Interview Questions

1. **What do you know about Java?**

   Java is a high-level programming language. It is platform independent. The java code runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX. Add answers of **Q:3 also**

2. **What are the supported platforms by Java Programming Language?**

   Java runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX/Linux like HP-Unix, Sun Solaris, Redhat Linux, Ubuntu, CentOS, etc.

3. **List any five features of Java?**

   Some features include Object Oriented, Platform Independent, Robust, Interpreted, Multi-threaded

4. **Why is Java Architectural Neutral?**

   Javac compiler generates an architecture-neutral object file format which is also called as byte code and makes the compiled code to be executable on many processors using JVM.

5. **What is JVM ?**

   When Java is compiled, it is not compiled into platform specific machine, rather into platform independent byte code. This byte code is distributed over the web and interpreted by Java Virtual Machine (JVM) on whichever platform it is being run.

6. **Is JVM platform independent?**

   It is designed to adapt to an evolving environment. Java programs can carry extensive amount of run-time information that can be used to verify and resolve accesses to objects on run-time.

7. **List some Java keywords(unlike C, C++ keywords)?**

   Some Java keywords are import, super, finally, etc.

8. **List the primitive data types of java?**

   byte, char, int, long, float, double and boolean

9. **What do you mean by Object?**

   Anything can be a object. A car, a pen, a laptop. An object its own attributes and behavior The attributes are like variables.

(eg) A pen has colour, size, shape
A car has model, price, mileage.
The behaviors are like methods what do you with car object
(ie) getFeatures(), getPriceDetails() etc.

10. Define class?

Class is a collection of objects. A class is a blueprint from which objects are created.
(eg)collection of car objects- Car class

11. What is a local variable?

Variables within methods, blocks and constructor are called local variable. Their scope is within the method only. They don't take any default value. They must be initialized before using them. If not will give compiler error.

12. What is a Instance variable ?

These are variables for an object. Each object has its own set of instance variables(values are different). The values are initialized to default values on creation of the object

13. What is a static variable ?

These variables are available as a single copy in a class. All the objects in a class share the same copy. Only static variables are initialized to their default values when the class gets loaded.

14. What is a constructor?

- Is used to initialize the instance variables
- Comes with class name & has no return type
- Can be private also
- Gets invoked automatically when an object is created.
- The compiler creates a default constructor (without parameters), if you do not create any constructor
- A class can have more than one constructor(parameterized).
- You can never call a constructor by name.

15. How will you create an Object?

Use new operator to create an object.

16. What is the default value of boolean?

false

17. What is the default floating point type in Java?

- double. It takes 64 bits.
- If you want declare float use float a= 10.5f;

18. What is the use of char?

'char' is a data type which is a Unicode representation of characters. It is used for giving backspace, newline, tab etc

19. What are the different access specifiers in java?

public, protected, default, private

20. What is the access specifier used for class?

public & default only. An inner class can be private, as it acts like a class member.

21. What is the access specifier used for method?

public, protected, default & private

22. What is the access specifier used for local variable?

final

23. What happens if you don't inilialize the local variables?

It gives a compile error

24. Is it mandatory to have default case?

No. It is optional.

25. What type of parameter does switch accept?

switch accepts variable that can be of type byte, short, and int. switch in java 7 also accepts String also.

26. Which is the top level of class in java?

Object is the class which is extended by all other class.

27. Which package is imported by default in java?

java.lang is imported by default.

28. Can a java file have more than one java class?

Yes, you can have any number of java classes in a single file. But only one class can be public.

29. Can I have more than one main method in a class?

Yes, but with different parameters. But JVM will check for the method with public static void main(String a[]) as parameter. If not available will throw an exception as Main method not found

30. What is String?

String is a final class. String is fixed and immutable.

31. Why String is immutable?

Once a string object is created, it cannot be changed. So it can be safely shared across threads.
(eg)
String s =”hello”;
s.toUpperCase();
System.out.println(s); // will print ”hello” only.
“HELLO” is created and it will be lost as it is not referenced by any variable.

32. What is inheritance?

Inheritance is used for reusability. The sub class can use the same variables and methods of the super class and also can have its own method.

33. What type of inheritance does java supports?

Simple and Multilevel. Purpose of inheritance is reusability.

