Testing Help

erview Questions and

ions:

reshers

pared to
ava

with

va

definitions, Oops concepts, Access specifiers, Collections, Exceptions, Threads, Serialization etc., with examples to make you get ready perfectly to face any JAVA interview confidently.

important and commonly asked basic and advanced
letailed answers.

ge and is platform independent.

ed by Sun Microsystems. There are a lot of
eveloped using Java.

n works on different platforms without any modification.

- **High Performance:** JIT (Just In Time compiler) enables high performance in Java. JIT converts the bytecode into machine language and then JVM starts the execution.
- **Multi-threaded:** A flow of execution is known as a Thread. JVM creates a thread which is called main thread. The user can create multiple threads by extending the thread class or by implementing Runnable interface.

Q #3) How does Java enable high performance?

Ans: Java uses Just In Time compiler to enable high performance. JIT is used to convert the instructions into bytecodes.

Q #4) What are the Java IDE's?

Ans: Eclipse and NetBeans are the IDE's of JAVA.

Q #5) What do you mean by Constructor?

Ans: The points given below explain what a Constructor is in detail:

- When a new object is created in a program a constructor gets invoked corresponding t
  class.
- The constructor is a method which has the same name as class name.

Menu

a parameter then he should create another
eter.

stance variable?

od and scope of the variables that have existed inside

ss and outside the method and scope of the variables

Class has variables and methods.

e of a class.

ess logic has to be done. It contains a set of
rticular requirement.

```
ne declaration

ation
```

Q #8) What is an Object?

Ans: An instance of a class is called object. The object has state and behavior.

Whenever the JVM reads the “new()” keyword then it will create an instance of that class.

Example:

```
1 public class Addition{
2 public static void main(String[] args){
3 Addion add = new Addition();//Object creation
4 }
5 }
```

The above code creates the object for the Addition class.

Q #9)What are the Oops concepts?

Ans: Oops concepts include:

- Inheritance
- Encapsulation
- Polymorphism
- Abstraction

l to another class. So that the codes can be reused

as the derived class is known as a sub class.

```
pulation(){
```

cted members only. Private members can't be

We are declaring 'a' as an integer variable and it should not be negative.

```
1 public class Addition(){
2 int a=5;
3 }
```

If someone changes the exact variable as "***a = -5***" then it is bad.

In order to overcome the problem we need to follow the below steps:

- We can make the variable as private or protected one.
- Use public accessor methods such as set<property> and get<property>.

So that the above code can be modified as:

```
1 public class Addition(){
2 private int a = 5; //Here the variable is marked as private
3 }
```

Below code shows the getter and setter.

Conditions can be provided while setting the variable.

nstance variables as private and create setter and
force others to call the setters rather than access the

ub-class depending on the reference type which is

```
er class

.pulation(){ // Sub class

;s[]){
.on();//Manipulation is reference type and
addition.add();
}
}
```

Using Manipulation reference type we can call the Addition class “add()” method. This ability is known as Polymorphism.

Polymorphism is applicable for **overriding** and not for **overloading**.

Q #13) What is meant by Method Overriding?

Ans: Method overriding happens if the sub class method satisfies the below conditions with the Super class method:

- Method name should be same
- Argument should be same
- Return type also should be same

The key benefit of overriding is that the Sub class can provide some specific information about that sub class type than the super class.

Example:

()){

olimorphism is applied
() method

d in the Sub class and not the parent class. So it
wn as Method Overriding.

ıt classes or within the same class.

ould satisfy the below conditions with the Super class
:

Example:

```
public class Manipulation{ //Super class
public void add(String name){ //String parameter
.....
}
}

Public class Addition extends Manipulation(){
Public void add(){//No Parameter
.....
}
Public void add(int a){ //integer parameter

}
Public static void main(String args[]){
Addition addition = new Addition();
addition.add();
}
}
```

Here the add() method having different parameters in the Addition class is overloaded in the class as well as with the super class.

in java. To overcome this problem Interface concept

thod declarations and not the method

```
ation{ //Interface declaration
d declaration
```

ernally **public abstract void**.
ernally **public static final** that is constants.
nd not extends.
ace should provide an implementation for all the

```
ts IManupulation{ //Manupulation class uses the
```

Q #16) What is meant by Abstract class?

