1. **Explain System.out.println()**

System is static class in the java.lang package and cannot be instantiated

out is a reference variable defined in System

println() is the method used to print on standard output

1. **Error vs Exception**

🡪An error indicates serious problems that a reasonable application should not try to catch. (eg. memory error, hardware error, JVM error)

🡪Exceptions are events that occur during the execution of programs that disrupt the normal flow of instructions (e.g. divide by zero, array access out of bound, etc.

|  |  |
| --- | --- |
| **Error** | **Exception** |
| Errors in java are of type java.lang.Error. | Exceptions in java are of type java.lang.Exception. |
| All errors in java are unchecked type. | Exceptions include both checked as well as unchecked type. |
| Errors happen at run time. They will not be known to compiler. | Checked exceptions are known to compiler where as unchecked exceptions are not known to compiler because they occur at run time. |
| It is impossible to recover from errors. | You can recover from exceptions by handling them through try-catch blocks. |
| Errors are mostly caused by the environment in which application is running. | Exceptions are mainly caused by the application itself. |
| Examples : java.lang.StackOverflowError, java.lang.OutOfMemoryError | Examples :  Checked Exceptions : SQLException, IOException Unchecked Exceptions : ArrayIndexOutOfBoundException, ClassCastException, NullPointerException |

1. **Types of exceptions**

Exception can occur at runtime (known as runtime exceptions) as well as at compile-time (known Compile-time exceptions).

**Checked exceptions**  
All exceptions other than Runtime Exceptions are known as Checked exceptions as the compiler checks them during compilation to see whether the programmer has handled them or not. If these exceptions are not handled/declared in the program, it will give compilation error.

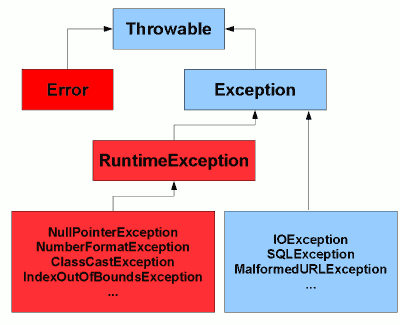
Examples of Checked Exceptions:-  
ClassNotFoundException  
IllegalAccessException  
NoSuchFieldException  
EOFException etc.

**Unchecked Exceptions**  
Runtime Exceptions are also known as Unchecked Exceptions as the compiler do not check whether the programmer has handled them or not but it’s the duty of the programmer to handle these exceptions and provide a safe exit.  
These exceptions need not be included in any method’s throws list because compiler does not check to see if a method handles or throws these exceptions.

**Examples of Unchecked Exceptions:-**  
ArithmeticException  
ArrayIndexOutOfBoundsException  
NullPointerException  
NegativeArraySizeException etc.

1. **Is error an unchecked exception?**

Error is unchecked. But it isn't an exception.



1. **What is concurrent hashmap?**

Concurrent Hash Map allow concurrent access to the map. Hash Tables too offers synchronized access to map, but your entire map is locked to perform any operation.

The logic behind Concurrent Hash Map is that your entire table is not getting locked, but only the portion[segments]. Each segments manages its own Hash Table. Locking is applied only for updates. In case of of retrievals, it allows full concurrency.

Let's take four threads are concurrently working on a map whose capacity is 32, the table is partitioned into four segments where each segments manages a hash table of capacity. The collection maintains a list of 16 segments by default, each of which is used to guard (or lock on) a single bucket of the map.

1. **When to use interface vs abstract class ?**

Where the client wants to only deal with a type and does not care about the actual implementation use interfaces. If you need to change your design frequently, you should prefer using interface to abstract.

1. **Name the scopes in JSP.**

The pageScope, requestScope, sessionScope, and applicationScope variables provide access to variables stored at each scope level.

1. **What are marker interfaces?**

The interfaces with no defined methods act like markers. They just tell the compiler that the objects of the classes implementing the interfaces with no defined methods need to be treated differently. Example Serializable, Cloneable etc.

1. **What is serialization? When to use it?**

Serialization is a process of reading or writing an object. It is a process of saving an object’s state to a sequence of bytes, as well as a process of rebuilding those bytes back into a live object at some future time.

An object is marked serializable by implementing the java.io.Serializable interface, which is only a marker interface -- it simply allows the serialization mechanism to verify that the class can be persisted, typically to a file.

A common use of serialization is to use it to send an object over the network or if the state of an object needs to be persisted to a flat file or a database

1. **How can you start garbage collection?**

You can nicely ask the garbage collector to collect garbage by calling System.gc() but you can’t force it.

1. **Can you override private or static method in Java?**

NO.

The point of polymorphism is that you can subclass a class and the objects implementing those subclasses will have different behaviors for the same methods defined in the superclass (and overridden in the subclasses). A static method is not associated with any instance of a class so the concept is not applicable.

1. **Define JSON?**

Expansion of JSON is “JavaScript Object Notation”. It is a much lighter and readable alternative to XML. It is an independent and easily parse-able in all [programming languages](https://intellipaat.com/big-data-data-science-training/). It is primarily used for Communicating between client – server or server -server communication. It is a much lighter and readable alternative to XML.

1. **Explain the Struts framework lifecycle**

Following is the life cycle of a request in Struct2 application −

* User sends a request to the server for requesting for some resource (i.e pages).
* The FilterDispatcher looks at the request and then determines the appropriate Action.
* Configured interceptors functionalities applies such as validation, file upload etc.
* Selected action is executed to perform the requested operation.
* Again, configured interceptors are applied to do any post-processing if required.
* Finally the result is prepared by the view and returns the result to the user.