34. Why multiple inheritance is not supported in java?

If two class have a method with the same name, when the child class extending these classes try to invoke the method, there is a confusion from which class to invoke. So multiple inheritance is not supported in java

35. What is compile time polymorphism?

It is also called as overloading. Java supports constructor overloading and method overloading. Since the method/constructors are paired during the complie time itself after checking for the parameter, it is called as compile time polymorphism

36. What is the method overloading & purpose?

Method overloading is, having the method being overloaded with one to many parameters. Here the method name must be same, return type same or different and number of parameters or parameter datatype should be different.
This is done to get different functionality from the overloaded methods having same name.

37. Why constructor is called as complier time polymorphism?

While creating an object, the complier checks for a constructor with matching parameters, if it is not available, it will give a compiler error.

38. How will you access the instance variables from a static method?

create Objects for the class to access the instance variables

39. What is constructor overloading?

Having the more than one constructor with one to many parameters is called as constructor overloading

40. Can a constructor be private?

Yes.

41. How to invoke one constructor from another constructor?

Using this()keyword with matching parameter.

42. What is the use of this() keyword?

this() keyword is used to call one constructor from another by passing matching parameter in the same class.

43. What is the use of this keyword?

this refers to the current object. It prevents the local variable hiding the instance variable

44. What is Var-args?

Var args is variable argument list. It can used within a method as parameter. It can values as zero to many. It similar to an array

- You can have only one var- args list in a method
- The var – args list must be the last parameter
- The data type is followed with 3 dots

(Eg) void calAvg (String a, int…marks);

45. What is the difference between the overloading and var args?

Overloading is used when the method name is same, but the method parameters and functionality are different.

Var-args list is used to when the functionality is same and the method values are different for different method call.

46. Can I have more than one main method ?

Yes. we can have more than one main method, but with different parameters. But to run the application, JVM will search for *public static void main(String a[])*.

47. What happens if i dont have the same method signature as *public static void main(String a[])* **? Whether the program will complile?**

Yes the program will compile fine. But during the runtime, it will throw NoSuchMethodException

48. What is the difference between == and equals method while using with String variables?

equals method
This compares the string literal values only
==
This compares the String Object references
Example
String s1="hello";
String s2 = "hello" ;
String s3=new String("hello world");
String s4 = new String("hello world");

s1. equals(s2) ---------->returns true
s3. equals(s4) ---------->returns true

s1==s2 ----------> returns true
s3==s4----------> returns false

49. What is the use of super() keyword?

super() should be in the first line of a subclass constructor.
It is used to call the constructor of the super class.
The parameter within super must match at least one constructor of super class.

50. What is Runtime Polymorphism?

It is also called as overriding. It happens in a super class, subclass scenario.
The super class decides which subclass to class to call only during the runtime.
This is done using *superclass ref = subclass object.*

51. What is the static?

static is a keyword. It can be as block, variables, method, and as top level of nested class.
The method/variable marked as static is available as a single copy only. All the objects share the same value only.
You can call the static method/variables without creating an object.
They can be called using *classname.method name* or *classname.variablename*.

52. How will you call a non static(normal) method from a static method?

Use object to call a non static method
(eg) calling a method from PSVM using object

53. What is the use of static import?

The static import is used to import the static member of one class to another class. The syntax
import static pkg.subpkg.classname.;*
This imports all the static members of a class into the calling class

54. Can you refer to a super class object or a current object inside a static method?

No. super() and this keyword cannot be used inside a static method.

55. Can a static method be overridden?

No. Even if you have the same method signature in the subclass, while calling using dynamic method dispatch [*super class ref= sub class object*], since the method is static only the super class method only will be called.

56. Can I have more than one static block?

Yes

57. In which sequence the static block will run?

In the order of creation.

58. What is the abstraction?

Abstraction is hiding the unwanted details from the user.
It is also used for extending the functionality. An abstract class can be extended by multiple subclasses to get different functionality.

59. What is the abstract class?

An abstract class is one, which has at least one or more abstract methods.

You cannot create an object for an abstract class
The abstract classes must be extended
The subclass provides the functionality for the abstract methods.

60. What is the abstract method?

An abstract method has only the method declaration and no definition.
An abstract method must be implemented in the subclass.

61. What is a java bean?

A java bean is a public class with private instance variables and public getter and setter methods.

