that refers to other objects.

**16. What is difference between aggregation and composition?**

-Aggregation implies a relationship where the child can exist independently of the parent. Composition implies a relationship where the child cannot exist independent of the parent.

**17. Why Java does not support pointers?**

**18. Can you use this() and super() both in a constructor?**

- We can use super () as well as this () only once inside the constructor. If we use super() twice or this() twice or super () followed by super() , then immediately we get compile time error i.e. we can use either super() or this() as first statement inside constructor and not both.

**19. Why method overloading is not possible by changing the return type in java?**

-Overloading is mechanism of binding the method call with method body dynamically based on the parameters passed to the method call. It is not possible to execute which method based on return type therefore , overloading not possible just by changing the return type.

**20. Can we overload main() method?Can we override static method?**

-Yes, main method can be overloaded .Overloaded main method has to be called from inside the “public static void main(string args[])” as this the entry point when class is launched by the JVM.

No, static method cannot be overridden.

No, we cannot override main method of java because static method cannot be overridden.

**21. Why we cannot override static method?**

-Method overriding is feature in java which is related to run time polymorphism. For Static methods, method call will depend on upon type of reference rather than the type of object. It means method call is decided at compile time only. So, static method cannot be overridden.

**22. Can we override the overloaded method?**

-Yes you can override the overload method.

-Overloading is when you define two method with same name, in the same class, distinguished by different signatures.

-Overriding is when you redefine a method that has already been defined in a parent class(using the exact signature)

-Overload is resolve at compile time.

-Overriding is resolve at runtime.

**23. Can we initialize blank final variable?**

–Yes, you can initialize a blank final variable in constructor or instance initialization block.

**24. Can you declare the main method as final?**

-Yes, you can declare the main method as final in java. The compiler does not throw any error. The main use of final method in java is they are not overridden.

We cannot override final method in subclasses.

**25. Can there be any abstract method without abstract class?**

-You can’t, It’s kind of definition of abstract class. Its same reason you can’t instantiate an object as an abstract class.

Either-

-you need to use interface

-leave the method empty in parent class

-Refactor your code so that you aren’t instantiating abstract objects.

**26. Can you use abstract and final both with a method?**

- When you declare a class as final, it means you cannot extend it, derive sub-class from it. On the other hand, the reason to we create an abstract class, is to ensure implementations of a method are defined in the derived classes. So it really makes no sense to use both together.

**27. Is it possible to instantiate the abstract class?**

-Abstract classes cannot be instantiated, but they can be sub-classed. When an abstract class is a sub-classed, the sub-class usually provide all implementations for the abstract methods in its parent class. However, if it does not, then sub-class must also be declared as abstract.

-Because abstract class is a incomplete class (it means it contains abstract methods without body and output). We cannot create an instance or object.

-You can instantiate an abstract class. You only need to provide concrete sub-class.

**28. Can you declare an interface method static?**

-All method in an interface are explicitly abstract and hence you cannot define them as static or final because static or final method cannot be abstract ,By default all the method present inside an interface are public and abstract.

**29. Can an Interface be final?**

- If you make an interface you cannot implement its methods which defies very purpose of the interface.

**30. Can we define private and protected modifiers for variables in interface?**

- If you declare a variable in interface with private and protected it will give you error. If you do not specify a modifier the compiler will add public and static final automatically. Having private and protected members in an interface doesn’t make sense conceptually.

**31. When can an object reference be cast to an interface reference?**

-An object reference be cast to an interface reference when an object implements the referenced interface.

**32. Do I need to import java.lang package any time? Why ?**

-No java.lang package is a default package of java therefore , there is no need to import it explicitly i.e. without importing you can access the classes of this package.

**33. Can I import same package/class twice? Will the JVM load the package twice at runtime?**

If-One can import the same package or same class multiple times. Neither compiler nor JVM complains about it. And the JVM will internally load the class only once no matter how many times you import the same class.

**34. What is static import ?**

Yes-With the help of static import we can access the static members of class directly class name or any object.

For example-we always use sqrt() method of Math class by using Math class i.e. Math.sqrt(), but by using static import we can access sqrt() method directly.

**35. Is it necessary that each try block must be followed by a catch block?**

-It is not necessary that each try block must be followed by a catch block. It should be followed by either catch block or finally block. And whatever exceptions are likely to be thrown should be declared in throws clause of the method.