Ans: We can create the Abstract class by using “Abstract” keyword before the class name. An abstract class can have both “Abstract” methods and “Non-abstract” methods that are a concrete class.

Abstract method:

The method which has only the declaration and not the implementation is called the abstract method and it has the keyword called “abstract”. Declarations are the ends with a semicolon.

Example:

```
1 public abstract class Manupulation{
2 public abstract void add();//Abstract method declaration
3 Public void subtract(){
4 }
5 }
```

- An abstract class may have a Non- abstract method also.
- The concrete Subclass which extends the Abstract class should provide the implementation for abstract methods.

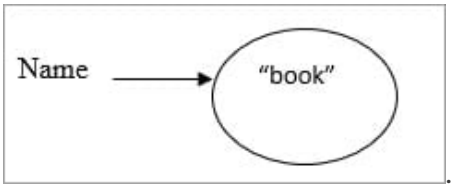
Q #17) Difference between Array and Array List.

	Array List
	Size may not be required. It changes the size dynamically. ArrayList name = new ArrayList
	No index required. name.add(“book”)
	ArrayList in java 5.0 are parameterized. Eg: This angle bracket is a type parameter which means a list of String.

der, and String Buffer.

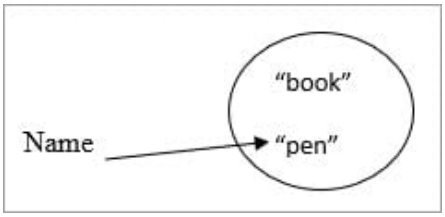
nstant string pool”. Once the string reference changes
ng pool”, it cannot be erased.

Constant string pool



If the name value has changed from “book” to “pen”.

Constant string pool



Then the older value retains in the constant string pool.

String Buffer:

Menu

is thread-safe.
Builder.

)en" then the "book" is erased in the stack.

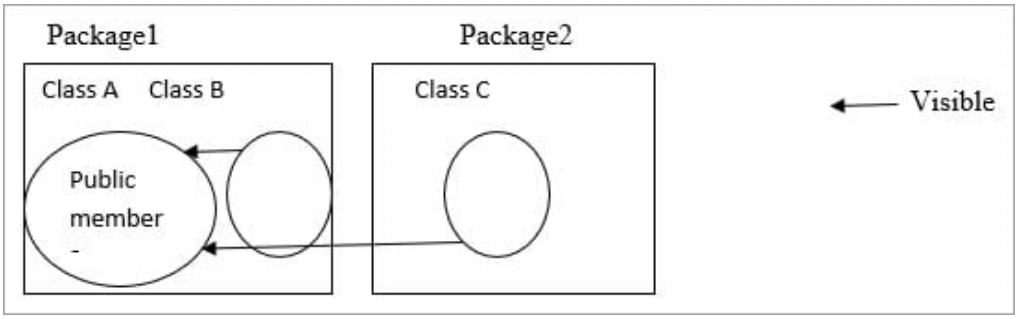
tring Builder which is not threaded safety that is not st.

ss specifiers.

vn as members.

Public:

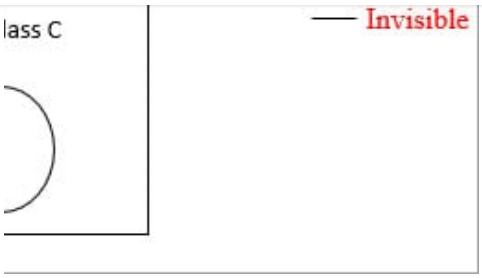
Public members are visible in the same package as well as the outside package that is for other packages.