62. How will you declare a variable constant?

Using final keyword

63. What keyword prevents the class from extending?

final

64. What keyword prevents the method from being overridden?

A method that is declared final will be prevented from being overridden.

65. Name few inbuilt classes that are final in java.

java.lang.String , java.lang.Math are final classes in java

66. What is the use of native keyword in java?

The native keyword is used with methods. If a method is declared as native, then it means the implementation of the method is given in C language, not in java
Example
public static native java.lang.Thread currentThread();

67. What is a package?

Package is a container for a class. It is nothing but the folder in which it is created.

68. Can a method declared private be used in the subclass?

No

69. What is the use of default / friendly?

default/ friendly is not a keyword. A representation of a class /method/variable without any keyword is default. The scope of this is only within a package.

70. What is the use of protected?

Protected keyword allows the access for all classes in the same package and for subclasses in different packages.

71. What is the use of interface?

Interface in java helps to get multiple implementations. It also supports abstraction. A class can implement

72. What are the methods in an interface?

Interface methods are public and abstract by default.
It does not allow concrete (normal) methods.
In java8, it allows default methods.

73. What type of variables are in an interface?

Variables are by default public, static, final

74. What is a tag/marker interface?

An interface without methods is called as tag/marker interface.

75. Name a marker interface?

java.io.Serializable

76. Can interface be extended?

Yes

77. What is the keyword used by a class to use an interface?

implements

78. How will you call a variable in an interface?

Can be called using
Interfacename.variablename
Implementingclassname.variablename.

79. Can a class implement multiple interfaces?

Yes

80. What is the Multithreading?

Multithreading is a concept where are two or more parts run simultaneously. Each part is called as a thread. This is used to utilize the ideal time of the CPU

81. What is the thread?

A thread is a separate part of execution. Multiple threads run simultaneously.

82. How will you create a thread?

By extending Thread class on implementing *Runnable* interface.

83. What is the default thread that runs the moment JVM starts the execution?

The main thread

84. What are the priorities of thread?

MAX_PRIORITY- 10
MIN_PRIOIRITY-1
NORM_PRIORITY-5

85. What are the life cycle method of a thread?

- When a thread is first created, it is in the **NEW** State
  - ***Thread t = new Thread();***
- When you invoke ***start*** method, it gets ready to get the CPU.
  - ***t.start()***
- The thread changes to **RUNNABLE** state (eligible for execution) depending on its priority
- When ***sleep/wait*** method is called on a **RUNNABLE** thread, it may enter the NOT RUNNABLE state.
  - ***Thread. Sleep();***
- When a thread is **BLOCKED,** it is still alive, but it is not eligible for execution.
- A **BLOCKED** thread becomes ready to **run** again when the sleeping thread wakes up.
- This thread occupies the CPU depending on its **PRIORITY**
- When a thread terminates, it is said to be **DEAD**

86. What method is used to identify the thread that is running currently?

Thread.CurrentThread() is the method that is used to get the thread that is occupying the CPU currently. It returns a thread reference.

87. What are Daemon Threads?

- Are like a service providers for other threads or objects running in the same process as the daemon thread.
- Are used for background supporting tasks and are only needed while normal threads are executing.
- If normal threads are not running and remaining threads are daemon threads then the interpreter exits.
- Does not prevent the JVM from exiting when the program finishes even if the thread is still running.
- Example for a daemon thread is the garbage collection.

88. How to create daemon thread?

By using **setDaemon(true)** method on the thread instance.

89. What is Synchronization?

- When two or more threads need access to a shared resource, they need some way to ensure that the resource will be used by only one thread at a time.
- Achieved with the process of synchronization.
- Synchronization in Java can be achieved using the keyword – **synchronized**

90. Where can I use synchronized keyword?

- In methods
- as synchronized blocks

91. What is a synchronized block?

- This is a synchronized block

```
synchronized(Object o){
    o.anymethod();
}
```

- performs two actions:
  - After getting a reference to an object, it locks that object
  - The thread entering this block gets the lock of this object
  - The method is called on the object
- After execution of the body has completed, either normally or abruptly, it unlocks that same lock.

92. What happens if a thread is in a synchronized method?

If a thread is in synchronized method, all the other threads trying to call this method or any other synchronized method using the same object have to wait.

93. What is the use of sleep method?

The sleep method holds the lock of the object and releases the CPU, so that any other thread can use the CPU.
The thread wakes up automatically after the time elapses and goes to the thread pool.

