***Core Java Interview Questions and Answers***

*1)When you go for Abstract class and when you go for Interface ? Explain in detail with sample programs.  
Ans) If few properties of a object changes from time to time, we go for abstract class.*

*If all properties of a object changes from time to time, we go for interface.*

*Example Code Snippets:-*

***Abstract Class****...*

*abstract class Car*

*{*

*void engine()*

*{*

*System.out.println("this is different for each car");*

*}*

*}*

*class BMW extends Car*

*{*

*void tyres()*

*{*

*System.out.println("car has 4 tyres");*

*}*

*void engine()*

*{*

*System.out.println("BMW mega\_Et10.25 engine");*

*}*

*public static void main(String[] args)*

*{*

*BMW c=new BMW();*

*c.tyres();*

*c.engine();*

*}*

*}*

***Interface****:-*

*interface Petrol*

*{*

*public void petrol\_Price();*

*public void price\_India();*

*public void price\_State();*

*}*

*class Price implements Petrol*

*{*

*public void petrol\_Price()*

*{*

*System.out.println("Petrol price:75.50Rupees");*

*}*

*public void price\_India()*

*{*

*System.out.println("Petrol price:75.50Rupees");*

*}*

*public void price\_State()*

*{*

*System.out.println("Petrol price:75.50Rupees");*

*}*

*public static void main(String[] args)*

*{*

*Price p=new Price();*

*p. petrol\_Price();*

*p.price\_India();*

*p.price\_State();*

*}*

*}*

*2) How many OOPS principles are there and what are those and provide the sample example for each one?*

*Ans) We Have 4 OOPS Principles*

*1) Encapsulation*

*2) Abstraction*

*3) Ploymorphism*

*4) Inheritance*

*1)Encapsulation:-*

*Encapsulation is an Object it hides certain features of the object not to expose to the user..*

*It encapsulates code and data and the data acts upon the code.*

*Code is instance variables*

*Data is instance methods.*

*The example for the encapsulation is Class which has private variables and public setters and getters methods. If we want to update or read this instance variables we can do from these public methods only.*

*In this way we provide security to the instance variables by keeping private access specifier.*

*For example if you don’t want to access particular variable outside of the class we can keep getter method as private....*

*(OR)*

*The Process of Creating a class by hiding internal data from the outside world and accessing it only through public method is known as Data Encapsulation or Data Hiding.*

*We Can define encapsulation in java two ways.*

*1)By Declaring Variables as private, to restrict it from direct access.*

*2)By Defining one pair of public setter and getter methods to access private variables.*

*Example:-*

*Package p1;*

*Public class Account{*

*Private double balance;*

*Public double getBalance()*

*{*

*Return balance;*

*}*

*Public void setBalance(double balance)*

*{*

*This.balance=balance;*

*} }*

*Class BankUser*

*{*

*Public static void main(String args[])*

*{*

*Account a=new Account();*

*a.setBalance(500000);*

*System.out.println(a.getBalance());*

*}*

*}*

*OUTPUT :- 500000.*

*2) Abstraction: -*

***Abstraction****is used to hide certain details and only show the essential features of the object. In other words, it deals with the outside view of an object (interface).*

*The Best example of the abstraction is interface and Abstract class.*

*Interface provides what we need to implement but not how we have to implement..*

*The class which is implemented the interface will provide the implementation for those methods.*

*(OR)*

*Hiding Internal Implementation and just to hide the set up services what we are offering is called Abstraction. By Using interface and abstract class, we can implement abstraction.*

*Example Code Snippets:-*

***Abstract Class****...*

*abstract class Car*

*{*

*void engine()*

*{*

*System.out.println("this is different for each car");*

*}*

*}*

*class BMW extends Car*

*{*

*void tyres()*

*{*

*System.out.println("car has 4 tyres");*

*}*

*void engine()*

*{*

*System.out.println("BMW mega\_Et10.25 engine");*

*}*

*public static void main(String[] args)*

*{*

*BMW c=new BMW();*

*c.tyres();*

*c.engine();*

*}*

*}*

***Interface****:-*

*interface Petrol*

*{*

*public void petrol\_Price();*

*public void price\_India();*

*public void price\_State();*

*}*

*class Price implements Petrol*

*{*

*public void petrol\_Price()*

*{*

*System.out.println("Petrol price:75.50Rupees");*

*}*

*public void price\_India()*

*{*

*System.out.println("Petrol price:75.50Rupees");*

*}*

*public void price\_State()*

*{*

*System.out.println("Petrol price:75.50Rupees");*

*}*

*public static void main(String[] args)*

*{*

*Price p=new Price();*

*p. petrol\_Price();*

*p.price\_India();*

*p.price\_State();*

*}*

*}*

*What is differences between Encapsulation and Abstraction?*

*Ans) In simple words: You do abstraction when deciding what to implement. You do encapsulation when hiding something that you have implemented.*

*3) Polymorphism:- Defining a method in multiple classes with the same name with different implementations for exhibiting different behaviours of the object is called polymorphism.*

*(Or)*

***Polymorphism****is the ability of an object to take on many forms.*

*The process of representing one form in multiple forms is known as****Polymorphism****.*

*The word "poly" means many and "morphs" means forms. So polymorphism means many forms.*

*Example in java is class having one method with different method signatures.*

*With different parameters.*

### *Real life example of polymorphism*

*Suppose if you are in class room that time you behave like a student, when you are in market at that time you behave like a customer, when you at your home at that time you behave like a son or daughter, Here one person present in different-different behaviors.*

*We can develop polymorphism by using*

* *Method Overloading*
* *Method Overriding.*

*To develop polymorphism ,we must define method in all subclasses with the same name with the same prototype as it is declared in the super class.*

*Java supports two types of polymorphism*

*1) Compile time polymorphism*

*2) Run-time polymorphism.*

1. *Compile time polymorphism: - When a method is invoked, and its method definition which is bind at compile time by compiler is only executed by jvm at runtime, then it is called compile time polymorphism. Static,non static and overloaded methods which are not overridden in sub class come under compile time polymorphism.*
2. *Run- time polymorphism:- When a method is invoked , the method definition which is bind at compilation time is not executed at runtime , instead it is executed from the subclass based on the object stored in the reference variable is called Run time Ploymorphism. Private non static methods and default non static methods from outside package are not overridden, so these method call comes under compile time polymorphisms.*

***Example on Method Overloading (Compile time Polymorphism)***

*Public class Test*

*{*

*int a=0,b=0;*

*Public void m1(int a)*

*{*

*This.a=a;*

*System.out.println(“Value of a is”+a);*

*}*

*Public void m1(int a,int b)*

*{*

*This.a=a;*

*This.b=b;*

*C=a+b;*

*System.out.println(“The value of C is ”+ c);*

*}*

*Public static void main(String args[])*

*{*

*Test t=new Test();*

*t.m1(1);*

*t.m1(1,2);*

*}*

*}*

*Example on Method Overriding(Runtime Ploymorphism)*

*Public class Test2*

*{*

*public int a=0,b=0;*

*public void m1(int a)*

*{*

*This.a=a;*

*System.out.println(“Iam from m1 of Super class Test1 “);*

*}*

*public static void main(String args[])*

*{*

*Test2 t2=new Test2();*

*T2.m1(2);*

*}*

*}*

*Public class Test3 extends Test2*

*{*

*Public int a=0,b=0;*

*Public void m1(int a)*

*{*

*This.a=a;*

*System.out.println(“The value of a is ”+ a);*

*}*

*Public static void main(String args[])*

*{*

*Test3 t3=new Test3();*

*T3.m1(3);*

*}*

*}*

***Polymorphism Example***

*An example of polymorphism is referring the instance of subclass, with reference variable of super-class. e.g.*

*Object o = new Object(); //o can hold the reference of any subtype*

*Object o = new String();*

*Object o = new Integer();*

*Here, String is subclass of Object class. This is basic example of polymorphism.*

*In java language, polymorphism is essentially considered into two versions.*

1. *Compile time polymorphism (static binding or method overloading)*
2. *Runtime polymorphism (dynamic binding or method overriding)*

*Here, it is important to understand the these divisions are specific to java. In context to software engineering, there are other form of polymorphisms also applicable to different languages, but for java, these are mainly considered.*

## *Compile time polymorphism (static binding or method overloading)*

*As the meaning is implicit, this is used to write the program in such a way, that flow of control is decided in compile time itself. It is achieved using method overloading.*

*In method overloading, an object can have two or more methods with same name, BUT, with their method parameters different. These parameters may be different on two bases:*

***1) Parameter type****: Type of method parameters can be different. e.g. java.util.Math.max() function comes with following versions:*

*public static double Math.max(double a, double b){..}*

*public static float Math.max(float a, float b){..}*

*public static int Math.max(int a, int b){..}*

*public static long Math.max(long a, long b){..}*

*The actual method to be called is decided on compile time based on parameters passed to function in program.*

***2) Parameter count****: Functions accepting different number of parameters. e.g. in employee management application, a factory can have these methods:*

*EmployeeFactory.create(String firstName, String lastName){...}*

*EmployeeFactory.create(Integer id, String firstName, String lastName){...}*

*Both methods have same name “create” but actual method invoked will be based on parameters passed in program.*

## *Runtime polymorphism (dynamic binding or method overriding)*

***Runtime polymorphism is essentially referred as method overriding.****Method overriding is a feature which you get when you implement inheritance in your program.*

*A simple example can be from real world e.g. Animal. An application can have Animal class, and its specialized sub classes like Cat and Dog. These subclasses will override the default behavior provided by Animal class + some of its own specific behavior.*

|  |
| --- |
| *public class Animal {*  *public void makeNoise()*  *{*  *System.out.println("Some sound");*  *}*  *}*    *class Dog extends Animal{*  *public void makeNoise()*  *{*  *System.out.println("Bark");*  *}*  *}*    *class Cat extends Animal{*  *public void makeNoise()*  *{*  *System.out.println("Meawoo");*  *}*  *}* |

*Now which makeNoise() method will be called, depends on type of actual instance created on runtime e.g.*

|  |
| --- |
| *public class Demo*  *{*  *public static void main(String[] args) {*  *Animal a1 = new Cat();*  *a1.makeNoise(); //Prints Meowoo*    *Animal a2 = new Dog();*  *a2.makeNoise(); //Prints Bark*  *}*  *}* |

1. *Inheritance: -****Inheritance in java****is a mechanism in which one object acquires all the properties and behaviors of parent object. The idea behind****inheritance in java****is that you can create new classes that are built upon existing classes.*

*Inheritance Rules*

***1. A class implements an interface:***

*When a class implements an interface, it has to provide implementation details for all the methods of that interface (overriding).*

*Consider the following interface:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8* | *public interface Animal {*  *public void eat();*    *public void move();*    *public void sleep();*    *}* |

*And here is a class that implements this interface:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11*  *12*  *13*  *14* | *public class Dog implements Animal {*    *public void eat() {*  *System.out.print("Eating...");*  *}*    *public void move() {*  *System.out.print("Moving...");*  *}*    *public void sleep() {*  *System.out.print("Sleeping...");*  *}*  *}* |

*Here, you can see that the Animalinterface declares 3 abstract methods with empty body. Being a subtype, the Dog class must provide implementation details for these methods - this is called overriding*

*Besides the responsibility of overriding the methods from the super interface, the Dog class can also implement methods of its own:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9* | *public class Dog implements Animal {*  *// overrides methods from Animal interface..*      *// dog-specific behavior*  *public void bark() {*  *System.out.print("Gow Gow!");*  *}*  *}* |

***2. An abstract class implements an interface:***

*When an abstract class implements an interface, it’s not required to override methods of the super interface. But the first non-abstract class in the inheritance tree must override the methods.*

*Consider the following interface:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7* | *public interface Moveable {*  *public void start();*    *public void stop();*    *public void move();*  *}* |

*And the abstract subclass:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9* | *public abstract class Vehicle implements Moveable {*  *public void start() {*  *System.out.print("Starting...");*  *}*    *public void stop() {*  *System.out.print("Stopping...");*  *}*  *}* |

*As the Vehicle is abstract, it’s legal to override only two methods from the Moveable interface. And the move() method must be implemented by the first non-abstract in the inheritance tree.*

*Suppose that Car is a subclass of Vehicle, so we have an inheritance tree like this:*

***Moveable > Vehicle > Car***

*Since Car is a non-abstract class, it must implement the move() method of the Moveable interface:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5* | *public class Car extends Vehicle {*  *public void move() {*  *System.out.print("Moving by engine...");*  *}*  *}* |

***3. A class extends another class:***

*When a class extends another class, the subclass inherits all the public and protected members of the super class. The default members are inherited only in the same package.*

*Let’s see the following Car class:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11*  *12*  *13*  *14*  *15*  *16*  *17* | *public class Car {*  *String name;*  *private int age;*  *protected int numberOfWheels = 4;*    *protected void start() {*  *System.out.print("The car is starting...");*  *}*    *public void move() {*  *System.out.print("The car is moving...");*  *}*    *private void test() {*  *System.out.print("The car is being tested...");*  *}*  *}* |

*And the following is the Truck class:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6* | *public class Truck extends Car {*  *public Truck() {*  *numberOfWheels = 8;*  *name = "Truck";*  *}*  *}* |

*Let’s look at the constructor of the Truck class - it can access the protected field numberOfWheels and the default field name. However, it cannot access the private field age. We can write:*

|  |  |
| --- | --- |
| *1*  *2*  *3* | *Truck truck = new Truck();*  *truck.start();*  *truck.move();* |

*But the Java compiler will complain if we try to invoke the private method test():*

|  |  |
| --- | --- |
| *1* | *truck.test();   // COMPILE ERROR, since private member is not inherited* |

***4. An interface extends another interface:***

*When an interface extends another interface, the sub interface inherits all the methods declared in the super interface, and it’s free to re-declare the methods or not. For example:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7* | *public interface Moveable {*  *public void start();*    *public void stop();*    *public void move();*  *}* |

*And here is a sub interface:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5* | *public interface Controllable extends Moveable {*  *public void turn();*    *public void brake();*  *}* |

***5. A class extends another class and implements another interface:***

*In Java, we can have a class inherits both an interface and a class directly. Consider the following example:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11*  *12*  *13*  *14*  *15*  *16*  *17*  *18*  *19* | *public interface Moveable {*  *public void move();*  *}*    *public class Car {*  *protected void start() {*  *System.out.print("Starting...");*  *}*    *public void stop() {*  *System.out.print("Stopping...");*  *}*  *}*    *public class Truck extends Car implements Moveable {*  *public void move() {*  *System.out.print("Moving...");*  *}*  *}* |

*In this case, the Truck class inherits the start() and stop() methods of its super class Car, and has to override the move() method from its super interface Moveable.*

*In practice with Swing development, it’s common to have a class extended a JFrame and implements an event listener interface, for example:*

|  |  |
| --- | --- |
| *1*  *2* | *public class MyFrame extends JFrame implements ActionListener {*  *}* |