Public members in Class A are visible to Class B (Same package) as well as Class C (Different package).

Private:

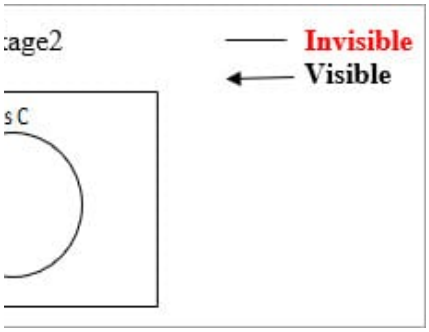
Private members are visible in the same class only and not for the other classes in the same package as well as classes in the outside packages.



hat class. It is invisible for class B as well as class C.

cted access specifiers.

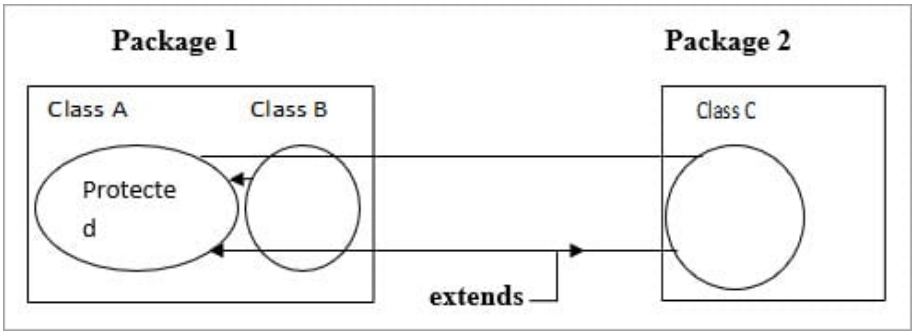
in a class without any access specifiers are called



other classes which are inside the package and package.

So Class A members are visible to the Class B and invisible to the Class C.

Protected:



Protected is same as Default but if a class extends then it is visible even if it is outside the package.

Class A members are visible to Class B because it is inside the package. For Class C it is invisible but if Class C extends Class A then the members are visible to the Class C even if it is outside the package.

Q #21) Difference between HashMap and HashTable.

HashTable
Key methods are synchronized
Thread safety
Enumerator is used to iterate the values
Doesn't allow anything that is null
Performance is slow

Set.

t can be seen below:

TreeSet
Maintains the elements in the sorted order
Couldn't store null objects
Performance is slow

d Interface.

nterface are as follows:

Abstract Class:

- Abstract classes have a default constructor and it is called whenever the concrete subclass is instantiated.
- Contains Abstract methods as well as Non-Abstract methods.
- The class which extends the Abstract class shouldn't require implementing all the methods, only Abstract methods need to be implemented in the concrete sub-class.
- Abstract Class contains instance variables.

Interface:

- Doesn't have any constructor and couldn't be instantiated.
- Abstract method alone should be declared.
- Classes which implement the interface should provide the implementation for all the methods.
- The interface contains only constants.

Q #24) What is mean by Collections in Java?

Ans: Collection is a framework that is designed to store the objects and manipulate the design to store the objects.

Collections are used to perform the following operations:

ll the classes and interfaces for collecting are

s that are available in the collections?

ces that are available in Collections:

Sets:

- Hash set
- Linked Hash Set
- Tree Set

Maps:

- Hash Map
- Hash Table
- Tree Map
- Linked Hashed Map

Queue:

- Priority Queue

Q #26) What is meant by Ordered and Sorted in collections?

Ans:

Menu

ction is based on the values that are added to the
the collection in a specific order.

or externally so that the group of objects sorted in a
f the objects.

able in the collection.

index position and it is ordered by index position.

d not sorted.
e.

tring>();

```
names.add (“cherry”);
names.add (“kiwi”);
names.add (“banana”);
names.add (“cherry”);
System.out.println (names);
}
}
```

Output:

[Apple, cherry, kiwi, banana, cherry]

From the output, Array List maintains the insertion order and it accepts the duplicates. But not sorted.