**36. Can finally block be used without catch?**

-Yes, we can try without catch block by using finally block. You can use try with finally. As you know finally block always executes even exception or return statement in try block except in case of system.

**37. Is there any case when finally will not be executed?**

-Yes, finally block is always get executed unless there is an abnormal program termination either resulting from a JVM crash or from a call to system.exit(0)

We cannot say the finally block is always executes because sometimes if any statement like System.exit(1) or some similar code is written to try block then program will automatically terminate and finally block will not be executed in this case.

**38. Can an exception be re thrown?**

-An exception can be thrown in catch block. This action will cause the exception to be passed to the calling method. If rethrow operation occur in the main method then the exception is passed to the JVM and displayed on the console.

**39. Can subclass overriding method declare an exception if parent class method doesn't throw an exception ?**

An-If a superclass method does not declare an exception , sub-class overridden method cannot declare a checked exception but it can declare checked exception.

**40. What is the meaning of immutable in terms of String?**

-Immutable means unmodifiable or unchangeable. Once string object is created its data or state can’t be changed but a new string object is created.

**41. Why string objects are immutable in java?**

-The String objects are immutable in java because String objects are cached in string pool.

Another reason of why String class is immutable could die due to HashMap.Since String is very popular as HashMap key, its important for them to be immutable so that they can retrieve the value object which was stored in HashMap.

**42. How many ways we can create the string object**

is

**43. Why java uses the concept of string literal?**

-To make java more efficient (because no new object created if it is exists already in string constant pool).

**44. How can we create immutable class in java ?**

-Make your class final***,***so that no other classes can extend it.

-Make all your fields final***,***so that they’re initialized only once inside the constructor and never modified afterward.

-Don’t expose setter methods.

-When exposing methods which modify the state of the class, you must always return a new instance of the class.

-If the class holds a mutable object:

-Inside the constructor, make sure to use a clone copy of the passed argument and never set your mutable field to the real instance passed through constructor, this is to prevent the clients who pass the object from modifying it afterwards.

-Make sure to always return a clone copy of the field and never return the real object instance.

**45. What is the purpose of to-string() method in java ?**

-String representation of String object or standard way of representing string.

**46. Can we access the non-final local variable, inside the local inner class.**

-Yes we can access the local final variables using method local inner class because the final variables are stored in the heap and live as the method local inner class object may live.

**47. What is the purpose of finalize() method?**

-The finalize()method is a pre-defined method in the Object class and it is protected. The purpose of a finalize()method can be overridden for an object to include the cleanup code or to dispose of the system resources that can be done before the object is garbage collected

**48. Can an unreferenced objects be referenced again?**

-Yes it is possible we can get the reference of unreferenced objects by using this keyword in finalize method. The finalize() method is called by garbage collector before releasing the instance from service. It gives the developer a chance to do operations on the instance.

**49. What kind of thread is the Garbage collector thread?**

-Daemon thread in java. Daemon thread is a low priority thread that runs in background tos perform tasks such as garbage collection

**50. What is difference between final, finally and finalize?**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Final** | **Finally** | **Finalize()** |
| 1. | Final is used to apply restriction on class, method or variable. Final class can’t be inherited, final method can’t be overridden and final variable can’t be changed | Finally is used to place important code , it will be executed whether exception is handled or not. | Finalize is used to perform clean up processing just before object is garbage collected |
| 2. | Final is a keyword. | Finally is a block | Finalize is method |

**51. What is transient keyword?**

-Transient is a variables modifier used in serialization. At the time of serialization, if we don’t save value of particular variable in a file, then we use transient keyword. When JVM comes across transient keyword, it ignores original value of the variable and save default value of the variable data type.

**52. What is the purpose of the System class?**

-The purpose of the System class is to provide access to the system resources. It contains accessibility to standard input, standard output, error output streams, current time in milis, terminating the system.

**53. What is singleton class?**

-A Singleton class is a class that can have only one object (an instance of class) at a time.

To design a Singleton class: Make a constructor as private. Write static method that has return type of object of this Singleton class.