***6. Multiple inheritance of state is not allowed:***

*Remember that Java does not allow a class inherits two or more classes directly. To understand why multiple inheritance is not allowed, consider the following example:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11*  *12*  *13*  *14*  *15*  *16*  *17*  *18*  *19*  *20* | *public class A {*  *public void methodA() {*    *}*    *public void foo() {*    *}*  *}*      *public class B {*  *public void methodB() {*    *}*    *public void foo() {*    *}*  *}* |

*Suppose that we want to write a class C that extends both A and B like this:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5* | *public class C extends A, B {*  *public void methodC() {*  *foo();*  *}*  *}* |

*As you can see, both A and B has a method called foo(), so which foo() method the class C invokes exactly? from A or B? This case is ambiguous hence Java does not allow.*

***7. Multiple inheritance of type is allowed:***

*This means Java does allow multiple inheritance between interfaces. For example:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11* | *public interface X {*  *public void methodX();*  *}*    *public interface Y {*  *public void methodY();*  *}*    *public interface Z extends X, Y {*  *public void methodZ();*  *}* |

*This is allowed because interfaces do not have concrete methods, thus there is no ambiguity.*

*Likewise, we can have a class implements multiple interfaces:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7* | *public class Sub implements X, Y, Z {*  *public void methodX() { }*    *public void methodY() { }*    *public void methodZ() { }*  *}* |

***8. Type Reference and Declaration:***

*One interesting point with inheritance in Java is that, we can use the super type when creating a new object of the sub type.*

*Suppose Car is the parent class of the Truck class as shown in the following code:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9* | *public class Car {*  *public void move() {*  *}*  *}*    *public class Truck extends Car {*  *public void load() {*  *}*  *}* |

*It’s legal to write:*

|  |  |
| --- | --- |
| *1* | *Car vehicle = new Truck();* |

*The rule is: the type on the left side can be super type of the actual object in the right side. Note that the only methods we can invoke are the ones declared in the super type, for example:*

|  |  |
| --- | --- |
| *1* | *vehicle.move(); //okay, since Truck inherits Car* |

*Also note that, if the subclass declares a new method, we cannot invoke it using the super type reference:*

|  |  |
| --- | --- |
| *1* | *vehicle.load(); // compile error* |

*It’s illegal to create a new Car object like this:*

|  |  |
| --- | --- |
| *1* | *Truck car = new Car();  // Compile error* |

*Why? It’s because Truck is the subclass which can declare new methods other than the ones inherited from the Car class, whereas the actual object is of type Car. We may write this:*

|  |  |
| --- | --- |
| *1* | *car.load();* |

*It’s seems okay because the Truck class declares the load() method. But in this case, the actual object is Car which does not have the load() method.*

***9. The super keyword:***

*In Java, the****super****keyword is used to access members of the super type from the sub type. For example:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11*  *12*  *13*  *14*  *15*  *16*  *17*  *18* | *public class Super {*  *protected void bar() {*  *System.out.println("Super.bar()...");*  *}*    *}*    *public class Sub extends Super {*  *public void bar() {*  *System.out.println("Sub.bar()...");*  *}*      *public void test() {*  *super.bar();*  *this.bar();*  *}*  *}* |

*Here, we can see that both classes have a method called bar() (overriding). Therefore, if we want to invoke the super’s version of the bar() method, the super keyword is used.*

***10. Only public and protected members are inherited:***

*Private members are not inherited. Default members are inherited only if both the subclass and super class are in the same package, otherwise default members are not inherited.*

***11. Constructors are not inherited:***

*The sub class cannot inherit constructors from its super class. Consider the following example:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11* | *public class Super {*  *protected String text;*    *public Super(String text) {*  *this.text = text;*  *}*  *}*    *public class Sub extends Super {*    *}* |

*We can create a new instance of Super like this:*

|  |  |
| --- | --- |
| *1* | *Super super = new Super(“Hello”);   //OK* |

*But we cannot do the same for the Sub class:*

|  |  |
| --- | --- |
| *1* | *Sub sub = new Sub(“Hello”); //Compile error, since constructor is not inherited* |

*The solution is to write a constructor for the sub class and call the super’s constructor, for example:*

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6* | *public class Sub extends Super {*    *public Sub(String text) {*  *super(text);*  *}*  *}* |

***12. Object is the base class of all classes in Java:***

*Whenever you use a class in Java, the Java compiler automatically makes the class inherited the Objectclass. In other words, Object is the top class in any inheritance tree.*

*For example, when we write a class like this:*

|  |  |
| --- | --- |
| *1*  *2* | *public class A {*  *}* |

*Then the compiler make it extended the Object class implicitly:*

|  |  |
| --- | --- |
| *1*  *2* | *public class A extends Object {*  *}* |

*You know, the Object class provides some behaviors which are common to all objects such as equals() and hashCode().*

*(OR)*

*The Process of getting the properties from one class to another class is called Inheritance.*

*Example Snippet:-*

*Public class Test2*

*{*

*public int a=0,b=0;*

*public void m1(int a)*

*{*

*This.a=a;*

*System.out.println(“Iam from m1 of Super class Test1 “);*

*}*

*public static void main(String args[])*

*{*

*Test2 t2=new Test2();*

*T2.m1(2);*

*}*

*}*

*Public class Test3 extends Test2*

*{*

*Public int a=0,b=0;*

*Public void m1(int a)*

*{*

*This.a=a;*

*System.out.println(“The value of a is ”+ a);*

*}*

*Public static void main(String args[])*

*{*

*Test3 t3=new Test3();*

*T3.m1(3);*

*}*

*}*

*Super class:- Any class which is getting extended by another class is known as super class.*

*Sub class:- any class which is extending another class is known as subclass.*

***Core Java - Interview Questions and Answers***

*1. What is the most important feature of Java?*

*Java is a platform independent language.*

*2. What do you mean by platform independence?*

*Platform independence means that we can write and compile the java code in one platform (eg Windows) and can execute the class in any other supported platform eg (Linux,Solaris,etc).*

*3. What is a JVM?*

*JVM is Java Virtual Machine which is a run time environment for the compiled java class files.*

*4. Are JVM's platform independent?*

*JVM's are not platform independent. JVM's are platform specific run time implementation provided by the vendor.*

*5. What is the difference between a JDK and a JVM?*

*JDK is Java Development Kit which is for development purpose and it includes execution environment also. But JVM is purely a run time environment and hence you will not be able to compile your source files using a JVM.*

*6. What is a pointer and does Java support pointers?*

*Pointer is a reference handle to a memory location. Improper handling of pointers leads to memory leaks and reliability issues hence Java doesn't support the usage of pointers.*

*7. What is the base class of all classes?*

*java.lang.Object*

*8. Does Java support multiple inheritance?*

*Java doesn't support multiple inheritance.*

*9. Is Java a pure object oriented language?*

*Java uses primitive data types and hence is not a pure object oriented language.*

*10. Are arrays primitive data types?*

*In Java, Arrays are objects.*

*11. What is difference between Path and Classpath?*

*Path and Classpath are operating system level environment variales. Path is used define where the system can find the executables(.exe) files and classpath is used to specify the location .class files.*

*12. What are local variables?*

*Local varaiables are those which are declared within a block of code like methods. Local variables should be initialised before accessing them.*

*13. What are instance variables?*

*Instance variables are those which are defined at the class level. Instance variables need not be initialized before using them as they are automatically initialized to their default values.*

*14. How to define a constant variable in Java?*

*The variable should be declared as static and final. So only one copy of the variable exists for all instances of the class and the value can't be changed also.*

*static final int PI = 2.14; is an example for constant.*

*15. Should a main() method be compulsorily declared in all java classes?*

*No not required. main() method should be defined only if the source class is a java application.*

*16. What is the return type of the main() method?*

*Main() method doesn't return anything hence declared void.*

*17. Why is the main() method declared static?*

*main() method is called by the JVM even before the instantiation of the class hence it is declared as static.*

*18. What is the arguement of main() method?*

*main() method accepts an array of String object as arguement.*

*19. Can a main() method be overloaded?*

*Yes. You can have any number of main() methods with different method signature and implementation in the class.(d)*

*20. Can a main() method be declared final?*

*Yes. Any inheriting class will not be able to have it's own default main() method.*

*21. Does the order of public and static declaration matter in main() method?*

*No. It doesn't matter but void should always come before main().*

*22. Can a source file contain more than one class declaration?*

*Yes a single source file can contain any number of Class declarations but only one of the class can be declared as public.*

*23. What is a package?*

*Package is a collection of related classes and interfaces. package declaration should be first statement in a java class.*

*24. Which package is imported by default?*

*java.lang package is imported by default even without a package declaration.*

*25. Can a class declared as private be accessed outside it's package?*

*Not possible.*

*26. Can a class be declared as protected?*

*A class can't be declared as protected. only methods can be declared as protected.*

*27. What is the access scope of a protected method?*

*A protected method can be accessed by the classes within the same package or by the subclasses of the class in any package.*

*28. What is the purpose of declaring a variable as final?*

*A final variable's value can't be changed. final variables should be initialized before using them.*

*29. What is the impact of declaring a method as final?*

*A method declared as final can't be overridden. A sub-class can't have the same method signature with a different implementation.*

*30. I don't want my class to be inherited by any other class. What should i do?*

*You should declared your class as final. But you can't define your class as final, if it is an abstract class. A class declared as final can't be extended by any other class.*

*31. Can you give few examples of final classes defined in Java API?*

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

*32. How is final different from finally and finalize()?*

*final is a modifier which can be applied to a class or a method or a variable. final class can't be inherited, final method can't be overridden and final variable can't be changed.*

*finally is an exception handling code section which gets executed whether an exception is raised or not by the try block code segment.*

*finalize() is a method of Object class which will be executed by the JVM just before garbage collecting object to give a final chance for resource releasing activity.*

*33. Can a class be declared as static?*

*No a class cannot be defined as static. Only a method, a variable or a block of code can be declared as static.*

*34. When will you define a method as static?*

*When a method needs to be accessed even before the creation of the object of the class then we should declare the method as static.*

*35. What are the restriction imposed on a static method or a static block of code?*

*A static method should not refer to instance variables without creating an instance and cannot use "this" operator to refer the instance.*

*36. I want to print "Hello" even before main() is executed. How will you acheive that?*

*Print the statement inside a static block of code. Static blocks get executed when the class gets loaded into the memory and even before the creation of an object. Hence it will be executed before the main() method. And it will be executed only once.*

*37. What is the importance of static variable?*

*static variables are class level variables where all objects of the class refer to the same variable. If one object changes the value then the change gets reflected in all the objects.*

*38. Can we declare a static variable inside a method?*

*Static varaibles are class level variables and they can't be declared inside a method. If declared, the class will not compile.*

*39. What is an Abstract Class and what is it's purpose?*

*A Class which doesn't provide complete implementation is defined as an abstract class. Abstract classes enforce abstraction.*

*40. Can a abstract class be declared final?*

*Not possible. An abstract class without being inherited is of no use and hence will result in compile time error.*

*41. What is use of a abstract variable?*

*Variables can't be declared as abstract. only classes and methods can be declared as abstract.*

*42. Can you create an object of an abstract class?*

*Not possible. Abstract classes can't be instantiated.*

*43. Can a abstract class be defined without any abstract methods?*

*Yes it's possible. This is basically to avoid instance creation of the class.*

*44. Class C implements Interface I containing method m1 and m2 declarations. Class C has provided implementation for method m2. Can i create an object of Class C?*

*No not possible. Class C should provide implementation for all the methods in the Interface I. Since Class C didn't provide implementation for m1 method, it has to be declared as abstract. Abstract classes can't be instantiated.*

*45. Can a method inside a Interface be declared as final?*

*No not possible. Doing so will result in compilation error. public and abstract are the only applicable modifiers for method declaration in an interface.*

*46. Can an Interface implement another Interface?*

*Intefaces doesn't provide implementation hence a interface cannot implement another interface.*

*47. Can an Interface extend another Interface?*

*Yes an Interface can inherit another Interface, for that matter an Interface can extend more than one Interface.*

*48. Can a Class extend more than one Class?*

*Not possible. A Class can extend only one class but can implement any number of Interfaces.*

*49. Why is an Interface be able to extend more than one Interface but a Class can't extend more than one Class?*

*Basically Java doesn't allow multiple inheritance, so a Class is restricted to extend only one Class. But an Interface is a pure abstraction model and doesn't have inheritance hierarchy like classes(do remember that the base class of all classes is Object). So an Interface is allowed to extend more than one Interface.*

*50. Can an Interface be final?*

*Not possible. Doing so so will result in compilation error.*

*51. Can a class be defined inside an Interface?*

*Yes it's possible.*

*52. Can an Interface be defined inside a class?*

*Yes it's possible.*

*53. What is a Marker Interface?*

*An Interface which doesn't have any declaration inside but still enforces a mechanism.*

*54. Which object oriented Concept is achieved by using overloading and overriding?*

*Polymorphism.*

*55. Why does Java not support operator overloading?*

*Operator overloading makes the code very difficult to read and maintain. To maintain code simplicity, Java doesn't support operator overloading.*

*56. Can we define private and protected modifiers for variables in interfaces?*

*No.*

*57. What is Externalizable?*

*Externalizable is an Interface that extends Serializable Interface. And sends data into Streams in Compressed Format. It has two methods, writeExternal(ObjectOuput out) and readExternal(ObjectInput in)*

*58. What modifiers are allowed for methods in an Interface?*

*Only public and abstract modifiers are allowed for methods in interfaces.*

*59. What is a local, member and a class variable?*

*Variables declared within a method are "local" variables.*

*Variables declared within the class i.e not within any methods are "member" variables (global variables).*

*Variables declared within the class i.e not within any methods and are defined as "static" are class variables.*

*60. What is an abstract method?*

*An abstract method is a method whose implementation is deferred to a subclass.*

*61. What value does read() return when it has reached the end of a file?*

*The read() method returns -1 when it has reached the end of a file.*

*62. Can a Byte object be cast to a double value?*

*No, an object cannot be cast to a primitive value.*

*63. What is the difference between a static and a non-static inner class?*

*A non-static inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.*

*64. What is an object's lock and which object's have locks?*

*An object's lock is a mechanism that is used by multiple threads to obtain synchronized access to the object. A thread may execute a synchronized method of an object only after it has acquired the object's lock. All objects and classes have locks. A class's lock is acquired on the class's Class object.*

*65. What is the % operator?*

*It is referred to as the modulo or remainder operator. It returns the remainder of dividing the first operand by the second operand.*

*66. When can an object reference be cast to an interface reference?*

*An object reference be cast to an interface reference when the object implements the referenced interface.*

*67. Which class is extended by all other classes?*

*The Object class is extended by all other classes.*

*68. Which non-Unicode letter characters may be used as the first character of an identifier?*

*The non-Unicode letter characters $ and \_ may appear as the first character of an identifier*

*69. What restrictions are placed on method overloading?*

*Two methods may not have the same name and argument list but different return types.*