Vector:

It is same as Array List.

- Vector methods are synchronized.
- Thread safety.
- It also implements the Random Access.
- Thread safety usually causes a performance hit.

ıg> ( );

accepts the duplicates.

:her.

methods peek( ), Pool ( ), Offer ( ) etc.

```
public class Fruit {
public static void main (String [ ] args){
Linkedlist <String> names = new linkedlist <String> ( ) ;
names.add(“banana”);
names.add(“cherry”);
names.add(“apple”);
names.add(“kiwi”);
names.add(“banana”);
System.out.println (names);
}
}
```

Output

[banana,cherry,apple,kiwi,banana]

Maintains the insertion order and accepts the duplicates.

Q #28) Explain about Set and their types in a collection?

Ans: Set cares about uniqueness. It doesn’t allow duplications. Here “equals ( )” method is us
determine whether two objects are identical or not.

sert the values.
luplicates and don't care about the order".

String>( ) ;

are not allowed.

- An ordered version of the hash set is known as Linked Hash Set.
- Maintains a doubly-Linked list of all the elements.
- Use this when the iteration order is required.

Example:

```
public class Fruit {
public static void main (String[ ] args){
LinkedHashSet<String> names = new LinkedHashSet <String>( ) ;
 names.add("banana");
```

ave been added to the Set. Duplicates are not

uarantees that the elements will be in an ascending

onstructor by using comparable (or) comparator.

ng>( ) ;

```
names.add("kiwi");
names.add("cherry");
System.out.println(names);
}
}
```

Output:

[apple, banana, cherry, kiwi]

TreeSet sorts the elements in an ascending order. And duplicates are not allowed.

Q #29). Explain about Map and their types.

Ans: Map cares about unique identifier. We can map a unique key to a specific value. It is a key/value pair. We can search a value, based on the key. Like set, Map also uses "equals ( )" method to determine whether two keys are same or different.

Hash Map:

- Unordered and unsorted map.
- Hashmap is a good choice when we don't care about the order.

Menu

Map<String,String>( );

= apple}

nsorted.

re synchronized.
performance.

Example:

```
public class Fruit{
public static void main(String[ ]args){
Hashtable<Sting,String> names =new Hashtable<String,String>( );
names.put("key1","cherry");
names.put("key2","apple");
names.put("key3","banana");
names.put("key4","kiwi");
names.put("key2","orange");
System.out.println(names);
}
 }
```

Output:

{key2=apple, key1=cherry,key4=kiwi, key3=banana}

Duplicate keys are not allowed.

Linked Hash Map:

LinkedHashMap<String,String>( );

ıana}

ɔrder with the constructor.

Example:

```
public class Fruit{
public static void main(String[ ]args){
TreeMap<Sting,String> names =new TreeMap<String,String>( );
names.put("key1","cherry");
names.put("key2","banana");
names.put("key3","apple");
names.put("key4","kiwi");
names.put("key2","orange");
System.out.println(names);
}
}
```

Output:

{key1=cherry, key2=banana, key3 =apple, key4=kiwi}

It is sorted in ascending order based on the key. Duplicate keys are not allowed.

Q #30) Explain the Priority Queue.

Ans: Queue Interface

or according to the comparator. The elements

ır during the normal flow of an execution. A method
.s at runtime. If that exception couldn't be handled,
completes the task.

. flow gets continued. Exceptions are a subclass of

```
is block
lock
```

elow in detail.

er at the time of compilation. Classes that extend
nd Error are called checked Exception.

Checked Exceptions must either declare the exception using throes keyword (or) surrounded by appropriate try/catch.

E.g. ClassNotFound Exception

Unchecked Exception:

These exceptions are not checked during the compile time by the compiler. The compiler doesn't force to handle these exceptions.

It includes:

- Arithmetic Exception
- ArrayIndexOutOfBounds Exception

Q #33) What are the different ways to handle exceptions?