**Collections :**

**1.What is the difference between Array-list and Vector?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Array-list** | **Vector** |
| 1. | Not Synchronized | Synchronized |
| 2. | ArrayList is not a legacy class. It is introduced in JDK 1.2 | Vector is legacy class |
| 3. | ArrayList uses iterator interface | Vector uses Enumeration interface to traverse the elements. But it can use Iterator also. |
| 4. | ArrayList is fast because it is non-synchronized. | Vector is slow because it is synchronized i.e. in multithreading environment; it will hold other thread in runnable or waiting state until current thread releases the lock of object. |

**2.What is the difference between Array-list and Linked-list?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Array-list** | **Linked-list** |
| 1. | Dynamic | Doubly linked list |
| 2. | Manipulation is not recommended here. | Manipulation recommended(delete,insert,update). |
| 3. | It is good for searching(because it search by index) | Searching is difficult (because it check every address) |

1. **What is the difference between Iterator and List-iterator?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Iterator** | **List-iterator** |
| 1. | The Iterator traverses the elements in the background direction only | ListIterator traverses the elements in backward and forward directions both |
| 2. | The Iterator can be used in List,Set and Queue. | LIstIterator can be used in List only |
| 3. | The Iterator can only perform remove operation while traversing the collection | ListIterator can perform add, remove and set operation while traversing the collection |

1. **What is the difference between Iterator and Enumeration?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Iterator** | **Enumeration** |
| 1. | The Iterator can traverse legacy and non-legacy elements | Enumeration can traverse only legacy elements. |
| 2. | The Iterator is slower than Enumeration | Enumeration is faster than Iterator. |
| 3. | The Iterator is fail-fast. | Enumeration is not fail-fast |
| 4. | The Iterator can perform remove operation while traversing the collection. | The Enumerationcan perform only traverse operation on the collection. |

1. **What is the difference between List and Set?**

|  |  |  |
| --- | --- | --- |
| **No.** | **List** | **Set** |
| 1. | List is an ordered collection it maintains the insertion order which means upon displaying the list content it will display the elements in the same order in which they got inserted into the list | Set is an unordered collection it doesn’t maintain any order. There are few implements of Set which maintain the order such as LinkedHashSet |
| 2. | List allows duplicates | Set doesn’t allows duplicates. All the elements of Set should be unique if yoy try to insert the duplicate element in set it would replace the existing value |
| 3. | List allows any no. of null value | Set can have only a single null value at most. |
| 4. | ListIterator can be used to traverse a list in both the direction(forward and backward) | ListIterator cannot be used to traverse a Set, We can use Iterator. |
| 5. | List Interface has one legacy class called Vector | Set Interface does not have wany legacy class. |

1. **What is the difference between Hash-Set and Tree-Set?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Hash-set** | **Tree-set** |
| 1. | HashSet gives better performance (faster) operations like add,remove,contains size etc. | TreeSet is slower than HashSet |
| 2. | HashSet offers constant time cost | TreeSet offers log(n) time cost for such operation |
| 3. | HashSet does not maintain any order of elements | TreeSet elements are stored in ascending order by default. |

1. **What is the difference between Set and Map?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Set** | **Map** |
| 1. | Set and all classes which implements this interface should have unique elements | Map stored the elements as Key & value pair. Map doesn’t allow duplicate keys while it allows duplicate values. |
| 2. | Set allows single null value at most | Map can have single null key at .most and any number of null values. |
| 3. | Set doesn’t maintain any order ; still few of its classes sort the elements in order such as LinkedHashSet maintain the insertion order | Similar to Set, Map also doesn’t stores the elements in an order, however few of its classes does the same. For ex. TreeMap sort the map in ascending order of keys and LinkedHashMap sorts the elements in an insertion order. |
| 4. | Commonly used classes: HashSet, LinkedHashSet,TreeSet, SortedSet etc. | Commonly used classes: HashMap, TreeMap,WeakHashMap,LinkedHashMap,IdentityHashMapetc. |

1. **What is the difference between Hash-Set and Hash-map?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Hash-Set** | **Hash-map** |
| 1. | HashSet implements the Set interface | HashMap implements the Map interface. |
| 2. | In HashSet we store objects(elements or values) | HashMap is used for storing key and value pair. In short it maintains the mapping of key and value. |
| 3. | HashSet doesn’t allow duplicate elements. | HashMap doesn’t allows duplicates keys however it allows to have duplicates values. |
| 4. | HashSet permits to have a single null value. | HashMap permits single null key and any number of null values. |