*70. What is casting?*

*There are two types of casting, casting between primitive numeric types and casting between object references. Casting between numeric types is used to convert larger values, such as double values, to smaller values, such as byte values. Casting between object references is used to refer to an object by a compatible class, interface, or array type reference.*

*71. What is the return type of a program's main() method?*

*void.*

*72. If a variable is declared as private, where may the variable be accessed?*

*A private variable may only be accessed within the class in which it is declared.*

*73. What do you understand by private, protected and public?*

*These are accessibility modifiers. Private is the most restrictive, while public is the least restrictive. There is no real difference between protected and the default type (also known as package protected) within the context of the same package, however the protected keyword allows visibility to a derived class in a different package.*

*74. What is Downcasting ?*

*Downcasting is the casting from a general to a more specific type, i.e. casting down the hierarchy*

*75. What modifiers may be used with an inner class that is a member of an outer class?*

*A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.*

*76. How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters?*

*Unicode requires 16 bits and ASCII require 7 bits Although the ASCII character set uses only 7 bits, it is usually represented as 8 bits.*

*UTF-8 represents characters using 8, 16, and 18 bit patterns.*

*UTF-16 uses 16-bit and larger bit patterns.*

*77. What restrictions are placed on the location of a package statement within a source code file?*

*A package statement must appear as the first line in a source code file (excluding blank lines and comments).*

*78. What is a native method?*

*A native method is a method that is implemented in a language other than Java.*

*79. What are order of precedence and associativity, and how are they used?*

*Order of precedence determines the order in which operators are evaluated in expressions. Associatity determines whether an expression is evaluated left-to-right or right-to-left.*

*80. Can an anonymous class be declared as implementing an interface and extending a class?*

*An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.*

*81. What is the range of the char type?*

*The range of the char type is 0 to 216 - 1 (i.e. 0 to 65535.)*

*82. What is the range of the short type?*

*The range of the short type is -(215) to 215 - 1. (i.e. -32,768 to 32,767)*

*83. Why isn't there operator overloading?*

*Because C++ has proven by example that operator overloading makes code almost impossible to maintain.*

*84. What does it mean that a method or field is "static"?*

*Static variables and methods are instantiated only once per class. In other words they are class variables, not instance variables. If you change the value of a static variable in a particular object, the value of that variable changes for all instances of that class. Static methods can be referenced with the name of the class rather than the name of a particular object of the class (though that works too). That's how library methods like System.out.println() work. out is a static field in the java.lang.System class.*

*85. Is null a keyword?*

*The null value is not a keyword.*

*86. Which characters may be used as the second character of an identifier, but not as the first character of an identifier?*

*The digits 0 through 9 may not be used as the first character of an identifier but they may be used after the first character of an identifier.*

*87. Is the ternary operator written x : y ? z or x ? y : z ?*

*It is written x ? y : z.*

*88. How is rounding performed under integer division?*

*The fractional part of the result is truncated. This is known as rounding toward zero.*

*89. If a class is declared without any access modifiers, where may the class be accessed?*

*A class that is declared without any access modifiers is said to have package access. This means that the class can only be accessed by other classes and interfaces that are defined within the same package.*

*90. Does a class inherit the constructors of its superclass?*

*A class does not inherit constructors from any of its superclasses.*

*91. Name the eight primitive Java types.*

*The eight primitive types are byte, char, short, int, long, float, double, and boolean.*

*92. What restrictions are placed on the values of each case of a switch statement?*

*During compilation, the values of each case of a switch statement must evaluate to a value that can be promoted to an int value.*

*93. What is the difference between a while statement and a do while statement?*

*A while statement checks at the beginning of a loop to see whether the next loop iteration should occur. A do while statement checks at the end of a loop to see whether the next iteration of a loop should occur. The do whilestatement will always execute the body of a loop at least once.*

*94. What modifiers can be used with a local inner class?*

*A local inner class may be final or abstract.*

*95. When does the compiler supply a default constructor for a class?*

*The compiler supplies a default constructor for a class if no other constructors are provided.*

*96. If a method is declared as protected, where may the method be accessed?*

*A protected method may only be accessed by classes or interfaces of the same package or by subclasses of the class in which it is declared.*

*97. What are the legal operands of the instanceof operator?*

*The left operand is an object reference or null value and the right operand is a class, interface, or array type.*

*98. Are true and false keywords?*

*The values true and false are not keywords.*

*99. What happens when you add a double value to a String?*

*The result is a String object.*

*100. What is the diffrence between inner class and nested class?*

*When a class is defined within a scope od another class, then it becomes inner class. If the access modifier of the inner class is static, then it becomes nested class.*

*101. Can an abstract class be final?*

*An abstract class may not be declared as final.*

*102. What is numeric promotion?*

*Numeric promotion is the conversion of a smaller numeric type to a larger numeric type, so that integer and floating-point operations may take place. In numerical promotion, byte, char, and short values are converted to int values. The int values are also converted to long values, if necessary. The long and float values are converted to double values, as required.*

*103. What is the difference between a public and a non-public class?*

*A public class may be accessed outside of its package. A non-public class may not be accessed outside of its package.*

*104. To what value is a variable of the boolean type automatically initialized?*

*The default value of the boolean type is false.*

*105. What is the difference between the prefix and postfix forms of the ++ operator?*

*The prefix form performs the increment operation and returns the value of the increment operation. The postfix form returns the current value all of the expression and then performs the increment operation on that value.*

*106. What restrictions are placed on method overriding?*

*Overridden methods must have the same name, argument list, and return type. The overriding method may not limit the access of the method it overrides. The overriding method may not throw any exceptions that may not be thrown by the overridden method.*

*107. What is a Java package and how is it used?*

*A Java package is a naming context for classes and interfaces. A package is used to create a separate name space for groups of classes and interfaces. Packages are also used to organize related classes and interfaces into a single API unit and to control accessibility to these classes and interfaces.*

*108. What modifiers may be used with a top-level class?*

*A top-level class may be public, abstract, or final.*

*109. What is the difference between an if statement and a switch statement?*

*The if statement is used to select among two alternatives. It uses a boolean expression to decide which alternative should be executed. The switch statement is used to select among multiple alternatives. It uses an int expression to determine which alternative should be executed.*

*110. What are the practical benefits, if any, of importing a specific class rather than an entire package (e.g. import java.net.\* versus import java.net.Socket)?*

*It makes no difference in the generated class files since only the classes that are actually used are referenced by the generated class file. There is another practical benefit to importing single classes, and this arises when two (or more) packages have classes with the same name. Take java.util.Timer and javax.swing.Timer, for example. If I import java.util.\* and javax.swing.\* and then try to use "Timer", I get an error while compiling (the class name is ambiguous between both packages). Let's say what you really wanted was the javax.swing.Timer class, and the only classes you plan on using in java.util are Collection and HashMap. In this case, some people will prefer to import java.util.Collection and import java.util.HashMap instead of importing java.util.\*. This will now allow them to use Timer, Collection, HashMap, and other javax.swing classes without using fully qualified class names in.*

*111. Can a method be overloaded based on different return type but same argument type ?*

*No, because the methods can be called without using their return type in which case there is ambiquity for the compiler.*

*112. What happens to a static variable that is defined within a method of a class ?*

*Can't do it. You'll get a compilation error.*

*113. How many static initializers can you have ?*

*As many as you want, but the static initializers and class variable initializers are executed in textual order and may not refer to class variables declared in the class whose declarations appear textually after the use, even though these class variables are in scope.*

*114. What is the difference between method overriding and overloading?*

*Overriding is a method with the same name and arguments as in a parent, whereas overloading is the same method name but different arguments*

*115. What is constructor chaining and how is it achieved in Java ?*

*A child object constructor always first needs to construct its parent (which in turn calls its parent constructor.). In Java it is done via an implicit call to the no-args constructor as the first statement.*

*116. What is the difference between the Boolean & operator and the && operator?*

*If an expression involving the Boolean & operator is evaluated, both operands are evaluated. Then the & operator is applied to the operand. When an expression involving the && operator is evaluated, the first operand is evaluated. If the first operand returns a value of true then the second operand is evaluated. The && operator is then applied to the first and second operands. If the first operand evaluates to false, the evaluation of the second operand is skipped.*

*117. Which Java operator is right associative?*

*The = operator is right associative.*

*118. Can a double value be cast to a byte?*

*Yes, a double value can be cast to a byte.*

*119. What is the difference between a break statement and a continue statement?*

*A break statement results in the termination of the statement to which it applies (switch, for, do, or while). A continue statement is used to end the current loop iteration and return control to the loop statement.*

*120. Can a for statement loop indefinitely?*

*Yes, a for statement can loop indefinitely. For example, consider the following: for(;;);*

*121. To what value is a variable of the String type automatically initialized?*

*The default value of an String type is null.*

*122. What is the difference between a field variable and a local variable?*

*A field variable is a variable that is declared as a member of a class. A local variable is a variable that is declared local to a method.*

*123. How are this() and super() used with constructors?*

*this() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.*

*124. What does it mean that a class or member is final?*

*A final class cannot be inherited. A final method cannot be overridden in a subclass. A final field cannot be changed after it's initialized, and it must include an initializer statement where it's declared.*

*125. What does it mean that a method or class is abstract?*

*An abstract class cannot be instantiated. Abstract methods may only be included in abstract classes. However, an abstract class is not required to have any abstract methods, though most of them do. Each subclass of an abstract class must override the abstract methods of its superclasses or it also should be declared abstract.*

*126. What is a transient variable?*

*Transient variable is a variable that may not be serialized.*

*127. How does Java handle integer overflows and underflows?*

*It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.*

*128. What is the difference between the >> and >>> operators?*

*The >> operator carries the sign bit when shifting right. The >>> zero-fills bits that have been shifted out.*

*129. Is sizeof a keyword?*

*The sizeof operator is not a keyword.*

***Development Bank of Singapore Company (Telephonic Interview Questions)***

*130. Can we store key as Object in Hash Map Collection?*

*131. Why Java people made String as final and immutable?*

*131. What are the new features added in Java 6.Java 7 and Java 8?*

*Ans) Java 8 Features:-* ***JAVA8*** *has introduced the concept called meta-space generation, hence* ***permgen*** *is no longer needed when you use jdk 1.8 versions.*

*Java 6 Features:-*

1. *New collection interfaces includes Deque, NavigableSet, NavigableMap.*
2. *2)****.***

### *java.io Enhancements:*

*New class” Console” is added and it contains methods to access a character-based console device. The readPassword()methods disable echoing thus they are suitable for retrieval of sensitive data such as passwords. The method System.console ()returns the unique console associated with the Java Virtual Machine.*

*Increased Developer Productivity*

***HCL Technologies Interview Questions***

*132. Write a program using join (Threading) method?*

*133. Explain the memory Management of Java?*

*134.when will come OutofMemory error?*

*Ans) OutOfMemoryError in Java is a subclass* ***of java.lang.VirtualMachineError*** *and JVM throws java.lang.OutOfMemoryError when it ran out of memory in the heap. OutOfMemoryError in Java can come anytime in heap mostly while you try to create an object and there is not enough space on the heap to allocate that object.*

*Types of OutOfMemoryError in Java*

*I have seen mainly two types of OutOfMemoryError in Java:  
  
1) The* ***java.lang.OutOfMemoryError: Java heap space*** *2) The* ***java.lang.OutOfMemoryError: PermGen space***

*Though both of them occur because JVM ran out of memory they are quite different to each other and their solutions are independent of each other.*

### *Solving initialization-time OutOfMemoryError*

*When the OutOfMemoryError due to PermGen exhaustion is triggered during the application launch, the solution is simple. The application just needs more room to load all the classes to the PermGen area so we just need to increase its size. To do so, alter your application launch configuration and add (or increase if present) the -XX:MaxPermSize parameter similar to the following example:*

*java -XX:MaxPermSize=512m com.yourcompany.YourClass...*

*The above configuration will tell the JVM that PermGen is allowed to grow up to 512MB before it can start complaining in the form of OutOfMemoryError.*

*PermGen Space stands for memory allocation for Permanent generation All Java immutable objects come under this category, like String which is created with literals or with String.intern() methods and for loading the classes into memory. PermGen Space speeds up our String equality searching.*

*The permgen space is the area of heap that holds all the reflective data of the virtual machine itself, such as class and method objects.*

*Perm space is used to keep informations for loaded classes and few other advanced features like String Pool(for highly optimized string equality testing), which usually get created by String.intern() methods. As your application(number of classes) will grow this space shall get filled quickly, since the garbage collection on this Space is not much effective to clean up as required, you quickly get Out of Memory : perm gen space error. After then, no application shall run on that machine effectively even after having a huge empty JVM.*

*Before starting your application you should java -XX:MaxPermSize to get rid of this error.*

*The difference between "java.lang.OutOfMemoryError: Java heap space" and "java.lang.OutOfMemoryError: PermGen space"*

*If you are familiar with different generations on the heap and* [*How garbage collection works in java*](http://javarevisited.blogspot.com/2011/04/garbage-collection-in-java.html) *and aware of new, old and permanent generation of heap space then you would have easily figured out this OutOfMemoryError in Java. Permanent generation of the heap is used to store String pool and various Metadata required by JVM related to Class, method and other java primitives.  
  
Since* ***in most of JVM default size of Perm Space is around "64MB"*** *you can easily run out of memory if you have too many classes or a huge number of Strings in your project.  
  
  
An important point to remember is that it doesn't depend on* ***–Xmx*** *value so no matter how big your total heap size you can run OutOfMemory in perm space. The good thing is you can specify* ***the size of permanent generation*** *using JVM options* ***"-XX: PermSize"*** *and* ***"-XX: MaxPermSize"*** *based on your project need.  
  
One small thing to remember is that "=" is used to separate parameter and value while specifying the* ***size of perm space in the heap*** *while "=" is not required while* [***setting maximum heap size in java***](http://javarevisited.blogspot.com/2011/05/java-heap-space-memory-size-jvm.html)*, as shown in below example.****export JVM\_ARGS="-Xmx1024m -XX:MaxPermSize=256m"***

*Another reason of "****java.lang.OutOfMemoryError: PermGen****" is memory leak through* [*Classloaders*](http://javarevisited.blogspot.com/2012/12/how-classloader-works-in-java.html) *and it’s very often surfaced in WebServer and application server like tomcat, WebSphere, glassfish or WebLogic.   
  
In Application server different classloaders are used to load different web applications so that you can deploy and undeploy one application without affecting other application on the same server, but while undeploying if container somehow keeps reference of any class loaded by application class loader then that class and all other related class will not be garbage collected and can quickly fill the PermGen space if you deploy and undeploy your application many times.   
  