Ans: Two different ways to handle exception are explained below:

#1) Using try/catch:

```
irgs){
```

e exception using throws keyword.

```
irgs){

i{
```

n handling?

Ans: Given below are the advantages:

- The normal flow of the execution won't be terminated if exception got handled
- We can identify the problem by using catch declaration

Q #35) What are Exception handling keywords in Java?

Ans: Given below are the two Exception Handling Keywords:

try:

When a risky code is surrounded by a try block. An exception occurring in the try block is caught by a catch block. Try can be followed either by catch (or) finally (or) both. But any one of the blocks is mandatory.

catch:

This is followed by try block. Exceptions are caught here.

finally:

This is followed either by try block (or) catch block. This block gets executed regardless of ar exception. So generally clean up codes are provided here.

Menu

od which is at the top of the stack. If it doesn't catch,
e previous method and so on until they are got.

```
irgs){
```

e as shown below:

thod is not caught, then it moves to the method
l and then it will stop the flow of execution. It is

Ans:

Final variable:

Once a variable is declared as final, then the value of the variable could not be changed. It is like a constant.

Example:

final int = 12;

Final method:

A final keyword in a method that couldn't be overridden. If a method is marked as a final, then it can't be overridden by the subclass.

Final class:

If a class is declared as final, then the class couldn't be subclassed. No class can extend the final class.

Q #38) What is a Thread?

Runnable interface. Threads are executed

```
irgs){//main thread starts here
```

make a thread.

n method. The thread is available in java.lang.thread.

```
ead {
```

hat we cannot extend any other classes because we
overload the run () method in our class.

terface. For that we should provide the
implementation for run () method which is defined in the interface.

Example:

```
Public class Addition implements Runnable {
public void run () {
}
}
```

Q #40) Explain about join () method.

Ans: Join () method is used to join one thread with the end of the currently running thread.

Example:

```
public static void main (String[] args){
Thread t = new Thread ();
t.start ();
t.join ();
}
```

From the above code, the main thread started the execution. When it reaches the code ***t.start()*** then 'thread t' starts the own stack for the execution. JVM switches between the main thread and 'thread t'.

overloaded version. So we can mention the time

l class do?

unning thread to a runnable state and allows the iority threads have a chance to run. It is a static

e Runnable state only, and not the thread to sleep (),

```
args){
```

ad to wait in the waiting pool. When a wait () method nmediately the thread gives up the lock on the object tells the thread to wait for a given amount of time.

Then the thread will wake up after notify () (or) notify all () method is called.

Wait() and the other above-mentioned methods do not give the lock on the object immediately until the currently executing thread completes the synchronized code. It is mostly used in synchronization.

Example:

```
1 public static void main (String[] args){
2 Thread t = new Thread ();
3 t.start ();
4 Synchronized (t) {
5 Wait();
6 }
7 }
```

Q #43) Difference between notify() method and notifyAll() method in Java.

Ans: Given below are few differences between notify() method and notifyAll() method

notify()	notifyAll()
This method is used to send a signal to wake up a single thread in the waiting pool.	This method sends the signal to wake u the threads in a waiting spool.

Menu

ving thread methods.

y executing thread for the given amount of time. Once
nable state. So sleep () method is used to delay the

ls. Sleep () method throws an uninterrupted exception,
y/catch.

```
nts Runnable{
 args){
```

```
}catch(InterruptedException e){
}
}
```

Q #45) When to use Runnable interface Vs Thread class in Java?

Ans: If we need our class to extend some other classes other than the thread then we can go with the runnable interface because in java we can extend only one class.

If we are not going to extend any class then we can extend the thread class.

Q #46) Difference between start() and run() method of thread class.

Ans: Start() method creates new thread and the code inside the run () method is executed in the new thread. If we directly called the run() method then a new thread is not created and the currently executing thread will continue to execute the run() method.

Q #47) What is Multi-threading?

