**Interview Questions at Genpact Headstrong for Java**

**Q. What will happen if you call return statement or System.exit on try or catch block? will finally block execute?**

finally block will execute even if you put return statement in try block or catch block  
 finally block won't run if you call System.exit from try or catch.

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

No. We cannot override private or static method in Java, if you create similar method with same return type and same method arguments that's called method hiding. (Parent and Child class concept).  
  
**Q. JSP Implicit Objects or Objects (Written Inside Scriplets)?**  
  
·         Implicit object are the object that are created by web container provides to a developer to access them in their program using JavaBeans and Servlets.  
·         These objects are called implicit objects because they are automatically instantiated.  
·         They are by default available in JSP page.  
1.       Application  
2.       config  
3.       Exception  
4.       out  
5.       page  
6.       PageContext  
7.       request  
8.       response  
9.       session.  
  
**Q. Why Java doesn't support multiple inheritance?**  
  
1) First reason is ambiguity (not clear or decided) around Diamond problem, consider a class A has foo() method and then B and C derived from A and has their own foo() implementation and now class D derive from B and C using multiple inheritance and if we refer just foo() compiler will not be able to decide which foo() it should invoke. This is also called Diamond problem because structure on this inheritance scenario is similar to 4 edge diamond, see below  
           A foo()  
           / \  
          /   \  
   foo() B     C foo()  
          \   /  
           \ /  
            D  
           foo()  
  
2). Second and more convincing reason to me is that multiple inheritances does complicate the design and creates problem during casting, constructor chaining etc. and given that there are not many scenario on which you need multiple inheritance its wise decision to omit it for the sake of simplicity. Also java avoids this ambiguity by supporting single inheritance with interfaces. Since interface only have method declaration and doesn't provide any implementation there will only be just one implementation of specific method hence there would not be any ambiguity.  
   
**Q. Difference between == and equals?**  
  
·         ‘==’ check’s if references are equal means if both have same address then it return’s true.  
·         ‘equals’ check’s value irrespective of address.  
·         Default implementation of equals() class provided by java.lang.Object compares memory location and only return true if two reference variable are pointing to same memory location i.e. essentially they are same object.  
  
1) If two objects are equal by equals() method then there hashcode must be same.  
2) If two objects are not equal by equals() method then there hashcode could be same or different.  
Main difference between == and equals in Java is that "==" is used to compare primitives while equals() method is recommended to check equality of objects.  
Another difference between them is that, If both "==" and equals() is used to compare objects than == returns true only if both references points to same object while equals() can return true or false based on its overridden implementation.  
Ex. String str="abc";  
   String str1="abc";  
Both case will return true.  
Default Implementation of Equals  
public boolean equals(Object obj) {  
      return (this == obj);  
    }  
  
  
  
  
  
**Q. Why Synchronization in Java?**  
  
·         Because Java is a multi -threaded language where multiple threads run in parallel to complete program execution.  
·         In multi-threaded environment synchronization of java object or synchronization of java class becomes extremely important.  
·         Synchronization in Java is possible by using java keyword "synchronized" and "volatile”.  
·         Prevents access of multiple Threads to shared resource at a time.  
·        Synchronized is a modifier.  
JVM guarantees that Java synchronized code will only be executed by one thread at a time.  
·         Volatile can be used only with variables.  
·         Synchronized only when applying block level and method level locks.  
·         With Block level, we need to pass Class as an agreement:  
      if (obj == null) {  
                  synchronized (ClassName.class) //Mandatory{  
                        obj = new Hello();  
                  }  
            }

**Q. this keyword?**  
  
·         Can be used to refer current class instance variable.  
·         this() can be used to invoke current class Constructor. And it should be the first statement. If we are calling in class constructor using first line, it is giving compilation errorè recursive calling of constructor.   
  
public class Test {  
      public Test() {  
    this();   // Recursive Constructor invocation Test()  
}     }  
·         this can be used to invoke current class method.  
·         this can be passed as an argument in method call.  
·         this can be passed as an argument in Constructor call.  
·         this() can be used to return current class instance.  
·         Cannot be used in static context.  
   
public class Test {  
      public Test() {  
  
      }  
  
      private void test(String name) {  
            this(); // Compile time error - Constructor call must be the first statement in constructor.  
      }  
  
}  
  
**Q. What are the different ways to create singleton ?**  
[**Answer**](http://www.journaldev.com/1377/java-singleton-design-pattern-best-practices-with-examples)  
**Q. super keyword in Java?**  
  
Java keyword that refers to the immediate parent class object.  