"java.lang.OutOfMemoryError: PermGen” has been observed many times in tomcat in our last project, but the solution of this problem are really tricky because first you need to know which class is causing a memory leak and then you need to fix that. Another reason of OutOfMemoryError in PermGen space is if any thread started by the application doesn't exit when you undeploy your application.*

*These are just some example of infamous classloader leaks, anybody who is writing code for loading and unloading classes has to be very careful to avoid this. You can also use* ***visualgc*** *for monitoring PermGen space, this tool will show the graph of PermGen space and you can see how and when Permanent space getting increased. I suggest using this tool before reaching to any conclusion.*

*135. How to stop a thread?*

|  |  |
| --- | --- |
| *Ans) vote* | *stop() and resume() have been deprecated in Java for a decade or so. destroy() has been deprecated in Java since Java 1.5* |

*Because stopping a thread in this manner is unsafe and can leave your application and the VM in an unpredictable state"*

*136.Write code for submitting the data in your project?*

*137.What is String Constant Pool?*

*Ans) As explained in these Stackoverflow questions:* [*question 1*](http://stackoverflow.com/questions/2009228/strings-are-objects-in-java-so-why-dont-we-use-new-to-create-them) *&* [*question 2*](http://stackoverflow.com/questions/1881922/questions-about-javas-string-pool) *I understand that "String literals" are* [*interned*](http://en.wikipedia.org/wiki/String_interning) *when:*

*String s = "abc";*

*And that the JVM will create a new String object instead of using an existing one from the String Pool when:*

*String s = new String("abc");*

*When the compiler encounters a String literal, it checks the pool to see if an identical String already exists. If a match is found, the reference to the new literal is directed to the existing String, and no new String literal object is created.*

*In this case, we actually end up with a slightly different behavior because of the keyword "new." In such a case, references to String literals are still put into the constant table (the String Literal Pool), but, when you come to the keyword "new," the JVM is obliged to create a new String object at run-time, rather than using the one from the constant table.*

*So if we also put a reference in nonpool memory* ***AND*** *in pool memory when we create an object using "new" and based on the definitions above.* ***Shouldn't the JVM also return the same reference when we do this?****:*

*String one = new String("test");*

*String two = "test";*

*System.out.println(one.equals(two)); // true*

*System.out.println(one == two); // false*

*Because when declaring the String literal String three = "test"; it will already exist in the pool? and therefore should return the same reference and print true? or do the previous statements mean that they will be put in pool memory but simply skipped when the new operator is used?*

*Maybe this will aid your understanding:*

*String literal = "test";*

*String one = new String(literal);*

*String two = "test";*

*System.out.println(literal == two); //true*

*System.out.println(one == two); //false*

*In the example you posted:*

*String one = new String("test");*

*String two = "test";*

*the reference passed to the constructor String(String) has the same value as the reference two due to interning. However, the string itself (referenced by these two references) is used to construct a new object which is assigned to reference one.*

*In this example, there are exactly two Strings created with the value "test": the one maintained in the constant pool and referenced whenever you use the literal "test" in an expression, and the second one created by the "new" operator and assigned to the reference one.*

*Perhaps you're confused about this statement:*

*When the compiler encounters a String literal, it checks the pool to see if an identical String already exists.*

*Note that this might be more clearly stated as:*

*When the compiler encounters a String literal, it checks to see if an identical String already exists in the pool.*

*Strings are only put in the pool when they are interned explicitly or by the class's use of a literal. So if you have, for example, this scenario:*

*String te = "te";*

*String st = "st";*

*String test = new String(te) + new String(st);*

*then while a String will exist with the value test, said String will not exist in the pool as the literal "test" has never occurred.*

*//Creates a new object even if one exists in the pool*

*String s1 = new String("Tendulkar");*

*// makes a new object string and then the reference is available to the pool*

*String s2 = s1.intern();*

*//this object is not created but references the address present in the pool*

*String s3 = "Tendulkar";*

*System.out.print(s1==s2); // results in false*

*System.out.print(s2==s3); //very very true !!!*

***So if we also put a reference in nonpool memory AND in pool memory when we create an object using "new" and based on the definitions above. Shouldn't the JVM also return the same reference when we do this?:***

***Ans*** *: When you create a new string object by using new keyword, the address generated will be a heap address, not a string constant pooled address. And both the addresses are different.*

***Questions*** *:*

*String one = new String("test");*

*String two = "test";*

*System.out.println(one.equals(two)); // true*

*System.out.println(one == two); // false*

***do the previous statements mean that they will be put in pool memory but simply skipped when the new operator is used?***

***Answer*** *: Yes, your assumption is correct. When programmer uses new keyword, JVM will simply ignore about string constant pool, and creates a new copy in the Heap. Hence both the addresses are not same.*

*138.What is the use of yield() method?*

*139.Explain concurrency in threads.*

*140.What is synchronization in threads.*

*How to represent composite key in hibernate*

*What is default scope of bean in spring*

*What are the scopes for bean in spring*

***Pyramid Stars 1***

*public class JavaProgram*

*{*

*public static void main(String args[])*

*{*

*int i, j;*

*for(i=0; i<5; i++)*

*{*

*for(j=0; j<=i; j++)*

*{*

*System.out.print("\* ");*

*}*

*System.out.println();*

*}*

*}*

*}*

*OUTPUT:-*

***\****

***\* \****

***\* \* \****

***\* \* \* \****

***\* \* \* \* \****

***Pyramid Stars2***

***public******class*** *PyramidStars2*

*{*

***public******static******void*** *main(String args[])*

*{*

***int*** *i, j;*

***for****(i=5; i>=1; i--)*

*{*

***for****(j=1; j<=i; j++)*

*{*

*System.out.print("\* ");*

*}*

*System.out.println();*

*}*

*}*

*}*

*OUTPUT:-*

***\* \* \* \* \****

***\* \* \* \****

***\* \* \****

***\* \****

***\****

***Pyramid Stars3***

***public******class*** *PyramidStars3 {*

***public******static******void*** *main(String args[])*

*{*

***int*** *i, j,k=1;*

***for****(i=0; i<5; i++)*

*{*

***for****(j=0; j<k; j++)*

*{*

*System.out.print("\* ");*

*}*

*k=k+2;*

*System.out.println();*

*}*

*}*

*}*

*/\**

*\**

*\**

*\**

*\**

*\* \**

*\* \* \**

*\* \* \* \* \**

*\* \* \* \* \* \* \**

*\* \* \* \* \* \* \* \* \**

*\**

*\**

*\**

*\*/*

***OUTPUT:-***

***\****

***\* \* \****

***\* \* \* \* \****

***\* \* \* \* \* \* \****

***\* \* \* \* \* \* \* \* \****

***Pyramid Stars4***

***public******class*** *PyramidStars4 {*

***public******static******void*** *main(String args[])*

*{*

***int*** *i, j, k=8;*

***for****(i=0; i<5; i++)*

*{*

***for****(j=0; j<k; j++)*

*{*

*System.out.print(" ");*

*}*

*k = k - 2;*

***for****(j=0; j<=i; j++)*

*{*

*System.out.print("\* ");*

*}*

*System.out.println();*

*}*

*}*

*}*

*/\**

*\**

*\* \**

*\* \**

*\* \* \**

*\* \* \* \**

*\* \* \* \* \**

*\*/*

***OUTPUT:-***

***\****

***\* \****

***\* \* \****

***\* \* \* \****

***\* \* \* \* \****

***Pyramid Stars5***

***public******class*** *PyramidStars5 {*

***public******static******void*** *main(String args[]) {*

***int*** *i, j, k = 16, tim = 1;*

***for*** *(i = 0; i < 5; i++) {*

***for*** *(j = 0; j < k; j++) {*

*System.out.print(" ");*

*}*

*k = k - 4;*

***for*** *(j = 0; j < tim; j++) {*

*System.out.print("\* ");*

*}*

*tim = tim + 2;*

*System.out.println();*

*}*

*}*

*}*

*/\**

*\* \**

*\* \* \**

*\* \* \* \* \**

*\* \* \* \* \* \* \**

*\* \* \* \* \* \* \* \* \**

*\**

*\* \*/*

***OUTPUT:-***

***\****

***\* \* \****

***\* \* \* \* \****

***\* \* \* \* \* \* \****

***\* \* \* \* \* \* \* \* \****

***Pyramid Stars6***

***public******class*** *PyramidStars6 {*

***public******static******void*** *main(String args[])*

*{*

***int*** *i, j, num;*

***for****(i=0; i<5; i++)*

*{*

*num=1;*

***for****(j=0; j<=i; j++)*

*{*

*System.out.print(num+ " ");*

*num++;*

*}*

*System.out.println();*

*}*

*}*

*}*

***OUTPUT:-***

*1*

*1 2*

*1 2 3*

*1 2 3 4*

*1 2 3 4 5*

*141)What is inner class and give an example and use?*

*Ans)*

*Inner classes are a security mechanism in Java. We know a class cannot be associated with the access modifier* ***private****, but if we have the class as a member of other class, then the inner class can be made private. And this is also used to access the private members of a class.*

*Inner classes are of three types depending on how and where you define them. They are:*

* *Inner Class*
* *Method-local Inner Classlass*
* *Anonymous Inner Class*

## *Inner Class*

*Creating an inner class is quite simple. You just need to write a class within a class. Unlike a class, an inner class can be private and once you declare an inner class private, it cannot be accessed from an object outside the class.*

*Given below is the program to create an inner class and access it. In the given example, we make the inner class private and access the class through a method.*

*class Outerclass {*

*//private variable of the outer class*

*private int num= 175;*

*//inner class*

*public class Inner\_Demo{*

*public int getNum(){*

*System.out.println("This is the getnum method of the inner class");*

*return num;*

*}*

*}*

*}*

*public class InnerClassMain{*

*public static void main(String args[]){*

*//Instantiating the outer class*

*Outerclass outer=new Outerclass();*

*//Instantiating the inner class*

*Outerclass.Inner\_Demo inner=outer.new Inner\_Demo();*

*System.out.println(inner.getNum());*

*}*

*}*

*142)What is Object class and its methods and explain the purpose of methods?*

*Ans)1)clone 2)getClass 3)hashcode 4)equals() 5)toString() 6)wait 7)notify 8)notifyAll() 9)finalize()*

*143)tell me about exceptions?*

*144)how can we write user defined exceptions?*

*145) throw vs. throws?*

*146) what is the this keyword in java?*

*145) what s the use of super keyword?*

*146)method calling in child and super methods if method is present in both child and super class?*

*147) for loop inside for loop iteration execution?*

*148) Where will be the memory created for static variables and methods and blocks..*

*CGI Faced Interview Questions*

*149)What is WSDL file and its use?(web services)*

*150) HashTable VS HashMap?*

*HashMap and Hashtable both are used to store data in key and value form. Both are using hashing technique to store unique keys.*

*But there are many differences between HashMap and Hashtable classes that are given below.*

*HashMap Hashtable*

*1) HashMap is non synchronized. It is not-thread safe and can't be shared between many threads without proper synchronization code. Hashtable is synchronized. It is thread-safe and can be shared with many threads.*

*2) HashMap allows one null key and multiple null values. Hashtable doesn't allow any null key or value.*

*3) HashMap is a new class introduced in JDK 1.2. Hashtable is a legacy class.*

*4) HashMap is fast. Hashtable is slow.*

*5) We can make the HashMap as synchronized by calling this code*

*Map m = Collections.synchronizedMap(hashMap); Hashtable is internally synchronized and can't be unsynchronized.*

*6) HashMap is traversed by Iterator. Hashtable is traversed by Enumerator and Iterator.*

*7) Iterator in HashMap is fail-fast. Enumerator in Hashtable is not fail-fast.*

*8) HashMap inherits AbstractMap class. Hashtable inherits Dictionary class.*

*151)What are the technologies used in SOAP web services? (web services) ...*

*)MYSQL Or PL/SQL*

*4)Hibernate*

*152)OOPS concepts?*

*oop's concepts are as follows:*

*1.Object*

*2.Class*

*3.Inheritance*

*4.Polymorphism*

*5.Abstraction*

*6.Encapsulation*

*153)Checked and Unchecked Exceptions?*

*There are two types of exceptions: checked exceptions and unchecked exceptions.*

*Checked exceptions are checked at compile-time. It means if a method is throwing a checked exception then it should handle the exception using try-catch block or it should declare the exception using throws keyword, otherwise the program will give a compilation error. It is named as checked exception because these exceptions are checked at Compile time.*

*Unchecked exceptions are not checked at compile time. It means if your program is throwing an unchecked exception and even if you didn’t handle/declare that exception, the program won’t give a compilation error. Most of the times these exception occurs due to the bad data provided by user during the user-program interaction. It is up to the programmer to judge the conditions in advance, that can cause such exceptions and handle them appropriately. All Unchecked exceptions are direct sub classes of RuntimeException class.*

*154)Hashcode() VS Equals()?*

*hashCode()*

*As you know this method provides the has code of an object. Basically the default implementation of hashCode() provided by Object is derived by mapping the memory address to an integer value. If look into the source of Object class , you will find the following code for the hashCode. public native int hashCode(); It indicates that hashCode is the native implementation which provides the memory address to a certain extent. However it is possible to override the hashCode method in your implementation class.*

*equals()*

*This particular method is used to make equal comparison between two objects. There are two types of comparisons in Java. One is using “= =” operator and another is “equals()”. I hope that you know the difference between this two. More specifically the “.equals()” refers to equivalence relations. So in broad sense you say that two objects are equivalent they satisfy the “equals()” condition. If you look into the source code of Object class you will find the following code for the equals() method.*

*public boolean equals(Object obj)*

*{*

*return (this == obj);*

*}*

*155)Methods of Object class?*

*Methods of Object class*

*The Object class provides many methods. They are as follows:*

*Method Description*

*1.public final ClassgetClass() returns the Class class object of this object. The Class class can further be used to get the metadata of this class.*

*2.public int hashCode() returns the hashcode number for this object.*

*3.public boolean equals(Object obj) compares the given object to this object.*

*4.protected Object clone() throws CloneNotSupportedException creates and returns the exact copy (clone) of this object.*

*5.public String toString() returns the string representation of this object.*

*6.public final void notify() wakes up single thread, waiting on this object's monitor.*

*7.public final void notifyAll() wakes up all the threads, waiting on this object's monitor.*

*8.public final void wait(long timeout)throws InterruptedException causes the current thread to wait for the specified milliseconds, until another thread notifies (invokes notify() or notifyAll() method).*

*9.public final void wait(long timeout,int nanos)throws InterruptedException causes the current thread to wait for the specified miliseconds and nanoseconds, until another thread notifies (invokes notify() or notifyAll() method).*

*10.public final void wait()throws InterruptedException causes the current thread to wait, until another thread notifies (invokes notify() or notifyAll() method).*

*11.protected void finalize()throws Throwable is invoked by the garbage collector before object is being garbage collected.*

*156)Vector VS ArrayList?*

*Difference between ArrayList and Vector*

*ArrayList and Vector both implements List interface and maintains insertion order.*

*But there are many differences between ArrayList and Vector classes that are given below.*

*ArrayList Vector*

*1) ArrayList is not synchronized. Vector is synchronized.*

*2) ArrayList increments 50% of current array size if number of element exceeds from its capacity. Vector increments 100% means doubles the array size if total number of element exceeds than its capacity.*