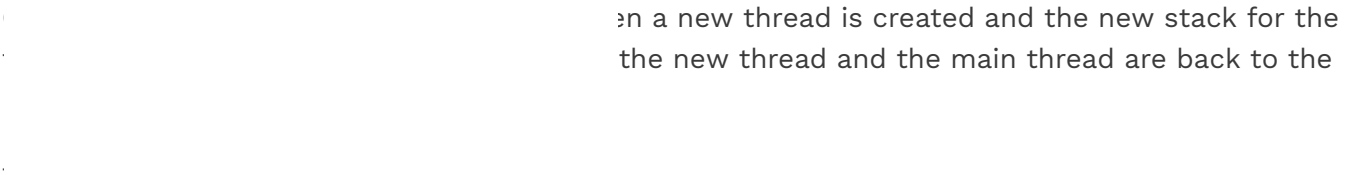
Ans: Multiple threads are executed simultaneously. Each thread starts their own stack based on the flow (or) priority of the threads.

Example Program:

Menu

```
ere
```

method and the main thread stack looks as shown



en a new thread is created and the new stack for the
the new thread and the main thread are back to the

Now, the user thread executed the code inside the run() method.

Menu

JVM switches back to the main thread and the User was disappeared.

n the threads are completed. This is called Multi-

ed but start () method is not yet invoked. Now the thread is not considered alive.

- **Runnable**:

The Thread is in runnable state after invocation of the start () method, but before the run () method is invoked. But a thread can also return to the runnable state from waiting/sleeping. In this state the thread is considered alive.

- **Running**:

The thread is in running state after it calls the run () method. Now the thread begins the execution.

- **Non-Runnable**(Blocked):

The thread is alive but it is not eligible to run. It is not in runnable state but also, it will return to runnable state after some time.

Example: wait, sleep, block.

- **Terminated** :

Once the run method is completed then it is terminated. Now the thread is not alive.

Menu

to access a block of code at a time. If multiple thread
hance for inaccurate results at the end. To avoid this
sensitive block of codes.

ad needs a key in order to access the synchronized

a lock. A lock has only one key. A thread can access a
get the key to the objects lock.

unnable{

ization?

Ans: Synchronization is not recommended to implement all the methods. Because if one thread accesses the synchronized code then the next thread should have to wait. So it makes slow performance on the other end.

Q #51) What is meant by Serialization?

Ans: Converting a file into a byte stream is known as Serialization. The objects in the file is converted to the bytes for security purposes. For this, we need to implement java.io.Serializable interface. It has no method to define.

Variables that are marked as transient will not be a part of the serialization. So we can skip the serialization for the variables in the file by using a transient keyword.

Q #52) What is the purpose of a transient variable?

Ans: Transient variables are not part of the serialization process. During deserialization, the transient variables values are set to default value. It is not used with static variables.

Example:

transient int numbers;

Q #53) Which methods are used during Serialization and Deserialization process?

e the object and write the serialized object to a file.

file and deserializes the object.

ne serializable interface. If superclass implements
ally be serializable.

able?

from the main memory and not from thread's cache
ization. It is applicable only for variables.

Deserialization in Java.

ization and deserialization in java:

	eserialization
	eserialization is the opposite process of serialization here we can get the objects back from the byte ream.
An object is serialized by writing it an ObjectOutputStream.	An object is deserialized by reading it from an ObjectInputStream.

Q #56) What is SerialVersionUID?

Ans: Whenever an object is Serialized, the object is stamped with a version ID number for the object class. This ID is called the SerialVersionUID. This is used during deserialization to verify that the sender and receiver that are compatible with the Serialization.

Conclusion

These are some of the core JAVA interview questions that cover both the basic and advanced Java concepts for programming as well as developer interview, and these are ones which have been answered by our JAVA experts.

I hope that this tutorial would have given you a great insight into JAVA core coding concepts in detail. The explanations given above will really enrich your knowledge and increase your understanding of JAVA programming.

Get ready to crack an interview confidently !!!