Exception and Thread Java Interview Questions

Q1. What is the difference between Error and Exception?

An error is an irrecoverable condition occurring at runtime. Such as OutOfMemory error. These JVM errors you cannot repair them at runtime. Though error can be caught in the catch block but the execution of application will come to a halt and is not recoverable.

While exceptions are conditions that occur because of bad input or human error etc. e.g. FileNotFoundException will be thrown if the specified file does not exist. Or a NullPointerException will take place if you try using a null reference. In most of the cases it is possible to recover from an exception (probably by giving the user feedback for entering proper values etc.

Q2. How can you handle Java exceptions?

There are five keywords used to handle exceptions in Java:

1. try
2. catch
3. finally
4. throw
5. throws

Q3. What are the differences between Checked Exception and Unchecked Exception?

Checked Exception

- The classes that extend Throwable class except RuntimeException and Error are known as checked exceptions.
- Checked exceptions are checked at compile-time.
- Example: IOException, SQLException etc.

Unchecked Exception

- The classes that extend RuntimeException are known as unchecked exceptions.
- Unchecked exceptions are not checked at compile-time.
- Example: ArithmeticException, NullPointerException etc.

Q4. What are the different ways of thread usage?

There are two ways to create a thread:

- Extending Thread class

This creates a thread by creating an instance of a new class that extends the Thread class. The extending class must override the run() function, which is the thread's entry point.

- Implementing Runnable interface

This is the easiest way to create a thread, by creating a class that implements the runnable interface. After implementing the runnable interface, the class must implement the public void run() method ()

The run() method creates a parallel thread in your programme. When run() returns, the thread will come to an end.

The run() method creates a parallel thread in your programme. When run() returns, the thread will come to an end.

Within the run() method, you must specify the thread's code.

Like any other method, the run() method can call other methods, use other classes, and define variables.

Java works as "pass by value" or "pass by reference" phenomenon?

Java is always pass-by-value. This means that it creates a copy of the contents of the parameter in memory. In Java, object variables always refer to the memory heap's real object.

Q5. Will the finally block get executed when the return statement is written at the end of try block and catch block as shown below?

The finally block always gets executed even hen the return statement is written at the end of the try block and the catch block. It always executes , whether there is an exception or not. There are only a few situations in which the finally block does not execute, such as VM crash, power failure, software crash, etc. If you don't want to execute the finally block, you need to call the System.exit() method explicitly in the finally block.

Q6. How does an exception propagate in the code?

If an exception is not caught, it is thrown from the top of the stack and falls down the call stack to the previous procedure. If the exception isn't caught there, it falls back to the previous function, and so on, until it's caught or the call stack reaches the bottom. The term for this is Exception propagation.

Q7. Can you explain the Java thread lifecycle?

The java thread lifecycle has the following states-

New-

When a thread is created, and before the program starts the thread, it is in the new state. It is also referred to as a born thread.

Runnable

When a thread is started, it is in the Runnable state. In this state, the thread is executing its task.

Waiting

Sometimes, a thread goes to the waiting state, where it remains idle because another thread is executing. When the other thread has finished, the waiting thread again comes into the running state.

Timed Waiting

In timed waiting, the thread goes to waiting state. But, it remains in waiting state for only a specified interval of time after which it starts executing.It remains waiting either till the time interval ends or till the other thread has finished.

Terminated

A thread is said to be in this state once it terminates. It may be because the thread has completed its task or due to any other reason.

Q8. What purpose do the keywords final, finally, and finalize fulfill?