*3) ArrayList is not a legacy class, it is introduced in JDK 1.2. Vector is a legacy class.*

*4) ArrayList is fast because it is non-synchronized. Vector is slow because it is synchronized i.e. in multithreading environment, it will hold the other threads in runnable or non-runnable state until current thread releases the lock of object.*

*5) ArrayList uses Iterator interface to traverse the elements. Vector uses Enumeration interface to traverse the elements. But it can use Iterator also.*

*157)What are the exceptions faced in daily work?*

*ATM is one of the exceptions faced in daily work.*

*158)Spring VS Spring web flow?*

*Spring MVC is the standard MVC solution provided by SpringSource for your web applications , so you have the concept of a view which is for example your JSP page , a controller class (annotated with @Controller ) and a model which represents your data.*

*Spring Webflow is built on top of Spring MVC and differs slightly in purpose although it inherits most of the features of Spring MVC , webflow is more suited for wizard style applications where you enter data on a form and that has to pass a series of validations or capture of number of additional data on different pages before you complete your business process.*

*159)comparable VS compartor and give the example?*

*Difference between Comparable and Comparator*

*Comparable and Comparator both are interfaces and can be used to sort collection elements.*

*But there are many differences between Comparable and Comparator interfaces that are given below.*

*Comparable Comparator*

*1) Comparable provides single sorting sequence. In other words, we can sort the collection on the basis of single element such as id or name or price etc. Comparator provides multiple sorting sequence. In other words, we can sort the collection on the basis of multiple elements such as id, name and price etc.*

*2) Comparable affects the original class i.e. actual class is modified. Comparator doesn't affect the original class i.e. actual class is not modified.*

*3) Comparable provides compareTo() method to sort elements. Comparator provides compare() method to sort elements.*

*4) Comparable is found in java.lang package. Comparator is found in java.util package.*

*5) We can sort the list elements of Comparable type by Collections.sort(List) method. We can sort the list elements of Comparator type by Collections.sort(List,Comparator) method.*

*160) HashMap implementation and how it works..*

*here are four things we should know about before going into internals of how HashMap works -*

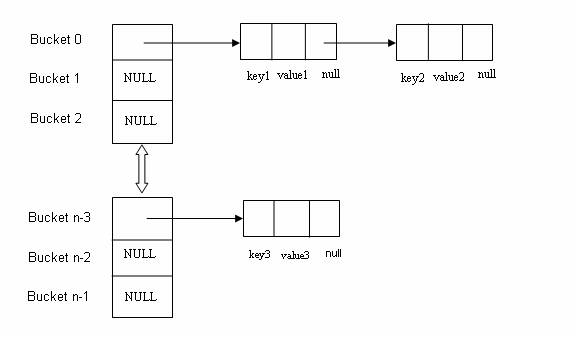
*HashMap works on the principal of hashing.*

*Map.Entry interface - This interface gives a map entry (key-value pair). HashMap in Java stores both key and value object, in bucket, as Entry object which implements this nested interface Map.Entry.*

*hashCode() -HashMap provides put(key, value) for storing and get(key) method forretrieving Values from HashMap. When put() method is used to store (Key, Value) pair, HashMap implementation calls hashcode on Key object to calculate a hash that is used to find a bucket where Entry object will be stored. When get() method is used to retrieve value, again key object is used to calculate a hash which is used then to find a bucket where that particular key is stored.*

*equals() - equals() method is used to compare objects for equality. In case of HashMap key object is used for comparison, also using equals() method Map knows how to handle hashing collision (hashing collision means more than one key having the same hash value, thus assigned to the same bucket. In that case objects are stored in a linked list.*

*Where hashCode method helps in finding the bucket where that key is stored, equals method helps in finding the right key as there may be more than one key-value pair stored in a single bucket.*

**

*161)Java 8 Features?*

*Some of the important Java 8 features are;*

*1.forEach() method in Iterable interface*

*2.default and static methods in Interfaces*

*3.Functional Interfaces and Lambda Expressions*

*4.Java Stream API for Bulk Data Operations on Collections*

*5.Java Time API*

*6.Collection API improvements*

*7.Concurrency API improvements*

*8.Java IO improvements*

*9.Miscellaneous Core API improvements*

*162)Java 7 Features?*

*The important features of JavaSE 7 are try with resource, catching multiple exceptions*

*etc.*

*1.String in switch statement (Java 7)*

*2.Binary Literals (Java 7)*

*3.The try-with-resources (Java 7)*

*4.Caching Multiple Exceptions by single catch (Java 7)*

*5.Underscores in Numeric Literals (Java 7)*

*163)Java 6 Features?*

*The important feature of JavaSE 6 is premain method (also known as instrumentation).*

*Instrumentation (premain method) (Java 6)*

*164)Java 5 Features?*

*The important features of J2SE 5 are generics and assertions. Others are auto-boxing, enum, var-args, static import, for-each loop (enhanced for loop etc.*

*1.For-each loop (Java 5)*

*2.Varargs (Java 5)*

*3.Static Import (Java 5)*

*4.Autoboxing and Unboxing (Java 5)*

*5.Enum (Java 5)*

*6.Covariant Return Type (Java 5)*

*7.Annotation (Java 5)*

*8.Generics (Java 5)*

*FEATURES JAVA 6:*

*JDK Documentation:*

*Java Platform, Standard Edition 6 is a major feature release. The following list highlights many of the significant features and enhancements in Java SE 6 since the prior major release, J2SE 5.0. It is followed by a detailed table with links to related bugs, enhancements, and JSRs.*

*Note, this web page relates to the Release Candidate milestone for Java SE 6. Its related Umbrella JSR 270 does not itself define specific features. Instead it enumerates features defined in other JSRs or through the concurrent maintenance review of the Java SE platform specification. The final release is expected to include all JSR 270 features, although*

*it is possible for an approved feature to be dropped.*

*Highlights of Technology Changes in Java SE 6:*

*1.Collections Framework*

*2.Deployment (Java Web Start and Java Plug-in)*

*3.Drag and Drop*

*4.Instrumentation*

*5.Internationalization Support*

*6.I/O Support*

*7.JAR (Java Archive Files) - An annotated list of changes*

*between the 5.0 and 6.0 releases to APIs, the jar command,*

*and the jar/zip implementation.*

*8.Java Web Start*

*9.JMX (Java Management Extensions) - A list of JMX API changes*

*between the J2SE 5.0 and Java SE 6 releases.*

*10.JPDA (Java Platform Debugger Architecture)*

*11.JVM TI (Java Virtual Machine Tool Interface)*

*12.lang and util Packages*

*13.Monitoring and Management for the Java Platform*

*14.JConsole is Officially Supported in Java SE 6*

*15.Networking Features*

*16.Performance*

*17.Reflection*

*18.RMI (Remote Method Invocation)*

*19.Scripting*

*20.Security*

*21.Serialization of Objects*

*22.Swing*

*23.VM (Java Virtual Machine)*

*Enhancements in JDK 6:*

*java.lang.instrument*

*The following functionality has been added to the java.lang.instrument package between its initial release in JDK 5.0 and JDK 6.*

*Support for retransformation of class files:*

*To facilitate dynamic transformation of classes that have already been loaded, the following methods have been added.*

*1.Instrumentation.retransformClasses(Class...)*

*2.Instrumentation.addTransformer(ClassFileTransformer, boolean)*

*3.Instrumentation.isModifiableClass(Class)*

*4.Instrumentation.isRetransformClassesSupported()*

*They are suited to operations such as adding instrumentation to methods whose classes have already been loaded. Support for retransformation means that access to the original class file is no longer required in order to instrument loaded classes. They enable the easy removal of applied*

*transformations and are designed to work in multi-agent environments.*

*Support for instrumenting native methods:*

*The following methods allow native methods to be instrumented by providing a JVM-aware mechanism for wrapping the native method.*

*1.Instrumentation.setNativeMethodPrefix(ClassFileTransformer, String)*

*2.Instrumentation.isNativeMethodPrefixSupported()*

*Support for appending to class loader search:*

*The following methods allow instrumentation support classes to be defined in the appropriate class loader.*

*1.Instrumentation.appendToBootstrapClassLoaderSearch(JarFile)*

*2.Instrumentation.appendToSystemClassLoaderSearch(JarFile)*

*JavaSE 7 Features:*

*The important features of JavaSE 7 are try with resource, catching multiple exceptions etc.*

*1.String in switch statement (Java 7)*

*2.Binary Literals (Java 7)*

*3.The try-with-resources (Java 7)*

*4.Caching Multiple Exceptions by single catch (Java 7)*

*5.Underscores in Numeric Literals (Java 7)*

*1.Strings in switch Statements:*

*In the JDK 7 release, you can use a String object in the expression of a switch statement:*

*public String getTypeOfDayWithSwitchStatement(String dayOfWeekArg) {*

*String typeOfDay;*

*switch (dayOfWeekArg) {*

*case "Monday":*

*typeOfDay = "Start of work week";*

*break;*

*case "Tuesday":*

*case "Wednesday":*

*case "Thursday":*

*typeOfDay = "Midweek";*

*break;*

*case "Friday":*

*typeOfDay = "End of work week";*

*break;*

*case "Saturday":*

*case "Sunday":*

*typeOfDay = "Weekend";*

*break;*

*default:*

*throw new IllegalArgumentException("Invalid day of the week:"*

*+ dayOfWeekArg);*

*}*

*return typeOfDay;*

*}*

*The switch statement compares the String object in its expression with the expressions associated with each case label as if it were usingthe String.equals method; consequently, the comparison of String objectsin switch statements is case sensitive. The Java compiler generates generally more efficient bytecode from switch statements that use String objects than from chained if-then-else statements.*

*2.Binary Literals:*

*In Java SE 7, the integral types (byte, short, int, and long) can also be expressed using the binary number system. To specify a binary literal, add the prefix 0b or 0B to the number. The following examplesshow binary literals:*

*// An 8-bit 'byte' value:*

*byte aByte = (byte)0b00100001;*

*// A 16-bit 'short' value:*

*short aShort = (short)0b1010000101000101;*

*// Some 32-bit 'int' values:*

*int anInt1 = 0b10100001010001011010000101000101;*

*int anInt2 = 0b101;*

*int anInt3 = 0B101; // The B can be upper or lower case.*

*// A 64-bit 'long' value. Note the "L" suffix:*

*long aLong = 0b1010000101000101101000010100010110100001010001011010000101000101L;*

*Binary literals can make relationships among data more apparent than they would be in hexadecimal or octal. For example, each successive number in the following array is rotated by one bit:*

*public static final int[] phases = {*

*0b00110001,*

*0b01100010,*

*0b11000100,*

*0b10001001,*

*0b00010011,*

*0b00100110,*

*0b01001100,*

*0b10011000*

*}*

*3.Try-with-resources:*

*In Java 7 you can write the code from the example above using the try-with-resource construct like this:*

*private static void printFileJava7() throws IOException {*

*try(FileInputStream input = new FileInputStream("file.txt")) {*

*int data = input.read();*

*while(data != -1){*

*System.out.print((char) data);*

*data = input.read();*

*}*

*}*

*}*

*Notice the first line inside the method:*

*try(FileInputStream input = new FileInputStream("file.txt")) {*

*This is the try-with-resources construct. The FileInputStream variable is declared inside the parentheses after the try keyword. Additionally, a FileInputStream is instantiated and assigned to the variable.When the try block finishes the FileInputStream will be closed automatically.*

*4.Caching Multiple Exceptions by single catch (Java 7):*

*In Java 7 it was made possible to catch multiple different exceptions in the same catch block. This is also known as multi catch.*

*Before Java 7 you would write something like this:*

*try*

*{*

*// execute code that may throw 1 of the 3 exceptions below.*

*}*

*catch(SQLException e)*

*{*

*logger.log(e);*

*}*

*catch(IOException e)*

*{*

*logger.log(e);*

*}*

*catch(Exception e)*

*{*

*logger.severe(e);*

*}*

*As you can see, the two exceptions SQLException and IOException are handled in the same way, but you still have to write two individual catch blocks for them.*

*In Java 7 you can catch multiple exceptions using the multi catch*

*syntax:*

*try*

*{*

*// execute code that may throw 1 of the 3 exceptions below.*

*}*

*catch(SQLException | IOException e)*

*{*

*logger.log(e);*

*}*

*catch(Exception e)*

*{*

*logger.severe(e);*

*}*

*Notice how the two exception class names in the first catch block are separated by the pipe character |. The pipe character between exception class names is how you declare multiple exceptions to be caught by the same catch clause.*

*5.Underscores in Numeric Literals (Java 7):*

*One of the Java 7 features is underscores in numeric literals. You can place underscores between digits of any numeric literal like int, byte, short, float, long, double. Using underscores in numeric literals will allow you to divide them in groups for better readability.*

*Underscores in Numeric Literals Tips:*

*1.Underscores can be placed only between digits.*

*2.You can’t put underscores next to decimal places, L/F*

*suffix or radix prefix. So 3.\_14, 110\_L, 0x\_123 are*

*invalid and will cause compilation error.*

*3.Multiple underscores are allowed between digits, so 12\_\_\_3*

*is a valid number.*

*4.You can’t put underscores at the end of literal. So 123\_ is*

*invalid and cause compile time error.*

*5.When you place underscore in the front of a numeric*

*literal, it’s treated as an identifier and not a numeric*

*literal. So don’t confuse with it.*

*int \_10=0;*

*int x = \_10;*

*6.You can’t use underscores where you are expecting a String*

*with digits. For example Integer.parseInt("12\_3"); will*

*throw java.lang.NumberFormatException.*

*GECE Faced Interview Questions on Sep 17 2016*

1. *Explain the JavaScript Lifecycle?*
2. *Write the JavaScript code for Two fields name and DOB entered then auto filling the value of Age Field using JavaScript code .*
3. *Write the AJAX code for request and responses.*
4. *Write the code for Creating Database connection?*
5. *Explain the Spring MVC flow?*
6. *Explain the flow of Struts and mvc flow?*
7. *Explain the sample flow of spring mvc used in your project?*
8. *Write the code for Creating HashMap?*
9. *Write the code for Creating Thread and how many types of Thread creation?*
10. *Write the code for two type of Exceptions Checked and Unchecked Exceptions?*
11. *Write the code for Taglibs used in the project?*

***Hash Table:- Hash table doesn’t allow Null as key and value ...For reference please see HashTable class Line No:- 458..***

***The Enumerations returned by Hashtable's keys and elements methods are***

***\* <em>not</em> fail-fast.***

***Fail Fast means when iterating the hashtable objects when doing any operatins like add or remove it will work safely without any issues...***