94. What is Interthread Communication?

- Communication between threads is called Inter-Thread communication.
- Assume data is produced by one thread(Producer), and consumed by another thread(Consumer).
- The Consumer has to check every now and then whether it has to data to act upon.
- This is a huge waste of CPU time.
- But using InterThread Communication
- If the Producer would communicate to the Consumer once it has finished producing the required data, the consumer need not use up CPU cycles just to check whether the producer has done its job.

95. How is InterThread communication achieved ?

Using *wait(), notify(), notify All()* methods. They are from Object class.

96. What is wait method used for ?

- It tells the thread to release the lock of this monitor & go to sleep.
- The thread *waits* until another thread notifies it.
- The thread wakes up either through a call to the *notify method* or the *notifyAll* method.
- The thread then waits until it can re-obtain ownership of the monitor and resumes execution.
- This method should only be called by a thread that is the owner of this object's monitor.
- Called within *synchronized* methods only.

97. What is the use of notify and notify All ?

- *notify()* method wakes up a single thread that is waiting on this object's monitor. If many threads are waiting on this object, one of them is chosen to be awakened.
- *notify All()* wakes up all threads that are waiting on this object's monitor.

98. Difference between sleep and wait()?

sleep

- holds the lock of the object. The thread wakes up automatically after the time elapses

wait

- can be called within synchronized method only. The thread releases the lock of the object. The thread wakes up when notify is called on the objects monitor.

99. What is an Exception ?

An abnormal condition that disrupts the normal flow of execution. This can be logical errors , database errors like connecting with the database.

100. What is the Unchecked Exceptions?

Unchecked Exception is an exception if not handled will be handled by the JVM.
JVM handles the exception and terminates the program. These are also called as RuntimeException.
All classes that come under ***RuntimeException*** are also called as Unchecked Exceptions.

101. What are Checked Exceptions?

Checked Exceptions are exceptions if not handled will throw a compiler error. The checked exceptions must be handled using *try-catch* or declared using *throws*.
Call the method throwing checked exception within *try* block or use *throws* keyword in method declaration

102. Name of few unchecked exceptions?

Arithmetic Exception
NumberFormat Exception
ArrayIndexOutOfBounds Exception

103. Name of few checked exception?

IOException
SqlException
InterruptedException

104. What is the super class for Error and Exception?

Throwable.

105. What are errors ?

Errors are system errors. They must not be handled.
eg. OutOfMemoryError
This must not be handled. Rather increase the heap size to avoid this error.

106. Difference between Exception and Error?

Exceptions are logical errors. They must be handled by the developer. They must be handled. Error is not meant to catch as even if you catch it you can not recover from it.

107. Name Few errors?

StackOverflowError
OutOfMemoryError

108. What are keywords used for handing exception?

try, catch, finally, throw, throws

109. What is the use of finally?

finally is a block. This is used to release the resources, like closing a file, database connection. Finally block will be called when there is an exception and also when there is no exception.

110. What is the use of try block?

The purpose of try block is to have all error prone statements. It has to be either with *catch()* or with *finally* or both.

111. What is the use of catch?

catch( ) is used to handle the exception that is thrown in the try block.
You can have more than one catch block for a single try. But the order of catch is the subclass exceptions must be written(handled) first, followed by super class exceptions.

112. Can I have a try catch in finally?

Yes , you can have a try and catch block in finally.
(eg) while closing file objects

113. What is the use of throw keyword?

'throw' is used to say that the exception is not handled here, but it is thrown to the method from where it is called. The syntax is ***throw Throwableinstance****;*
eg. throw new Exception();

114. What is the use of throws keyword?

Any exception that comes along throws keyword is a checked exception. When the method is used, if the exception is not handled, the program does not compile. Throws keyword throws exception to next class until main class handles it by surrounding it by try and catch block.

115. Which method is used to terminate the application?

System.exit(0)

116. If System.exit(0) is called inside the try block, will finally be called?

No. It will completely terminate the application.

117. What is Auto boxing?

Auto boxing is used to convert primitive data types into objects
eg. int x =10; Integer y = x;
'x' is converted into Integer object;

118. What is Autounboxing?

Autounboxing is used to convert objects into primitive datatypes
eg. Integer y = new Integer(90); int x = y;
'y' object is converted into int primitive data type.