**Q : What are basic features of java ?**  
  
[Abstraction](http://www.tutorialspoint.com/java/java_abstraction.htm)  
[Polymorphism](http://www.tutorialspoint.com/java/java_polymorphism.htm)  
[Encapsulation](http://www.tutorialspoint.com/java/java_encapsulation.htm)  
  
**Q : Difference between JDK, JVM and JRE ?**  
  
[Answer](http://stackoverflow.com/questions/11547458/what-is-the-difference-between-jvm-jdk-jre-openjdk)  
  
**Q : What is Garbage Collection. ?**  
  
[Garbage Collection inside view](http://www.oracle.com/webfolder/technetwork/tutorials/obe/java/gc01/index.html)  
  
**Q : Can System.gc() start the java garbage collection process?**  
  
Whether it will run the Garbage collector or not, cannot be confirmed. It is not guaranteed that it will start the garbage collection.

**Data Structures and Collections**

**Q: How Hashmap works internally ?**

[Reference 1](http://www.javacodegeeks.com/2014/03/how-hashmap-works-in-java.html)

[Reference 2](http://javahungry.blogspot.com/2013/08/hashing-how-hash-map-works-in-java-or.html)

**Q : What is an iterator ?**

[Java Docs](http://docs.oracle.com/javase/7/docs/api/java/util/Iterator.html)

**Q : What is the difference between Iterator and List Iterator ?**

[List Iterator](http://docs.oracle.com/javase/7/docs/api/java/util/ListIterator.html)

[Iterator](http://docs.oracle.com/javase/7/docs/api/java/util/Iterator.html)

**Q : How hashMap works internally in Java ? Can we safely use hashMap in multithreading environment ?**  
  
[Reference\_1](http://www.javacodegeeks.com/2014/03/how-hashmap-works-in-java.html)  
  
No we cannot use HashMap in multithreading environment.  
  
  
**Q : What will happen if we override only hashcode method but not equals method while creating my custom class (say Employee) whose instances we want to store as key in hashmap.**  
  
Overriding only hashcode method and not the equals method will not give any compile time or runtime error but it is a logical error. For example if you do not override equals method, then different Employee class instances will be compared by using equals method present in Object class and it will mark the objects as duplicate only when two objects are pointing to one instance of Employee class. However the comparison between two Employe objects should be based on custom parameters like employee id, name, designation etc. Comparison based on custom parameters is done by overriding the equals method along with hashcode method.

# ***1. Implement equals and hashcode methods  for  demo class***

*. Class Demo {    private String key;    private int value; } 2. Interface Demo{ m1(); m2(); } Class A implements Demo{ m1(); m2(); m3(): } main(){    Demo object = new A();     object.m3(); } 3. What happens when Immutable class contains mutable reference as instance variable. How to avoid the issues with it. 4. Hashmap working , collision. 5.spring bean definitions 6. Caching in hibernate. 2nd round Types of class loaders. Rest and soap difference Jre and JVM How will be the out of memory will be tackled after reaching max size available Difference between logical and physical database Which design patterns are been used in your project What is build design pattern For implementing caching in application, which kind of collection do you use Difference between list and set Implementation of custom set in java What all required changes are done on class if it is being used as key in collection What need to be considered inside implementation of equals and hascode Contact equals and hashcode. Explain executor framework. Difference between interface and abstract class. Why we explicitly give public static final for constants if JVM internally treats this as same in interface. What does final keyword actually does to an object - internal working How do you display duplicates with their count using collection.*