Final:

Final is used to apply restrictions on class, method, and variable. A final class can't be inherited, final method can't be overridden and final variable value can't be changed. Let's take a look at the example below to understand it better.

```
class FinalVarExample {
public static void main( String args[])
{
final int a=10; // Final variable
a=50; //Error as value can't be changed
}
```

Finally

Finally is used to place important code, it will be executed whether the exception is handled or not. Let's take a look at the example below to understand it better.

```
class FinallyExample {
```

```
public static void main(String args[]){
try {
int x=100;
}
catch(Exception e) {
System.out.println(e);
}
finally {
System.out.println("finally block is executing");}
}}
}
```

Finalize

Finalize is used to perform clean up processing just before the object is garbage collected. Let's take a look at the example below to understand it better.

```
class FinalizeExample {
public void finalize() {
System.out.println("Finalize is called");
}
public static void main(String args[])
{
FinalizeExample f1=new FinalizeExample();
FinalizeExample f2=new FinalizeExample();
f1= NULL;
f2=NULL;
System.gc();
}
}
```

Q9. What are the differences between throw and throws?

throw keyword	throws keyword
Throw is used to explicitly throw an exception.	Throws is used to declare an exception.

Checked exceptions can not be propagated with throw only.	Checked exception can be propagated with th
Throw is followed by an instance.	Throws is followed by class.
Throw is used within the method.	Throws is used with the method signature.
You cannot throw multiple exception	You can declare multiple exception e.g. public method()throws IOException,SQLException.

In case you are facing any challenges with these java interview questions, please comment on your problems in the section below.

Q10. What is exception hierarchy in java?

The hierarchy is as follows:

Throwable is a parent class of all Exception classes. There are two types of Exceptions: Checked exceptions and UncheckedExceptions or RunTimeExceptions. Both type of exceptions extends Exception class whereas errors are further classified into Virtual Machine error and Assertion error.

Q11. How to create a custom Exception?

To create you own exception extend the Exception class or any of its subclasses.

- class New1Exception extends Exception { } // this will create Checked Exception
- class NewException extends IOException { } // this will create Checked exception
- class NewException extends NullPonterExcpetion { } // this will create UnChecked exception

Q12. What are the important methods of Java Exception Class?

Exception and all of it's subclasses doesn't provide any specific methods and all of the methods are defined in the base class Throwable.

1. **String getMessage()** – This method returns the message String of Throwable and the message can be provided while creating the exception through it's constructor.
2. **String getLocalizedMessage()** – This method is provided so that subclasses can override it to provide locale specific message to the

calling program. Throwable class implementation of this method simply use getMessage() method to return the exception message.

3. **Synchronized Throwable getCause()** – This method returns the cause of the exception or null id the cause is unknown.
4. **String toString()** – This method returns the information about Throwable in String format, the returned String contains the name of Throwable class and localized message.
5. **void printStackTrace()** – This method prints the stack trace information to the standard error stream, this method is overloaded and we can pass PrintStream or PrintWriter as an argument to write the stack trace information to the file or stream.

Q13. What are the differences between processes and threads?

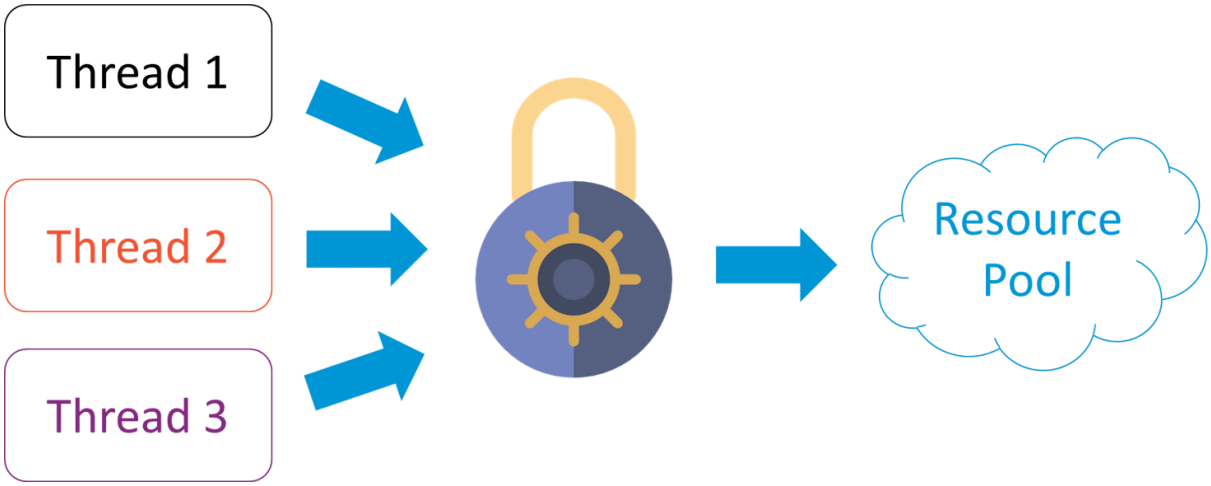
	Process	Thread
Definition	An executing instance of a program is called a process.	A thread is a subset of the process.
Communication	Processes must use inter-process communication to communicate with sibling processes.	Threads can directly communicate wi threads of its process.
Control	Processes can only exercise control over child processes.	Threads can exercise considerable co over threads of the same process.
Changes	Any change in the parent process does not affect child processes.	Any change in the main thread may a behavior of the other threads of the p
Memory	Run in separate memory spaces.	Run in shared memory spaces.
Controlled by	Process is controlled by the operating system.	Threads are controlled by programm program.
Dependence	Processes are independent.	Threads are dependent.