119. What are Wrapper classes?

Wrapper classes are classes that wrap the primitive data types into objects. Each primitive data type is having the corresponding wrapper class.
eg. float primitive has Float Wrapper class.
double primitive has Double Wrapper class.

120. Name the Wrapper classes ?

Integer, Float, Byte, Short, Long, Double, Boolean, Character

121. Name the Wrapper classes that come under Number class?

Integer, Float, Byte, Short, Long, Double

122. What is the use of hash code() and equals() method?

Every object has access to the *equals()* method because it is inherited from the *Object* class. The default implementation - compares the memory addresses of the objects.

You can override the default implementation of the *equals()* method from *Object class*. While overriding equals must also override *hashCode()*. Otherwise a violation of the general contract for Object.hashCode() will occur.

Set implementation classes override equals method to identify duplicate objects.

123. What is cloning ?

Cloning is creating a copy of the original object. It is field by field copy. The cloned object has separate memory address. The types of cloning supported by Java

- **Shallow Cloning**
- **Deep Cloning**

In Cloning if **x2 = x1.clone()**, then

- **x2== x1** returns false
- **x2.getClass()==x1.getClass()** returns true

124. What are steps involved in cloning?

The Class must implement **Cloneable interface** and override **clone** method
No constructor is called on the object being cloned.

125. What is the serialization?

Serialization is saving the state of an object into a file. It is the process of converting an object into stream of bytes in order to store the object into memory. The reverse process is called as deserialization

126. How serialization is achieved?

To achieve serialization, the class must implement **java.io.Serializable** interface

127. What are the classes that are used for serialization?

ObjectInputStream and ObjectOutputStream

128. Identify the methods used for serilaization / deserialization?
Serialization: public void writeObject(Object o)
Deserialization: public Object readObject()

129. What keyword is used to prevent seriialization of certain instance variables?

'transient' is the keyword used with the instance variables to avoid serialization of that particular field.

130. What happens to the sub class variable if only super class implements Serializable?

The subclass instance variables also will be serialized.

131. What happens to the super class variables if only sub class implements Serializable?

The super class instance variables will not be serialized.

132. What interface must a class implement to create duplicate objects?

The class must implement Cloneable interface.

133. What type of cloning does java support?

Shallow cloning

134. What is Shallow Cloning?

In Shallow Cloning

- If the class has only primitive data type members,
  - then a completely new copy of the object will be created
  - the reference to the new object copy will be returned.
- If the class contains members of any class type,
  - then only the object references to those members are copied
  - hence the *member references in both the original object as well as the cloned object refer to the same object.*

135. What is Deep Cloning?

In Deep Cloning

- The object is copied along with the objects it refers to.
- Deep clone copies all the levels of the object from top to the bottom recursively.
- All the member classes in original class should support cloning
- In clone method of original class, call super.clone() on all member classes.
- If any member class does not support cloning , then in clone method, one must create a new instance of that member class and copy all its attributes one by one to new member class object. This new member class object will be set in cloned object.
- Or use Serialization for deep copy

136. What is enum ?

enum is a keyword. It is used as data type in java to restrict with specific set of values. It contains fixed constants.

eg

```
enum weekdays{
SUN,MON,TUE,WED,THURS,FRI
}
```

137. Which method will be called by the garbage collection before doing the garbage collection?

finalize() is the method that is called before doing gc.

138. What does finalize () do?

The resources associated with the object are released inside the method, before garbage collecting the object.

139. Which method is used to call garbage collector explicity?

System.gc()
Runtime.getRuntime().gc();

140. What is generics?

Generics in Java is one of important feature added in Java 5. It is used to provide compile time type-safety.

141. What is the use of toString()?

toString() method is from Object class. It is used to give string representation of an object. To get this, it is mandatory to override toString() method in the required class and give the implementation**.**

142. What is Collections?

Collections is a class in java.util package. It is used for operations like sorting, searching collection classes like ArrayList, HashSet etc

143. What is collections framework?

Collection framework represents a unified architecture for storing and manipulating group of object.

It has:

- Interfaces and its implementations classes
  - List,Queue,Set are interfaces
  - ArrayList,LinkedList, HashSet are implementation classes
- Algorithm.