1. **Explain the mappings in hibernate**

|  |  |
| --- | --- |
| [**Many-to-One**](https://www.tutorialspoint.com/hibernate/hibernate_many_to_one_mapping.htm) | Mapping many-to-one relationship using Hibernate |
| [**One-to-One**](https://www.tutorialspoint.com/hibernate/hibernate_one_to_one_mapping.htm) | Mapping one-to-one relationship using Hibernate |
| [**One-to-Many**](https://www.tutorialspoint.com/hibernate/hibernate_one_to_many_mapping.htm) | Mapping one-to-many relationship using Hibernate |
| [**Many-to-Many**](https://www.tutorialspoint.com/hibernate/hibernate_many_to_many_mapping.htm) | Mapping many-to-many relationship using Hibernate |

1. **What is HQL**

Hibernate Query Language (HQL) is an object-oriented query language, similar to SQL, but instead of operating on tables and columns, HQL works with persistent objects and their properties. HQL queries are translated by Hibernate into conventional SQL queries which in turns perform action on database.

1. **When to use volatile variable in Java?**

Volatile keyword is used with only variable in Java and it guarantees that value of volatile variable will always be read from main memory and not from Thread's local cache. So we can use volatile to achieve synchronization because it’s guaranteed that all reader thread will see updated value of volatile variable once write operation completed, without volatile keyword different reader thread may see different values. Volatile modifier also helps to prevent reordering of code by compiler and offer visibility guarantee by happens-before relationship.

1. **When to use transient variable in Java?**

Transient in Java is said to indicate that the variable should not be serialized. Serialization is a process of saving an object's state in Java. When we want to persist and object's state by default all instance variables in the object is stored. In some cases, if you want to avoid persisting some variables because we don’t have the necessity to transfer across the network. So, declare those variables as transient. If the variable is declared as transient, then it will not be persisted.

1. **Git vs CVS**

The main difference is that CVS is (old) centralized version control system, while Git is distributed.  Git stores repository in .git directory in top directory of your project; CVS require setting up CVSRO. File renames are not supported in CVSOT, a central place for storing version control info for different projects (modules)

1. **Why use angular over JavaScript?**

JavaScript is a scripting language. The other is a framework/ library. It extends the capabilities of JavaScript, adding new classes and functions.

1. **Can you make an abstract class as final?**

An abstract class cannot be declared as "final", because of following reasons: If we declare a class as abstract then we can't instantiate that class (only we can extend that class). And if create a class as final then we can't extend that class.

1. **Types of action classes in Struts**

Types of action class:

We have following types of action classes in struts:

**Action** - The basic action class in which we implement our business logic.

**Include Action** - Similar as include page directive in jsp.

**Forward Action** - Used in case we need to forward the request from one JSP to another. If we directly forward the request from one jsp it violates the MVC architecture. Hence an action class is used to do this job.

**Dispatch Action** - Handles multiple operations in multiple methods. It is better to have one method per operation instead of merging the entire business logic in a single execute method of an action class.

**Look up Dispatch Action -** Same as dispatch action but recommended not to use.

**Switch Action** - used to switch between different modules in struts application.

1. **What is the advantage of Hibernate over jdbc?**

The advantages of Hibernate over JDBC are:

1. Hibernate code will work well for all databases, for ex: Oracle, MySQL, etc. whereas JDBC is database specific.  
  
2. No knowledge of SQL is needed because Hibernate is a set of objects and a table is treated as an object, where as to work with JDBC, one need to know SQL.  
  
3. Query tuning is not required in Hibernate. The query tuning is automatic in hibernate by using criteria queries, and the result of performance is at its best. Where as in JDBC the query tuning is to be done by the database authors.  
  
4. With the support of cache of hibernate, the data can be placed in the cache for better performance. Where as in JDBC the java cache is to be implemented.

1. **Difference between Abstraction and Encapsulation**

Abstraction and encapsulation are complementary concepts. On the one hand, abstraction focuses on the behavior of an object. On the other hand, encapsulation focuses on the implementation of an object’s behavior. Encapsulation is usually achieved by hiding information about the internal state of an object and thus, can be seen as a strategy used in order to provide abstraction.

1. **So why do we need garbage collection? Many people can easily tell the advantages of garbage collection, but get confused when asked about the disadvantages.**

The most obvious benefit of having garbage collection is that it makes programming much easier. More specifically, garbage collection helps developers avoid several memory problems. First, it prevents accessing dangling pointers that point to an object that no longer exists. Secondly, it prevents freeing a region of memory that is already freed. Last, it avoids memory leak, which means an unreachable region of memory that can never be freed. All of them are common pitfalls when developers try to manage memory manually.

So what are the disadvantages? Apparently, there are many languages that don’t have built-in garbage collection like C++.

The biggest disadvantage is that garbage collection consumes computing resources. Think about this, not only does garbage collection need to implement logics to recycle memory, it also consumes memory to store the status of objects. In some naive garbage collection implementation, the recycle process may even block the program potentially.

1. **Is Java pass-by reference or value ?**

Other languages use pass-by-reference or pass-by-pointer. But in Java no matter what type of argument you pass the corresponding parameter (primitive variable or object reference) will get a copy of that data, which is exactly how pass-by-value (i.e. copy-by-value) works.