Q14. What is a finally block? Is there a case when finally will not execute?

Finally block is a block which always executes a set of statements. It is always associated with a try block regardless of any exception that occurs or not. Yes, finally will not be executed if the program exits either by calling System.exit() or by causing a fatal error that causes the process to abort.

Q15. What is synchronization?

Synchronization refers to multi-threading. A synchronized block of code can be executed by only one thread at a time. As Java supports execution of multiple threads, two or more threads may access the same fields or objects. Synchronization is a process which keeps all concurrent threads in execution to be in sync. Synchronization avoids memory consistency errors caused due to inconsistent view of shared memory. When a method is declared as synchronized the thread holds the monitor for that method's object. If another thread is executing the synchronized method the thread is blocked until that thread releases the monitor.



Q16. Can we write multiple catch blocks under single try block?

Yes we can have multiple catch blocks under single try block but the approach should be from specific to general. Let's understand this with a programmatic example.

```
public class Example {
public static void main(String args[]) {
try {
int a[]= new int[10];
a[10]= 10/0;
}
catch(ArithmeticException e)
{
System.out.println("Arithmetic exception in first catch block");
}
```

```
catch(ArrayIndexOutOfBoundsException e)
{
System.out.println("Array index out of bounds in second catch block");
}
catch(Exception e)
{
System.out.println("Any exception in third catch block");
}
}
```

Q17. What are the important methods of Java Exception Class?

Methods are defined in the base class Throwable. Some of the important methods of Java exception class are stated below.

1. **String getMessage()** – This method returns the message String about the exception. The message can be provided through its constructor.
2. **public StackTraceElement[] getStackTrace() –** This method returns an array containing each element on the stack trace. The element at index 0 represents the top of the call stack whereas the last element in the array represents the method at the bottom of the call stack.
3. **Synchronized Throwable getCause()** – This method returns the cause of the exception or null id as represented by a Throwable object.
4. **String toString()** – This method returns the information in String format. The returned String contains the name of Throwable class and localized message.
5. **void printStackTrace()** – This method prints the stack trace information to the standard error stream.

Q18. What is OutOfMemoryError in Java?

OutOfMemoryError is the subclass of java.lang.Error which generally occurs when our JVM runs out of memory.

Q19. What is a Thread?

A thread is the smallest piece of programmed instructions which can be executed independently by a scheduler. In Java, all the programs will have at least one thread which is known as the main thread. This main thread is created by the JVM

when the program starts its execution. The main thread is used to invoke the main() of the program.

Q20. What are the two ways to create a thread?

In Java, threads can be created in the following two ways:-

- By implementing the Runnable interface.
- By extending the Thread

Q21. What are the different types of garbage collectors in Java?

Garbage collection in Java a program which helps in implicit memory management. Since in Java, using the new keyword you can create objects dynamically, which once created will consume some memory. Once the job is done and there are no more references left to the object, Java using garbage collection destroys the object and relieves the memory occupied by it. Java provides four types of garbage collectors:

- Serial Garbage Collector
- Parallel Garbage Collector
- CMS Garbage Collector
- G1 Garbage Collector