144. Name the interfaces in collection framework?

- List, Set, Queue

145. What is List?

List is an interface. List is an ordered collection by index. List accepts duplicates. List interface has got concrete implementations as ArrayList, LinkedList and Vector.
ArrayList - growable array - ordered by index

LinkedList - can act as Stack and Queue - ordered by insertion
Vector - growable array - ordered by index

146. What is Set?

Set is an interface. It has ordered, unordered and sorted collection. Set does not accept duplicates, has only unique elements.
Set has get concrete implementations as HashSet, LinkedHashSet and TreeSet.
HashSet - unordered Collection
LinkedHashSet - ordered by insertion
TreeSet - Sorted Collection

147. What is difference between HashMap and HashTable?

Both collections implements Map and store value as key-value pairs.
HashMap

- is not synchronized. so good performance.
- Uses Iterator and concurrent modification is not possible
- Accepts null values

Hashtable

- Is a legacy class
- Is synchronized. so can be used with multiple threads.
- Uses Enumerator and concurrent modification is possible
- doesn't allow key or value as null.

148. Name the collection that is sorted?

TreeSet.

149. What is ArrayList?

ArrayList class extends abstract class List and implements the List interface. ArrayList supports dynamic arrays that grows as elements are added and shrinks as elements are removed.
ArrayList is an ordered collection, where the elements can be accessed through index.
ArrayList helps in faster iteration

150. What is LinkedList?

LinkedList is a class that implements the List, Queue interface. It supports dynamic arrays that grows as elements are added and shrinks as elements are removed.
LinkedList can be made to act like a queue or stack.

The elements are double linked to each other. They help in faster insertion of the elements

151. What is Vector?

Vector is a legacy class that implements the List interface. Vector supports dynamic arrays that grows as elements are added and shrinks as elements are removed. The methods in Vector are synchronized, so the performance is less.

152. What is the difference between ArrayList and Vector?

ArrayList:

- ArrayList is non-synchronized which means multiple threads can work on ArrayList at the same time.
- Array List gives better performance as it is non-synchronized.
- Both ArrayList and Vector can grow and shrink dynamically to maintain the optimal use of storage

Vector:

- Vector is synchronized.
- Vector operations gives poor performance
- Enumeration returned by Vector allows concurrent modification
- ListIterator returned by ArrayList does not allow concurrent modification.

153. Which is an unordered collection?

HashSet

154. How does Set identifies duplicates?

By overriding hashcode() and equals() method.

155. How to sort list of Strings ?

Use **Collections.sort(list)** method

156. How to sort list of user defined objects(eg. Employee) ?

Using Comparable

The(Employee) class must implement Comparable and override compareTo(Object o) method. Inside the method identify the instance variable you want to sort with and compare.

Then use **Collections.sort(list)**

Using Comparator

Create a separate class(SalarySorter) that implements Comparator and override compare(Object o1,Object o2) method. Inside the method identify the instance variable(in tis case salary) you want to sort with and compare.
Then use **Collections.sort(list, instanc of SalarySorter**). The list of employees will be sorted by salary.

157. What are the interface used for sorting ?

java.lang.Comparable.

java.util.Comparator

158. Can I add a null element to a Treeset ?

In a TreeSet the elements are ordered using their natural ordering, or by a Comparator provided at set creation time, depending on which constructor is used. So adding null element to an existing TreeSet will throw NullPointerException.
Null can be added only if the set contains one element

159. What is Map?

Map is an interface. It accepts key/value pars. Here key is an object and value is also an object. Map has many concrete implementation as HashMap, LinkedHashMap and TreeMap.

160. How to iterate an ArrayList?

You can use Iterator interface, ListIterator interface, for-each loop

161. What is an Iterator interface?

java.util.Iterator is a traversal interface. This interface is used to step through the elements of a Collection. It allows iteration in forward direction only. It follows Iterator design pattern.

162. What does ListIterator do?

ListIterator allows the elements of the ArrayList to be iterated either from top to bottom or bottom up or even from middle.

163. Difference between HashMap and TreeMap?

HashMap adds the key as per the hashcode value. The elements are not sorted

TreeMap sorts the keys and then adds to the map. The elements in TreeMap are sorted by keys.

164. Which type of collection does faster Iteration?

ArrayList

165. Which type of collection does faster insertion?

LinkedList

166. When to use ArrayList and when to use LinkedList ?

Adding new elements is fast for both types of List.

