

Differences

1. HashMap vs TreeMap

Property \ Map	HashMap	TreeMap
Ordering	not guaranteed	sorted, natural ordering
get / put / remove complexity	$O(1)$	$O(\log(n))$
Inherited interfaces	Map	Map NavigableMap SortedMap
NULL values / keys	allowed	only values

2. Array vs Collection

Array	Collection
<ol style="list-style-type: none"> 1. Arrays are fixed in size 2. Good performance but poor memory utilization 3. It allow only homogeneous elements 4. Programming complexity is more 5. Array can hold primitive data types 6. Array is part of java language 	<ol style="list-style-type: none"> 1. They are growable in nature 2. Good memory utilization but poor performance 3. It allows heterogeneous elements 4. Less coding complexity 5. Collection only allow objects 6. Collection is provided by API

3. LinkedList vs ArrayList

<u>ArrayList</u>	<u>LinkedList</u>
It is the best choice if our frequent operation is retrieval	It is the best choice if our frequent Operation is insertion and deletion
ArrayList is the worst choice if our frequent operation is insertion or deletion	LinkedList is the worst choice if our frequent operation is retrieval operation
Underlying data structure for ArrayList is resizable or growable Array.	Underlying data structure is Double Linked List.
ArrayList implements RandomAccess interface	LinkedList doesn't implement RandomAccess interface

4. HashMap vs HashTable

	Synchronized	Thread Safe	Null Keys And Null Values	Performance	Extends	Legacy
HashMap	No	No	Only one null key and multiple null values	Fast	AbstractMap	No
HashTable	Yes	Yes	No	Slow	Dictionary	Yes

5. == vs .equals()

== => reference comparison

.equals() => content comparison

6. Checked vs Unchecked Exception

Checked Exception vs Unchecked Exception

- The exception which are checked by compiler for smooth execution of program at runtime are called checked exceptions.
- We will get compile time error if we didn't handle checked exceptions.
- The exceptions which are not checked by compiler are called unchecked exceptions.
- Compiler do not produce any error whether you handle or ignore unchecked exceptions.

www.geekyshows.com

7. Thread vs Process

Process	Thread
1. Process means any program is in execution	1. Part of a process
2. Process takes more time	2. Takes less time
3. Process is less efficient in term of communication.	3. More efficient
4. Consumes more resources	4. Consumes less resources
5. Heavy weight	5. Light weight
6. Process is isolated	6. Threads share memory

7. String vs StringBuffer vs StringBuilder

Factor / Class	String	StringBuffer	StringBuilder
Mutability	Immutable	Mutable	Mutable
Thread Safety	Not thread safe	Thread safe	Not thread safe
Performance	Very high	Moderate	Very high

8. Abstraction vs Encapsulation

Abstraction	Encapsulation
1. Abstraction solves the problem in the design level.	1. Encapsulation solves the problem in the implementation level.
2. Abstraction is used for hiding the unwanted data and giving relevant data.	2. Encapsulation means hiding the code and data into a single unit to protect the data from outside world.
3. Abstraction lets you focus on what the object does instead of how it does it	3. Encapsulation means hiding the internal details or mechanics of how an object does something.
4. Abstraction - Outer layout, used in terms of design. For Example:- Outer Look of a Mobile Phone, like it has a display screen and keypad buttons to dial a number.	4. Encapsulation - Inner layout, used in terms of implementation. For Example:- Inner Implementation detail of a Mobile Phone, how keypad button and Display Screen are connect with each other using circuits.

9. Overloading Vs Overriding

Method Overloading	Method Overriding
1. Defining multiples methods in same class with different parameters.	1. Defining same methods in different class with same parameters.
2. Method Signature is different.	2. Return type + Method Signature is same.
3. Checked at compile time.	3. Checked at run time.
4. Also called as Compile time polymorphism/Early binding/Static binding	4. Run time polymorphism/Late binding/Dynamic binding.
5. May or may not need inheritance.	5. Must need inheritance.

Anil Jatale

10. Vector vs ArrayList

ArrayList

- ☐ Asynchronous.
- ☐ Not Thread Safe.
- ☐ High performance.
- ☐ Grows by half of its size.
- ☐ Used in single-user application.

Vector

- ☐ Synchronous
- ☐ Thread Safe
- ☐ Slow performance
- ☐ Double the size when grow
- ☐ Used in multi-user application.

17. Comparable vs Comparator

Comparable	Comparator
1. Java.lang package	1. Java.util package
2. Default natural sorting order	2. Customized sorting order
3. compareTo()	3. compare() & equals()

11. Final vs finally vs finalize

Final	Finally	Finalize
Keyword	Block	Method
Final is used to apply restrictions on class, method and variable. Final class can't be inherited, final method can't be overridden and final variable value 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.

18. Runnable vs Callable

Runnable	Callable
1. java.lang package	1. Java.util.concurrent package
2. A runnable object doesn't return any value	2. May return value
3. Can't throw checked exception	3. Can throw exception
4. Introduced in 1.0	4. In 1.5
5. run() method	5. call() method

19. sleep() vs wait()

Sleep()	Wait()
1. Thread class	1. Object class
2. Called from anywhere	2. Called from only synchronized block
3. Doesn't release lock	3. Releases lock
4. Awaken by interrupt() or time expires	4. Awaken by notify() or notifyAll()

20. new() vs newInstance()

`new()` => object creation by using constructor


NoClassDefFoundError is an error that occurs when a particular class is present at compile time, but was missing at run time

`newInstance()` => If we want to decide type of object to be created at runtime. In this case, we have to use `newInstance()` method

ClassNotFoundException is an exception that occurs when you try to load a class at run time using **`Class.forName()`**

21. Abstract class vs Interface

Abstract Class		Interface
1. <i>abstract</i> keyword	•	1. <i>interface</i> keyword
2. Subclasses <i>extends</i> abstract class	•	2. Subclasses <i>implements</i> interfaces
3. Abstract class can have implemented methods and 0 or more abstract methods	•	3. Java 8 onwards, Interfaces can have default and static methods
4. We can extend only one abstract class	•	4. We can implement multiple interfaces



22. Overriding vs Method Hiding

Difference between Method Hiding and Overriding:

Method Hiding	Overriding
1)Both method should be static.	1)Both method should be non-static.
2)method resolution takes care by compiler based on reference type.	2) method resolution takes care by JVM based on Runtime object.
3)It is considered as compiletime polymorphism or static polymorphism or early binding.	3)It is considered as Runtime polymorphism or dynamic polymorphism or late binding.

Procedural Oriented Programming	Object Oriented Programming
In procedural programming, program is divided into small parts called <i>functions</i> .	In object oriented programming, program is divided into small parts called <i>objects</i> .
Procedural programming follows <i>top down approach</i> .	Object oriented programming follows <i>bottom up approach</i> .
There is no access specifier in procedural programming.	Object oriented programming have access specifiers like private, public, protected etc.
Adding new data and function is not easy.	Adding new data and function is easy.
Procedural programming does not have any proper way for hiding data so it is <i>less secure</i> .	Object oriented programming provides data hiding so it is <i>more secure</i> .
In procedural programming, overloading is not possible.	Overloading is possible in object oriented programming.
In procedural programming, function is more important than data.	In object oriented programming, data is more important than function.
Procedural programming is based on <i>unreal world</i> .	Object oriented programming is based on <i>real world</i> .
Examples: C, FORTRAN, Pascal, Basic etc.	Examples: C++, Java, Python, C# etc.