*Oct 31 2016-10-31*

*Company: Quantilus  
Title: Java Developer  
Location: Salem, OR*

*Faced Interview Questions.*

* *1) What is the size of Byte?*

*Ans) –128 to 127*

*November 01 2016*

***Cusset InfoTech***

*Client:- South Africa Power Corporation, Johanesburg..*

1. *Is this below code correct?*

*public class A*

*{*

*A a = new A();*

*}*

## *Ans:*

*Yes,correct.*

1. *Find the output?*

*class A*

*{*

*static int i = 1111;*

*static*

*{*

*i = i-- - --i;*

*}*

*{*

*i = i++ + ++i;*

*}*

*}*

*class B extends A*

*{*

*static*

*{*

*i = --i - i--;*

*}*

*{*

*i = ++i + i++;*

*}*

*}*

*public class MainClass*

*{*

*public static void main(String[] args)*

*{*

*B b = new B();*

*System.out.println(b.i);*

*}*

*}*

## *Ans: 6*

1. *Can we override?*

*class A*

*{*

*void shashi(int i)*

*{*

*//method(int)*

*}*

*}*

*class B extends A*

*{*

*@Override*

*void shashi(Integer i)*

*{*

*//method(Integer)*

*}*

*}*

## *Ans:*

*We can’t override bcz,parameter list must be same to override a method .*

1. *Whats the output?*

*public class MainClass*

*{*

*public static void main(String[] args)*

*{*

*String s = "Six"6+2+"Seven"+"Eight"+3+4+"Nine"+"Ten"+5;*

*System.out.println(s);*

*}*

*}*

## *Ans:*

*Syntax Error,we should have ‘+’ sign before number 6.*

*5)What is the output?*

*class Ad*

*{*

*int methodOfA()*

*{*

*return (true ? null : 0);*

*}*

*}*

*6)What’s the output?*

*class A*

*{*

*static void staticMethod()*

*{*

*System.out.println("Static Method");*

*}*

*}*

*public class MainClass*

*{*

*public static void main(String[] args)*

*{*

*A a = null;*

*a.staticMethod();*

*}*

*}*

## *Ans:*

*The method staticMethod cannot be declared static; static methods can only be declared in a static or top level type*

*7) Why we need to implement Single Thread model in the case of Servlet?(d)*

*I think it is deprecated now, however, originally a servlet handles multiple simultaneous requests, each in its own thread. By annotating the servlet you could funnel requests into serial processing, because the servlet only allowed one thread to process all its request.*

*But as I said, it is deprecated now, and you shouldn't use it. If you have special synchronization needs, you should handle it manually..*

*8) How to call static and dynamic page in jsp?*

*9)Can i call destroy method in init method of servlet?*

*Dstroy() gets executed and the initialization process continues.In java servlet, destroy() is not supposed to be called by the programmer. But, if it is invoked, it gets executed.*

*10)Is static block extends?*

*NO. You cannot do that . Static initialzier blocks are not inherited . Static blocks are executed when class is loaded since your base class extends a super class , even the super class definition will be loaded by JVM when referring to your class.*

*11)Can we override static method?*

*We can declare static methods with same signature in subclass, but it is not considered overriding as there won’t be any run-time polymorphism. Hence the answer is ‘No’.*

*If a derived class defines a static method with same signature as a static method in base class, the method in the derived class hides the method in the base class.*

*/\* Java program to show that if static method is redefined by*

*a derived class, then it is not overriding. \*/*

*// Superclass*

*class Base {*

*// Static method in base class which will be hidden in subclass*

*public static void display() {*

*System.out.println("Static or class method from Base");*

*}*

*// Non-static method which will be overridden in derived class*

*public void print() {*

*System.out.println("Non-static or Instance method from Base");*

*}*

*}*

*// Subclass*

*class Derived extends Base {*

*// This method hides display() in Base*

*public static void display() {*

*System.out.println("Static or class method from Derived");*

*}*

*// This method overrides print() in Base*

*public void print() {*

*System.out.println("Non-static or Instance method from Derived");*

*}*

*}*

*// Driver class*

*public class Test {*

*public static void main(String args[ ]) {*

*Base obj1 = new Derived();*

*// As per overriding rules this should call to class Derive's static*

*// overridden method. Since static method can not be overridden, it*

*// calls Base's display()*

*obj1.display();*

*// Here overriding works and Derive's print() is called*

*obj1.print();*

*}*

*}*

## *Output:*

*Static or class method from Base*

*Non-static or Instance method from Derived*

*12)Can we overload static method?*

*The answer is ‘Yes’. We can have two ore more static methods with same name, but differences in input parameters. For example, consider the following Java program.*

*// filename Test.java*

*public class Test {*

*public static void foo() {*

*System.out.println("Test.foo() called ");*

*}*

*public static void foo(int a) {*

*System.out.println("Test.foo(int) called ");*

*}*

*public static void main(String args[])*

*{*

*Test.foo();*

*Test.foo(10);*

*}*

*}*

## *Output:*

*Test.foo() called*

*Test.foo(int) called*

*13)W.A.P to convert char to string.*

*Below are two simple ways you can convert Char[] to String in Java.*

## *Java Code:*

*package com.crunchify;*

*public class CrunchifyCharArrayToString {*

*public static void main(String[] args) {*

*char[] myString = new char[] {'T', 'H', 'I', 'S', ' ', 'I', 'S', ' ', 'T', 'E', 'S', 'T'};*

*String output1 = new String(myString);*

*System.out.println("output1 : " + output1);*

*String output2 = String.valueOf(myString);*

*System.out.println("\noutput2 : " + output2);*

*}*

*}*

## *Output:*

*output1 : THIS IS TEST*

*output2 : THIS IS TEST*

*15)W.A.P to convert String to charArray.*

*The simplest way to convert String to Char Array. How to convert String to Char Array in Java?*

## *Java Code:*

*package com.crunchify;*

*public class CrunchifyStringToCharArray {*

*public static void main(String[] args) {*

*String testString = "This Is Test";*

*char[] stringToCharArray = testString.toCharArray();*

*for (char output : stringToCharArray) {*

*System.out.println(output);*

*}*

*}*

*}*

## *Output:*

*T*

*h*

*i*

*s*

*I*

*s*

*T*

*e*

*s*

*t*

***Portware Solutions, Raheja IT Park,Mind Space,Hyderabad.***

***Attended on Nov 26 2016.***

***1)Program on ThreadPooling.***

***2)JavaSwings Questions-6Questions***

***Ex:- Whats the use of MouseListenerInterface?***

***3)Program to swap the numbers***

***a=1,***

***b=2,***

***c=3,***

***d=4.***

***4)Write the output of the program***

***package*** *com.ocjp;*

***import*** *java.util.ArrayList;*

***public******class*** *ArrayListDemo {*

***public******static******void*** *main(String[] args) {*

*//* ***TODO*** *Auto-generated method stub*

*ArrayList<String> list=****new*** *ArrayList<String>();*

*list.add("A");*

*list.add("B");*

*list.add("C");*

*list.add("D");*

*list.add("E");*

***for****(****int*** *i=0; i<list.size();i++)*

*{*

*System.out.println("array elementsare :" +list.remove(i));*

*}*

*}*

*}*

***OUTPUT:- ACE.***

***5)what the use of Transient Variable and program?***

***6)InterruptedException program***

***7)What is the differences between JDK and JRE?***

***Ans) By Using JDK ,we can compile and run the program also...***

***By Using JRE, we can only run the program but we cannot develop it..***

***8)***

***List Of Java Interview Questions:***

*1.How garbage collector knows that the object is not in use and needs to be removed?*

***Answer****:*

*Garbage collector reclaims objects that are no longer being used, clears their memory, and keeps the memory available for future allocations. This is done via bookkeeping the references to the objects. Any unreferenced object is a garbage and will be collected.*

*2.Can Java thread object invoke start method twice?*

***Answer****:*

*If we invoke start method twice we get the output for the first time but we get the IllegalThreadStateException for the second time.*

*3.Give the list of Java Object class methods.*

***Answer****:*

*clone() - Creates and returns a copy of this object.*

*equals() - Indicates whether some other object is "equal to" this one.*

*finalize() - Called by the garbage collector on an object when garbage collection*

*determines that there are no more references to the object.*

*getClass() - Returns the runtime class of an object.*

*hashCode() - Returns a hash code value for the object.*

*notify() - Wakes up a single thread that is waiting on this object's monitor.*

*notifyAll() - Wakes up all threads that are waiting on this object's monitor.*

*toString() - Returns a string representation of the object.*

*wait() - Causes current thread to wait until another thread invokes the notify() method*

*or the notifyAll() method for this object.*

*4.Can we call servlet destory() from service()?*

***Ans****:*

*As you know, destory() is part of servlet life cycle methods, it is used to kill the servlet instance. Servlet Engine is used to call destory(). In case, if you call destory method from service(), it just execute the code written in the destory(), but it wont kill the servlet instance. destroy() will be called before killing the servlet instance by servlet engine.*

*5.Can we override static method?*

***Ans****: We cannot override static methods. Static methods are belogs to class, not belongs to object. Inheritance will not be applicable for class members .*

*6.Can you list serialization methods?*

***Ans****: Serialization interface does not have any methods. It is a marker interface. It just tells that your class can be serializable.*

*7.What is the difference between super() and this()?*

***Ans****:super() is used to call super class constructor, whereas this() used to call constructors in the same class, means to call parameterized constructors.*

*8.How to prevent a method from being overridden?*

***Ans****: By specifying final keyword to the method you can avoid overriding in a subcalss. Similarlly one can use final at class level to prevent creating subclasses.*

*9.Can we create abstract classes without any abstract methods?*

***Ans****: Yes, we can create abstract classes without any abstract methods.*

*10.How to destroy the session in servlets?*

***Ans****: By calling invalidate() method on session object, we can destory the session.*

*11.Can we have static methods in interface?*

***Ans****: By default, all methods in an interface are decleared as public, abstract. It will never be static. But this concept is changed with java 8. Java 8 came with new feature called "default methods" with in interfaces click here for more details.*

*12.What is transient variable?*

***Ans****: Transient variables cannot be serialized. During serialization process, transient variable states will not be serialized. State of the value will be always defaulted after deserialization.*

*13.Incase, there is a return at the end of try block, will execute finally block?*

***Ans****:Yes*

*14.What is abstract class or abstract method?*

***Ans****: We cannot create instance for an abstract class. We can able to create instance for its subclass only. By specifying abstract keyword just before class, we can make a class as abstract class.*

***public abstract class MyAbstractClass{ }***

*Abstract class may or may not contains abstract methods. Abstract method is just method signature, it does not containes any implementation. Its subclass must provide implementation for abstract methods. Abstract methods are looks like as given below:*

***public abstract int getLength();***

*15.What is default value of a boolean?*

***Ans****: zero(false).*

*16.When to use LinkedList or ArrayList?*

***Ans****: Accessing elements are faster with ArrayList, because it is index based. But accessing is difficult with LinkedList. It is slow access. This is to access any element, you need to navigate through the elements one by one. But insertion and deletion is much faster with LinkedList, because if you know the node, just change the pointers before or after nodes. Insertion and deletion is slow with ArrayList, this is because, during these operations ArrayList need to adjust the indexes according to deletion or insetion if you are performing on middle indexes. Means, an ArrayList having 10 elements, if you are inserting at index 5, then you need to shift the indexes above 5 to one more.*

*17.What is daemon thread?*

***Ans****: Daemon thread is a low priority thread. It runs intermittently in the back ground, and takes care of the garbage collection operation for the java runtime system. By calling setDaemon() method is used to create a daemon thread.*

*18.Does each thread in java uses seperate stack?*

***Ans****: In Java every thread maintains its own separate stack. It is called Runtime Stack but they share the same memory.*

*19.What is the difference between Enumeration and Iterator?*

***Ans****: The functionality of Enumeration and the Iterator are same. You can get remove() from Iterator to remove an element, while while Enumeration does not have remove() method. Using Enumeration you can only traverse and fetch the objects, where as using Iterator we can also add and remove the objects. So Iterator can be useful if you want to manipulate the list and Enumeration is for read-only access.*

*20.Find out switch statement output.*

*Code:*

*public static void main(String a[]){*

*int price = 6;*

*switch (price) {*

*case 2: System.out.println("It is: 2");*

*default: System.out.println("It is: default");*

*case 5: System.out.println("It is: 5");*

*case 9: System.out.println("It is: 9");*

*}*

*}*

***Ans****:*

*It is: default*

*It is: 5*

*It is: 9*

*21.Does system.exit() in try block executes finally block?*

***Ans****: It will not execute the finally block.*

*22.What is fail-fast in java?*

***Ans****:*

*A fail-fast system is nothing but immediately report any failure that is likely to lead to failure. When a problem occurs, a fail-fast system fails immediately. In Java, we can find this behavior with iterators. Incase, you have called iterator on a collection object, and another thread tries to modify the collection object, then concurrent modification exception will be thrown. This is called fail-fast.*

*23.What is final, finally and finalize?*

*24.In java, are true and false keywords?*

*25.What are the different session tracking methods?*

*26.What is the purpose of garbage collection?*

*27.What are the types of ResultSet?*

*What is difference between wait and sleep methods in java?*

*What is servlet context?*

*What happens if one of the members in a class does not implement Serializable interface?*

*What is race condition?*

*How to get current time in milli seconds?*

*How can you convert Map to List?*

*What is strictfp keyword?*

*Q:What is System.out in Java?*

***Ans****: Here out is an instance of PrintStream. It is a static member variable in System class. This is called standard output stream, connected to console.*

*What is difference between ServletOuptputStream and PrintWriter?*

*What is java static import?*

*When to use String and StringBuffer?*

*What is difference between StringBuffer and StringBuilder?*

*What is wrapper class in java?*

*Is Iterator a Class?*

*What is java classpath?*

*Can a class in java be private?*

*Is null a keyword in java?*

*What is the initial state of a thread when it is started?*

*What is the super class for Exception and Error?*

*What is Class.forName()?*

*Can interface be final?*

*What is the difference between exception and error?*

*What is default value of a local variables?*

*What is local class in java?*

*Can we initialise uninitialized final variable?*

*Can we declare abstract method as final?*

*Can we have finally block without catch block?*

*What is pass by value and pass by reference?*

*Can we declare main method as private?*

*What is the difference between preemptive scheduling and time slicing?*

*Can non-static member classes (Local classes) have static members?*

*What are the environment variables do we neet to set to run Java?*

*Can you serialize static fields of a class?*

*What is the difference between declaring a variable and defining a variable?*

*Where can we use serialization?*

*What modifiers are allowed for methods in an Interface?*

*What is the purpose of Runtime and System class?*

*Which one is faster? ArrayList or Vector? Why?*

*What is the difference between static synchronized and synchronized methods?*

*What is the order of catch blocks when catching more than one exception?*

*What is the difference between the prefix and postfix forms of the increment(++) operator?*