Retrieval is faster in case of ArrayList, as using "get" is fast, but for LinkedList, it's slow. It's slow because there's no efficient way to index into the middle of a linked list.

When removing elements, using ArrayList is slow. This is because all remaining elements in the underlying array of Object instances must be shifted down for each remove operation. But removal is faster using LinkedList , because deletion can be done simply by changing a couple of links.
An ArrayList works best for doing random access on the list.
A LinkedList works better for doing a lot of editing in the middle of the list.

167. What is StringBuilder ?

StringBuilder is a mutable sequence of characters. StringBuilder is like a String, but can be modified. All the operations performed are done on the same object. StringBuilders are not thread safe as methods are not synchronised. But they are used for faster performance.

168. What is StringBuffer?

StringBuffer is a thread-safe, mutable sequence of characters. A StringBuffer is like a String, but can be modified. All the operations performed are done on the same object. String buffers are safe for use by multiple threads, as the methods are synchronized.

169. What is JDBC?

JDBC stands for Java Database Connectivity. It is a standard Java API for database-independent connectivity between the Java programming language and a wide range of databases. The inbuilt package used with JDBC is java.sql.*;

170. Describe a general JDBC Architecture.

General JDBC Architecture consists of two layers:

JDBC API: This provides the application-to-JDBC Manager connection.

JDBC Driver API: This supports the JDBC Manager-to-Driver Connection.

171. What are the common JDBC API components?

JDBC API consists of following interfaces:

Driver, Connection, Statement, ResultSet, PreparedStatement.

and a class - **DriverManager**

172. What is JDBC Driver ?

The JDBC Driver provides vendor-specific implementations of the abstract classes provided by the JDBC API. This driver is used to connect to the database.

173. What is the use of DriverManager ?

DriverManager is a class in java.sql.* package. It is the basic service for managing a set of JDBC drivers.

As part of its initialization, the DriverManager class will attempt to load the driver classes referenced in the "jdbc.drivers" system property. This allows a user to customize the JDBC Drivers used by their applications.

I**n your properties file you can specify:**

jdbc.drivers=foo.bah.Driver:wombat.sql.Driver:bad.taste.ourDriver

A program can also explicitly load JDBC drivers at any time.

The driver (com.mysql.Driver) can loaded with the following statement:

Class.forName("com.mysql.Driver");

When getConnection() is called the DriverManager will attempt to locate a suitable driver from amongst those loaded at initialization.

Once loaded, it will establish a connection

174. What is Connection?

Connection represents a connection (session) with a specific database. SQL statements are executed and results are returned within the context of a connection.

175. What are the different JDB drivers available?

There are four types of driver available. They are:

Type 1 : JDBC-ODBC Bridge Driver

Type 2: Native API Partly Java Driver-

Type 2: Native API Partly Java Driver-
Type 4: JDBC Net pure Java Driver

176. What is Connection pooling?

Connection pooling is a technique used for sharing server resources among requesting clients. Connection pooling increases the performance of Web applications by reusing active database connections instead of creating a new connection with every request. Connection pool manager maintains a pool of open database connections.

177. What are the types of statements in JDBC?

Statement, PreparedStatement, CallableStatement

178. What is an Statement?

Statement encapsulates an SQL statement which is passed to the database to be parsed, compiled, planned and executed.

179. What is a PreparedStatement?

A SQL statement is pre-compiled and stored in a Prepared Statement object. It is used to run Pre compiled SQL. This object can then be used to efficiently execute this statement multiple times. The object of PreparedStatement can be created using connection.prepareStatement() method. This extends Statement interface.

Example:

PreparedStatement statement = connection.prepareStatement(insert into emp values(?,?));

180. What is the CallableStatement?

This interface is used to execute the stored procedures. This extends Prepared Statement interface. The object of Callable Statement class can be created using connection.prepareCall() method.

181. What is a ResultSet?

ResultSet is an interface. It represents set of rows retrieved due to query execution of the database.

Example:

ResultSet rs = stmt.executeQuery(sqlQuery);

182. Which type of JDBC driver is the fastest one?

JDBC Net Pure Java Driver(Type 4) is the fastest driver because it converts the JDBC calls into vendor specific protocol calls and it directly interacts with the database.