1. **What is the difference between Hash-map and Tree-map?**

|  |  |  |
| --- | --- | --- |
| **Basis.** | **Hash-map** | **Tree-map** |
| **Definition** | Java **HashMap** is a hashtable based implementation of Map interface. | Java **TreeMap** is a Tree structure-based implementation of Map interface. |
| **Interface Implements** | HashMap implements **Map, Cloneable**, and **Serializable** interface. | TreeMap implements **NavigableMap, Cloneable**, and **Serializable** interface. |
| **Null Keys/ Values** | HashMap allows a single null key and multiple null values. | TreeMap does not allow null keys but can have multiple null values. |
| **Homogeneous/ Heterogeneous** | HashMap allows heterogeneous elements because it does not perform sorting on keys. | TreeMap allows homogeneous values as a key because of sorting. |
| **Performance** | HashMap is faster than TreeMap because it provides constant-time performance that is O(1) for the basic operations like get() and put(). | TreeMap is slow in comparison to HashMap because it provides the performance of O(log(n)) for most operations like add(), remove() and contains(). |
| **Data Structure** | The HashMap class uses the hash table. | TreeMap internally uses a Red-Black tree, which is a self-balancing Binary Search Tree. |
| **Comparison Method** | It uses equals() method of the Object class to compare keys. The equals() method of Map class overrides it. | It uses the compareTo() method to compare keys. |
| **Functionality** | HashMap class contains only basic functions like get(), put(), KeySet(), etc. . | TreeMap class is rich in functionality, because it contains functions like: tailMap(), firstKey(), lastKey(), pollFirstEntry(), pollLastEntry(). |
| **Order of elements** | HashMap does not maintain any order. | The elements are sorted in natural order (ascending). |
| **Uses** | The HashMap should be used when we do not require key-value pair in sorted order. | The TreeMap should be used when we require key-value pair in sorted (ascending) order. |

1. **What is the difference between Hash-map and Hash-table?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Hash-map** | **Hash-table** |
| 1. | HashMap is non-synchronized. It is not thread safe and can’t shared between many threads without proper synchronization code. | HashTable is synchronized. It is thread safe and can be shared with many threads. |
| 2. | HashMap allows one null key and multiple null values | HashTable doesn’t allow any null key or value. |
| 3. | HashMap is a new class introduced in JDK 1.2. | HashTable is a legacy class |
| 4. | HashMap is fast | HashTable is slow |
| 5. | We can make HashMap as synchronized by calling this code  Map m= Collections.synchronizedMap(hashMap); | HashTable is internally synchronized and can’t be unsynchronized |
| 6. | Iterator in HashMap is Fail-fast | Enumeration in HashTable is not fail-fast |
| 7. | HashMap is traverse by Iterator | HashTable is traverse by Enumerator and iterator |
| 8. | HashMap inherits AbstractMap | HashTable inherits Dictionary class |

1. **What is the difference between Collection and Collections?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Collection** | **Collections** |
| 1. | Final is used to apply restriction on class, method or variable. Final class can’t be inherited, final method can’t be overridden and final variable can’t be changed | Finally is used to place important code , it will be executed whether exception is handled or not. |
| 2. | Final is a keyword. | Finally is a block |

1. **What is the difference between Comparable and Comparator?**

|  |  |  |
| --- | --- | --- |
| **No.** | **Comparable** | **Comparator** |
| 1. | Comparable provides single sorting sequence. In other words , we can sort the collection on the basis of single element such as id or name or price | Comparator provides multiple sorting sequences. In other words, we can sort the collection on the basis of multiple elements such as id, name and price etc. |
| 2. | Comparable affects the original class i.e. actual class is modified. | Comparator doesn’t affect the original class i.e. actual class is not modified. |
| 3. | Comparable provides compareTo() method to sort elements | Comparator provides compare() method to sort elements. |
| 4. | Comparable is found in java.lang package | Comparator I found in java.util package |
| 5. | We can sort the list of elements of Comparable type by Collections.sort(List) method | We can sort the list elements of Comparator type by Collections.sort(List,Comparator) method |

**13. What is the advantage of Properties file?**

**14. What does the hash-code() method?**

-HashMap and HashSet uses hashing to manipulating data. They use haseCode() method to check hash values. The default implementation of hashCode() in object class returns distinct integers for different objects.

**15. Why we override equals() method?**

**16. How to synchronize List, Set and Map elements?**

**17. What is the advantage of generic collection?**