*What is hashCode?*

*What is the difference between Hashtable and HashMap?*

*What are the restrictions when overriding a method?*

*What is the use of assert keyword?*

*What is adapter class?*

*What is difference between break, continue and return statements?*

*What is the difference between while and do-while statements?*

*When does the compiler provides the default constructor?*

*Difference between C++ and Java.*

*Usages of java packages.*

*What is dynamic class loading?*

*What happens if you do not provide a constructor?*

*Difference between shallow cloning and deep cloning of objects?*

*Can we have interfaces with no defined methods in java?*

*What is the difference between “==” and equals() method?*

*How can you create an immutable class in java?*

*What are access modifiers?*

*Can we have private constructor in java?*

*Why do we need generics in java?*

*What is the difference between a product and a project?*

*How does substring() method works on a string?*

*What is the difference between a Java Library and a framework?*

*JPMorgan Chase Interview Questions on Jan 25TH 2017.*

*1)what is the differences between List and Set?*

*2)Write the program on Singleton Object?*

*3)Write the program on reversing a string ?*

*4)Write the program on reversing a string using recursion?*

*5)Explain about the Spring web flow framework?*

*6)Why Spring framework?*

*7)What is the differences between SOAP and Restful Webservices?*

*8)I have one static method executing by Thread T1 and one instance method executing by Thread T2 then what will happen?*

*9)About Cucumber how can we access data from Database in the future file and Step Definition file?*

*10)HashTable and HashMap Differences?*

*11)HashMap Implementation?*

***4Soft Interview Questions***

*written test contains two papers*

*1)java-------------------------------25 qs 30minutes*

*2)aptitude and english--------25 qs 30minutes*

*1)try*

*{*

*do something*

*}*

*finally*

*{*

*}*

*will this program complie and run?*

*2)try*

*{*

*an exception raised*

*}*

*catch(Exception e)*

*{*

*sop("hai"+ e);*

*}*

*finally*

*{*

*sop("hello");*

*}*

*ans)hai java.lang.exception*

*what is the output of the program?*

*3)try // asked in intense technology also...........*

*{*

*an exception not raised*

*}*

*catch(Exception e)*

*{*

*sop("hai"+ e);*

*}*

*finally*

*{*

*sop("hello");*

*}*

*ans)hello*

*what is the output of the program?*

*4)what will happen a java program contains more than one main(String args[])?*

*a)successfully compile and executes*

*b)compile error*

*c)runtime error*

*d)none*

*ans)b*

*5)String a="australia";*

*String b=a.subString(5,7);*

*String c= b.concat("e");*

*sop(c);*

*what is the output of the program?*

*6)a java program has main() with that signature is not there in our program what will happen*

*a)no such method error*

*b)compile time error*

*c)runtime error*

*d)none*

*ans)a*

*7)write a java program to reversing a number ex:329 to 923?*

*8)write a java program to reversing a string without using java api?*

*9)write a java program to print fibonnaci numbers?*

*10)write a java program to sort the elements in bubblesort?*

*11)write a java program to print all prime numbers*

*12)try*

*{*

*System.exit(0);*

*an exception raised*

*}*

*catch(Exception e)*

*{*

*sop("hai"+ e);*

*}*

*finally*

*{*

*sop("hello");*

*}*

*what is the output of the program?*

*13)will constructor return a value?*

*14)write a javascript program to validate a mobile number?//intense technology...........*

*15)public class FourSoft*

*{*

*public static void main(String args[])*

*{*

*int a=012;*

*System.out.println(a);*

*}*

*}*

*what is the output?*

*a)12 b)012 c)compile time error d)runtime error*

*16)what will be the array size of 5 and initialising the 5 values*

*a)int []a={12,13,14,15,16};*

*b)new array(5);*

*c)int a[];*

*d)none...*

*17)transient variable*

*a)not used for serializing*

*b)to initialize the serializable object*

*c)used in sysnchronization*

*d)none...*

*18)public class HashSetDemo // intense technology*

*{*

*public static void main(String args[])*

*{*

*HashSet hs=new HashSet();*

*hs.add("australia");*

*HashSet hs1=new HashSet();*

*hs.add(hs1);*

*String s1=new String("australia");*

*hs.add(s1);*

*String s2=new String("australia");*

*hs.add(s2);*

*System.out.println("HashSet size is given by size() method is"+ hs.size());*

*}*

*}*

*19)i have an imported package,package of my programm class declaration is there in my java program.what is the order of declaring these three in java   
s*

*program?*

***FactSet Interview Questions***

*1)diff bwn php and java?*

*2)session in php same as session in java or different...*

*3)gc() in lang package or java.util package*

*4)include tag syntax in jsp?*

*5)to open a data in new window*

*a)type="self"*

*b)type="top"*

*c)type=""*

*d)type=""*

*6)*

***MedPlus Interview Questions***

*-------------------MedPlus Written Test on 10/11/2013--------------------*

*1--What is the output of the program??*

*class Test{*

*public static void main(String args[]){*

*int that\_number=3;*

*int this\_number=0;*

*while(that\_number<10){*

*this\_number=that\_number;*

*this\_number=that\_number+that\_number/2;*

*}*

*System.out.println("Number is"+this\_number);*

*}*

*}*

*2--What is incorrect statement in following??*

*a)Class error extends throwable*

*b)Out Of Memory Exception is type Exception*

*c)If Exception occures that must be handled by try/catch block*

*d)Class Exception extends throwable*

*3--Which data type is used to store the image in database??*

*4--What are the keywords used for create,update,delete operations performed on table??*

*5--what is the output of the program if we pass values of a and b as 10,0 ??*

*class Exception{*

*public static void main(String args[]){*

*int method(int a,int b)*

*try{*

*System.out.println("1");*

*c=a/b;*

*System.out.println("2");*

*}*

*catch(ArithmaticException ae){*

*System.out.println("3");*

*}*

*catch(Exception e){*

*System.out.println("4");*

*}*

*finally{*

*System.out.println("5");*

*}*

*}*

*}*

*6--How many address lines used in 2084\*4memory chip??*

*7--What is mean by pagefault??*

*8--What it ment by throwput??*

*9--How can you remove particular record from database table??*

*10--What is the output of the program??*

*class Test{*

*public static void main(String args[]){*

*int i=1;*

*int j=i++;*

*if((i>++j)&&(i++=j)){*

*i+=j;*

*}*

*System.out.println(i);*

*}*

*}*

*11--Which statement is used to free the memory in java??*

*12--Emp table contains some details and dept table has detois with out dept.no.Print the Emp table common details and employees which doesnt have dept.no??*

*13--Which interface doen't support the duplicate values??*

*14--Which of the following are java keywords??*

*15--What is the time complexity and space complexity of Quick sort??*

*United Health Group Interview Questions*

***UNITED HEALTH GROUP*** *INTERVIEW QUESTIONS FACED BY PARASURAM ON OCTOBER 9 2013*

*---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------*

*1)WHAT IS MEANT BY DETACHED CRITERIA AND WHERE YOU USED IT?*

*2) I HAVE ONE HASHMAP COLLECTION I WANT YOU TO PUT THE KEYS AND VALUES IN ASCENDING ORDER(SORTING) AND KEEP THEM IN ANOTHER*

*HASHMAP COLLECTION(COLLECTION)?*

*3) HOW TO YOU DEFINE(WRITE) USERDEFINED EXCEPTIONS?*

*4)WHAT IS THE PURPOSE OF SET INTERFACE AND ITS CLASSES LIKE LIST,TREE?*

*5)IN USERDEFINED EXCEPTION CLASS ,SHALL I OVERRIDE METHODS AND IF OVERRIDE WHAT METHODS.IF OVERRIDE(toString()) WHY WE NEED TO*

*OVERRIDE IT?*

*6)IN YOUR HIBERNATE APPLICATIONS,YOU USED HBMMAPPING FILE OR ANNOTATIONS?*

*7)TELL ME ABOUT HASHSET COLLECTION?*

*8)IN ARRAYLIST,WHAT IS THE RETURN TYPE OF add() and put() RETURN TYPE IN HASHMAP?*

*9)IN MY USERDEFINED EXCEPTION,IAM HARDCODING THE MESSAGES LIKE AGE SHOULD NOT LESS THAN 0.HERE HARD CODING RECOMMENDED?*

*IF NOT WHAT IS THE OTHER APPROACH?*

*10)TELL ME WHAT IS THE USE OF FINALLY BLOCK?*

*WHAT WILL BE EXECUTION FLOW AT BELOW CASES?*

*System.exit(0)-->not recommended*

*System.exit(1)*

*System.exit(2)*

*System.exit(3)*

*11)WRITE HASHMAP COLLECTION USING GENERICS?*

*12)WHAT IS MEANT BY CHECKED EXCEPTION AND UNCHECKED EXCEPTION?*

*I TOLD CLASSNOTFOUNDEXCEPTION IS THE CHECKED EXCEPTION.HE SAID WRONG?*

*VCloud Pro Interview Questions*

*v cloud pro technologies*

*20 bits 20 marks 25 minutes*

*1)"4"+2+3*

*output*

*a)42 b)43 c)423 d)none*

*ans)423*

*2)5+"3"+"9"*

*a)39 b)539 c)59 d)none*

*ans)539*

*3)on file concept*

*4)i have Sample class declare in v.cloud.pro package*

*a)v.cloud.pro*

*b)package v.cloud.pro.Sample*

*c)import v.cloud.pro.Sample*

*d)package v.cloud.pro*

*ans)d....*

*5)int number=0;*

*int number2=20;*

*while(number<number2)*

*{*

*number=number+1;*

*}*

*sop(number);*

*output of number*

*a)5 b)12 c)20 d)none*

*ans)20....*

*6)A class has same method with different parameters in one object*

*called*

*a)method overriding b)method overloading*

*c)inheritance d)none*

*7)can i convert lower datatype to higher datatype i.e byte to int?*

*8)assume i have floating data 12389.56 and i want to represent that data in floating variable intrest*

*how can i represent?*

*a)12389.56f*

*b)intrest=(FLOAT)12389.56;*

*c)12389.56*

*d)none*

*ans)12389.56f*

*9)which command used to compile a java program?*

*a)java b)javac c)javap d)none*

*ans)javac*

*10)JOB keyword where will be used in JLC?*

*a)to mark start a job and initiate*

*b)end*

*c)stop*

*d)none*

*11)JOBTIME keyword something he asked*

*12)on arrays incrementing concept*

*13)javascript exception in java is availablie in*

*1)netscape.javascript exception*

*2)something*

*3)""*

*4)none*

*ans)import netscape.javascript.\*;*

***14)how we can declare the constant in pl/sql?***

*ans)Data constant number:=value;*

*credit constant c:=10;*

*15)*

*Covalense Interview Questions*

*1) Describe Yourself for 2 minutes?*

*2) what is struts?*

*3)why struts and why not other frameworks?*

*4)In Which Year struts and Spring came into market?*

*5)How to know this struts framework is suitable to current project and who will be decided?*

*6)what are the standards followed in your struts framework?*

*7)How to know code is acceptable or not before deploying into client?*

*8)How to know which developer developed which module and how can you integrate it?*

*9)Diff Btwn Java and J2EE?*

*10)what is web application and how to develop web applications?*

*11)what are the uses of web applications?*

*12)How many developers are there in your project?*

*13)Did you face any issue(Risk) in your development?*

*14)when integrating modules that are developed by different developers did you face any issue?*

*15)There are some rules and regulations when integrating different modules what are they and did you follow?*

*16)what are the modules present in your previous project?*

*17)what was the duratin of your previous project?*

*18)what is the duration of the current project?*

*19)how did you integrate retailers module and users module and what issues came when integrating them?*

*20)what were the exceptins raised in your project and how did you resolve it?*

*21)you got a problem and solved it .. did you get any other problem after solved first?*

*22)what is the size of the testing team?*

*23)did you perform manual testing and automation testing?*

*24)How developers interact with testers?*

*25)how do you get daily task?*

*26)what is agile development process and explain about it?*

*27)what is the name of your bug tracker tool?*

*28)You given a task that you dont know it .. how will you finish it?*

*29)You told your team size is 6 and less duration for your project which is e-commerce how did you manage it with less man power and less duration?*

*30)was there any bugs after production?*

*31)what is the current version of the tomcat?*

*32)when you joined in your organization?*

*33)How you are interacting with the Client?*

*34)Whom reported your project?*

*35)Did you perform smoke testing and whom performs?*

*36) What are your roles and responsibilities in your project?*

*37)who is your client?*

*38)what is Client Location?*

*39)What is CVS and SVN?*

*40)Diff Between CVS and SVN?*

*41)You Finished your module you need to deploy to testing team..How did you deploy that code to testing team?*

*42)if client ask you to release the product by tomorrow But Not Possible how do you manage it?*

*43)Have you Involved in project budget and whom involved?*

*44)If I have given 3 modules to you..How much time do you take and how do you know it?*

*45)How do you know this project is suitable for this technology (or) for this concept?*

*46)How do you interact with the testing team?*

*47)Tester Reported a bug but you Rejected it and again he reported ? how do you handle the bug?*

*48)What is the status of the bug at the time of implementing the bug to the Testers?*

*49)Have you prepared unit test cases and what was the plan for that?*

*50)what do you maintain in loggers?*

*51)what are the standards followed in your sdlc?*

*52)Who will design use cases?*

*53)Diff Btwn Usecases and functionalities?*

*54)what is usecase and what is the use of usecase?*

*55)what is the language used to design an application?*

*56)How Many Employers working in your organization?*

*57)How Many Projects running in your Organization?*

*58)what is the environment you are using in your Project?(d)*

*59)Diff Btwn Client/Server Application and Web Based Application?*

*60)What is OOPS?*

*61)What is a Class?*

*62)What is a Object?*

*63)What is an Abstraction?*

*64)What are your Daily Activities in your Organization?*

*65)Why You Would Like to Change your Company?*

*66)If i gave you a offer how long do you work for your organization?*

*67)What is Mutual Understanding?*

*68)Any Drawbacks in your current Project?*

*69)What is Current CTC?*

*70)what is Expected CTC?*

*71)If I Give only 15% hike will you Join?*

*72)How Did your Experience with your Company?*

*73)Did you face any Situation that you speak to Project Manager and what was the reason to meet him?*

*74)Who was your Team Leader ?*

*75)what do you know about development?*

*76)When Your Current project started?*

*77)Tell me About Your Organization?*

***ABM Technologies, Mumbai 2nd Telephonic Round***

*1)i have country and state drop down list if I select country how corresponding states will be loading in our spring controller and jsp and it will display tell me the code flow?*

*2)Diff Hibernate Session and Normal Jsp Session?*

*3)Vector Vs ArrayList?*

*4)Why We use First Level Cache and Second Level Cache?*

*5)What is Diff StringBuilder and StringBuffer?*

*6)What is Diff btwn Final, Finally and Finalize()?*

*7)How we Sort the Collection Objects?*

*8)What is Spring AutoWiring?*

*9)What is Spring Transaction?*

*10)What is Diff between saveorUpdate() and merge()?*

*11)How we get Connection Objects in your Spring and Hibernate Apps?*

*12)How good in JavaScript and JQuery?*

*13)i have div tag and have its id=”mydiv” and I want to print some data how can we print some data in that div and what we have to do?*