183. How do you register a driver?

Class.forName(String drivername):
DriverManager.registerDriver():

184. What is Batch Processing ?

Batch Processing allows you to group related SQL statements into a batch and submit them with one call to the database.
The **addBatch()** method of Statement, PreparedStatement, and CallableStatement is used to add individual statements to the batch. The **executeBatch()** is used to start the execution of all the statements grouped together.

185. What is a transaction?

A transaction is a logical unit of work. To complete a logical unit of work, several actions may need to be taken against a database. Transactions are used to provide data integrity, correct application semantics, and a consistent view of data during concurrent access.

186. What is the use of blob, clob data types in JDBC?

These are used to store large amount of data into database like images, movie etc which are extremely large in size.

187. What is difference between JDBC, JNDI and Hibernate?

Hibernate is an Object-Relational Mapping tool. It maps Objects to relational data.

The **Java Naming and Directory Interface (JNDI)** is an API to access different naming and directory services. You use it to access something stored in a directory or naming service without haveing to code specifically to that naming or directory service.

Java DataBase Connectivity (JDBC) API is an API to access different relational databases. You use it to access relational databases without embedding a dependency on a specific database type in your code.

188. What is ResultSetMetaData ?

JDBC API has this interface **ResultSetMetaData.** This provides comprehensive information about the types and properties of the columns in a ResultSet object. get You can get metadata of a table like total number of columns, column name, column type, whether the column supports the given features etc. ,

189. What is DatabaseMetadata ?

JDBC API has this interface **DatabaseMetaData.** This provides comprehensive information about the database as a whole. You can see what tables are defined in the database, what is the driver name and what columns each table has, whether given features are supported etc.

190. What is an IO stream?

It is a stream of data that flows from source to destination. A stream is an abstraction that either produces or consumes data.
Example: File Copying.
Two streams are involved in file copying – FileInputStream and FileOutputStream. An input stream reads from the file and stores the data in the process. The output stream reads from the process and writes to the destination file.

191. What is the difference between Reader/Writer and InputStream/Output Stream?

The Reader/Writer classes are character-oriented and the InputStream/OutputStream classes are byte-oriented.

192. What is an out.println()?

out is an object of PrintStream class defined in System class.
println() is a method of PrintStream class.

193. What is composition?
Composition is an association in which one class belongs to a collection. This is a part of a whole relationship where a part cannot exist without a whole. If a whole is deleted then all parts are deleted.
Example
An order is a whole and line items are parts. If an order is deleted then all corresponding line items for that order should be deleted.
So composition has a stronger relationship.

194. What is Aggregation?

Aggregation is an association in which one class belongs to a collection. This is a part of a whole relationship where a part can exist without a whole. If a whole is deleted then all parts can exist

Example

A School is a whole and Student are parts. If the school object is deleted then all corresponding student object can still survive.

So aggregation has a weaker relationship.

195. What is HAS-A relationship in Java?

Aggregation is used to represent HAS-A relationship in **Java**. HAS-A relationship is based on usage, rather than inheritance. If a class has a reference of another class as instance variable, it is called as HAS-A relationship aka Aggregation.

Example

```
class Employee{
String name;
Address address;                // has- a relationship
}
class Address{
String city,state;
}
```

196. What is Reflection in Java?

Reflection helps to inspect classes, interfaces, fields and methods at runtime, without knowing the name of the classes, methods at compile time. This is done with multiple method from **java.lang.reflect** package

197. Where is Reflection used ?

- By Tomcat - to identify servlets with mapping
- By Eclipse - for content assist
- By Spring - for DI
- By Hibernate - for mapping classes to tables
- By JAXB - for marshalling and unmarshalling

198. What is the use of Class.forName(String class name) method?

When this method is called, the JVM loads the class and calls all the static blocks inside the class. If the class is not found its throws ClassNotFoundException.

199. What is the use of annotation?

Annotations are data about data. They are far more powerful than java comments and javadoc comments. They can be carried over to runtime , other two(comments and javadoc)stops with compilation level.

Annotations can be used as
Information for the compiler — Annotations can be used by the compiler to detect errors or suppress warnings.
Compile-time and deployment-time processing — Software tools can process annotation information to generate code, XML files, and so forth.
Runtime processing — Some annotations are available to be examined at runtime.

200. Name few annotations in java
@Override, @Deprecated, @SuppressWarnings.

----------**Happy Learning**----------