*14)What is HttpListener?*

*15)What is Criteria?*

***Accenture Interview Questions***

*1)Could you please explain brief on java experience and first project you worked on?*

*2)Primary work you have done on client tier of the application (or) web tier of the application?*

*3)Are you comfortable with hibernate?*

*4)On Core Java which areas you worked most?*

*5)Tell me specific area which you are more comfortable in Core java?*

*6)In Collections which Collection Object you used?*

*7)Why did you use vector Object and In what kind of Scenarios you used ?*

*8)Why Vector and why not ArrayList?*

*9)How can we Provide Synchronization using vector?*

*10)In Vector Do you Initialize the capacity of the vector?*

*11)How can you Convert Vector into ArrayList?*

*12) Can I Construct an ArrayList by Passing vector as an argument?*

*13) How can you Convert ArrayList into Normal Array?*

*14)we have Collection of objects we have one scenario where we need to select any one of the collection in this time what factors will you determine to select particular collection?*

*15) What is the front controller in struts framework?*

*16)How is the mapping between Action Class and ActionServlet Done?*

*17)How ActionServlet Knows that required information available in Strust-Config.xml?*

*18)Have you worked on Struts Tiles?*

*19)Can You Explain about Struts tag Lib?*

*20)How Can You Display POJO Class data in JSP and What Code You can Use to Achieve This Requirement?*

*21) How Java calls Asynchronous Calling?*

*22)where did you JQuery In Your Project?*

*23)Tell me About IOC in Spring in Brief?*

*24)Why are you Typecasting Your Own Class Triangle into context object?*

*25) After Perform Unit Testing of your developed code then How will you Report that Unit Testing?*

***Infosys Interview Questions***

*1)Tell me about yourself?*

*2)Can We Develop Struts Application without Using Action Form?*

*3)Why you use Ajax and where you used Ajax n tell me one situation?*

*4) Why you use JQuery and where you used JQuery n tell me one situation?*

*5)How Can we call Ajax from JavaScript?*

*6)Tell me About Spring n Hibernate?*

*7)Tell me About DeadLock Situtation in java?*

*8)Can You Tell me about Design Patterns n did you design patterns in coding?*

*9)In Current how do you perform database connections?*

*10)did you use singleton design pattern in connection object?*

*11)When u r using database connection do u use two phase commit or single phase commit?*

*12)What type of collections are in java?*

*13)Did u face DeadLock Situtation in java?*

*14)why would we handle deadlock Situtation in java n what happen if we don’t handle ?*

*15)what was the situation that got deadlock in your project?*

*16)what is the typical function that you write in JQuery?*

*17)Do you know evil twelve issue in CVS Server?*

***Jala Technologies***

*1)what is the use of CallableStatement?*

*ans)In CallableStatement we use procedures. By using this procedures we can insert the data into two tables at a time....*

*ex:table 1:*

*create table personal\_details (fname varchar2,lname varchar2,addr varchar2);*

*table 2:*

*create table job (qualification varchar2,design varchar2);*

*Procedure Creation*

*create or replace procedure proc3(fname in varchar2,lname in varchar2,addr in varchar2,qualification in varchar2, design in varchar2)is*

*begin*

*insert into personal\_details values(fname,lname,addr);*

*insert into job values(qualification,design);*

*end;*

*2)which CollectionFramework does not support key and null values as null?*

*ans: Hashtable CollectionFramework.*

*3)how we can represent the data in Map interface?*

*ans: we can represent the data in Map in the form of key and value pairs.*

*HashMap hm=new HashMap();*

*hm.put(101,"A");*

*hm.put(102,"B");*

*hm.put(103,"C");*

*hm.put(104,"D");*

*System.out.println(hm);*

*4)Is HashMap order preserved?*

*ans:order is not preserved.*

*5)what is the keySet() method on HashMap(class)?*

*ans)Used to print the keys.*

*Question 1) What is difference between Vector and ArrayList in Java?*

*Answer : One of the most popular Java question at 2 years experience level which aims to check your knowledge on Java collection API. key point to mention is synchronization and speed, since ArrayList is not synchronized its fast compare to Vector. See Vector vs ArrayList in Java for more difference between both of them.*

*Question 2) What is difference in LinkedList and ArrayList in Java?*

*Answer : If you don't get previous Java question then you are likely to get this question at 2 to 3 years experience level Java interview. Unlike synchronization, key point to mention here is underlying data structure. See LinkedList vs ArrayList in Java for detail answer of this java question.*

*3) What is difference between fail-fast and fail-safe Iterator in Java?*

*This is relatively new Java question compare to previous ones but increasingly getting popular into 2 to 3 years level Java interviews. key difference here is that fail-fast Iterator throw ConcurrentModificationException while fail-safe doesn't. See fail-safe vs fail-fast Iterator in Java for more differences between two.*

*4) Difference between throw and throws in Java?*

*This Java question belongs to Exception handling category which is another popular category for 2 to 4 years experienced Java programmer. Main difference between these two is that one declares exception thrown by a Java method while other is actually used to throw Exception. See Difference between throw and throws in Java for full answer of this Java exception handling question.*

*5) What is difference between checked and unchecked Exception in Java.*

*Another java interview question for 2 to 4 years experienced Java programmer from Exception handling. key point to mention here is that checked Exception requires mandatory exception handling code while unchecked doesn't. See checked vs unchecked Exception in Java for more differences.*

*6) Write code to print Fibonacci series in Java?*

*You are bound to expect some coding interview question in Java interview for 2 to 4 years experience. Though Fibonacci series is one of the most classical and popular question not every Java programmer is able to do it correctly in interview, things get more complicated if interviewer ask to do this by using recursion. So better to prepare for approach, See how to write Java program for Fibonacci series using recursion for details and code example.*

*7) Write Java program to reverse String in Java without using StringBuffer?*

*Another Java coding interview question asked on 2 ot 4 years experienced Java developer. Many times interviewer specifically mentioned that you can not use StringBuffer because it has reverse() method which makes this taks trivial. So you must know how to do this by using iteration as well as recursion to perform well. Look at Java program to reverse String in Java for full code sample and explanation.*

*8) What is difference between Runnable and Thread in Java ?*

*Frequently asked Java interview question on 2 to 4 years experienced level from Threading fundamentals. there are two ways to implement Thread in Java, in fact three e..g extending java.lang.Thread class or implementing java.lang.Runnable or Callable interface. See Thread vs Runnable in Java for exact differences on following each approach.*

*9) What happens if you don't call wait and notify from synchronized block?*

*Another common Java interview question from multi-threding and inter thread communication. As I said earlier you must know concept of wait and notify in Java if you have worked for 2 to 3 years in Java. Check why wait and notify needs to call from synchronized block for exact reason to answer this Java question.*

*10) Write code to solve producer consumer problem in Java.*

*A good java question in my opinion which is mix of threading, synchronization, inter-thread communication and coding abilities. This particular java question can make any body's list at any day, not just 2 to 4 years experienced programmer. despite being so common it has something on it which confuse average programmer and allow you to differentiate between good and average programmer. See Producer consumer problem solving using BlockingQueue in Java for full code example.*

*These were 10 frequently asked Core Java interview question for 2 to 3 years experienced Java programmers. If you want to increase numbers there are lot many questions floating around web for 2 to 3 or 2 to 4 years experienced Java developers but key things it to remember topics e.g. Collection, Exception handling, Coding, Threading and OOPS principles are the main areas from which most of 2 to 3 years experienced Java programmer interview question appear. Just get them right and you can crack any Java interview upto 2 to 4 years experienced Java programmer.*

***Menlo Technologies ,Aditya Trade Centre,Ameerpet***

*1)Can we have private variable and methods in interface?. If we put what will happen compile time error or runtime error?*

*2)What is the difference between interface and abstract class?*

*3)Asked a program to print list objects in descending order without using Api with logic?*

*4)Asked a program to print duplicate of list objects from list with string generic and its count?*

*5)Tell me how did you write a code for JQuery for date pickers and disable copy and paste in website?*

*6)Tell me the implicit objects of jsp?*

*7)How can you create session ?*

*8)How can you add attribute to the session?*

*9)How can you delete the session write the logic?*

*10)what is the differences between session. close() and session. invalidate()?(d)*

*11)Tell me about collections in java?*

*12)Tell me about current project?*

*13)Tell me about yourself?*

*14)What is session and what is its use?*

***Minle Technologies***

*1)Diff List and Set?*

*2)How can you convert list to set?*

*3)How can you redirect from one servlet to another servlet?*

*4)tell me the workflow of struts?*

*5) i got exception in struts i didn't handle it i want to display in jsp how?*

*6)Map features?*

*7)How do you rate yourself in core java out of 5?  
8)How do you rate yourself in collection out of 5?*

*Opentext Interview Questions*

*1st round:*

*Written Test : for Profile screening*

1. *Write a program to print duplicate numbers in an array of n numbers?*
2. *Write a program to swap two numbers without using temp variable?*

*2nd Round*

1. *Tell me about yourself?*
2. *Can you briefly tell me about your Projects?*
3. *How do you rate yourself in Collection out of 10? I said 9*
4. *How do you rate yourself in Threading out of 10? I said 7*
5. *What is super constructor call?*
6. *What is the purpose of static keyword and static import?*
7. *What are the OOP principles?*
8. *What is the Thread lifecycle?*
9. *What is your C C.T.C and E C.T.C ? (d)*

*3rd Round*

*-------------------------------------------------------*

1. *Tell me about yourself?*
2. *What are OOP concepts? Explain them briefly?*
3. *Why do we need polymorphism? Explain with an Example?*
4. *When should we go for Abstract classes and Interfaces?*
5. *Can you briefly explain what Struts is?*
6. *Can we implement a web application without using Struts?*
7. *What are the limitations of Struts?*
8. *How many Design patterns do you know? Explain any of it with usage?*
9. *What is Single ton Write a program for it?(d)*
10. *Write a program to print Triangle using \* (asterisk)?*

*4th Round:*

*----------------------------------------------*

1. *Tell me about yourself?*
2. *Explain me about old project?*
3. *Some more deeper discussion happened about my project, like technologies used, architecture or flow of your project?*
4. *What is the difference between overloading and overriding?*
5. *He has given two sample codes on overloading and overriding , and asked me to predict the output?*
6. *Does static method can be overridden?*
7. *What is the purpose of finally block?*
8. *What happens when I write try with 2 catch blocks where 2nd catch block is having NFE type and its 1st catch block having RuntimeException?*
9. *Can I have throw and throws in a same method?(d)*
10. *What are the differences between List and ArrayList?*
11. *What are the differences between Hashtable and HashMap?*
12. *What are the differences between Comparable and Comparator?*
13. *How can I make a collection as synchronized?*
14. *What is the use of LinkedHashMap?*
15. *What is purpose of start() method?*
16. *What is the purpose of run() method?*
17. *Can I invoke run() method before calling start()?*
18. *I have 3 threads, How Can I join one thread after the other?*
19. *What is the purpose of synchronization?*
20. *Can I apply synchronization outside method and within the class?*
21. *What is the purpose Serialization?*
22. *Can I call remove() method before calling next() method of Iterator while retrieving the elements?*
23. *Write a program to print Fibonacci numbers upto the given number?*
24. *Find the 2nd Largest number in the Array of elements?*
25. *What is the life cycle phases of Servlet?*
26. *What are the lifecycle methods of servlet?*
27. *Can we override init() method of a Servlet?*
28. *What are difference between sendRedirect() and forward() method ? Give an Example?*
29. *What are the differences between RequestDispatcher and ServletContext?*
30. *What is PreparedStatement?*
31. *How to submit a Procedure from JDBC application?*
32. *What is connection polling?*
33. *What does happen when no.of connections exceeded ?ex fixed = 100 and when 101 connection came?*
34. *What is JSP? What is the life cycle of JSP?*
35. *What the difference is between include attribute and include action?*
36. *How do you handle exception in JSP?*
37. *How can you upload a file into a server?*
38. *What is the struts architecture?*
39. *Can you explain why request and response objects are provided as arguments to the method execute() of an Action class?*
40. *What is the Dispatcher action in Struts?*
41. *How do you validate a form in Struts?*
42. *Write a query which displays all the employees whose department is “Information Sciences”?*
43. *What is the difference between Primary Key and Unique Key?*
44. *What is Index?*
45. *What is Trigger?*
46. *Write a Query which displays 2nd highest salary from the emp table?*

*5th Round*

1. *Tell me about yourself?*
2. *Tell me about your role in old project?*
3. *What is Struts and architecture of Struts?*
4. *Can you explain your architecture of Old project?*
5. *How do you handle validation using Struts?*
6. *How did you perform client side validations using Java script?*
7. *What is RegExp in Java script?*
8. *Explain about SA Desk Project?*
9. *What is your role is this project?*
10. *He asked me some more relevant questions on that project ?and also asked me write the code of it?*
11. *Any questions from your side? I asked him to tell about the current running project in OpenText and Why Open Text is hiring candidates?*
12. *He answered about his current projects and upcoming projects?*

*6th Round*

1. *Tell me about yourself?*
2. *Tell me briefly about old and present projects?*
3. *What is the Team size of your old project? How do you communicate others?*
4. *Are they in your work location or in any other branch?*
5. *How do you get the requirement specification from your client?*
6. *How do you communicate your manager?*
7. *In your Team how does work share among the team of members?*
8. *To whom you report?*
9. *What is web server maintenance?*
10. *What are the frequent changes you do in web server?*
11. *Have you used Logging in your project?*
12. *Do you ve any questions? I asked most likely question which I asked in the previous round?*

*Randstad Interview Questions*

*1) Write a Java Logic to find Leap Year?*

*2) Write HTML code to Print TEST as Header and A, B and C as text definitions in table row.*

*3)Write a Logic To Print Date in Jsp Script let?*

*4)Write HTML code to Validate User Name Using JavaScript Validation and store it in HttpSession and Print In Jsp by Retrieving from Session?*

*5) a)Write a Single SQL Query to Print first-name,last-name of all Persons in Person Table?*

*b)Write a Single SQL Query to Print first-name ,last-name,total orders where person whom average age is more and orders weight  Exceeds 100?*

*Note:- Person id is a primary key in person table and it is foreign key in order table.*

*Person table has 1) First Name 2) Last Name 3) Person Id 4) Weight*

*Order table has  1)Order id 2)Person Id 3)Orders*

*Note:- Query would be like this.No Wonder if it has some changes.I don't remember completely.*

*Yash Technologies*

*1)Tell me about yourself briefly?*

*2)how do you rate yourself in J2EE,Core Java,Struts,Spring and Hibernate?*

*3)Could you Please explain me about OOPS Concept?*

*4)What is the diff between overloading and overriding anything else for the difference?*

*Its not enough I want the basic difference of overloading and overriding?*

*What we can do in overloading and what we can do in overriding?*

*5)Can we have constructors in abstract class n can we define constructors in abstract class?*

*6)How do you override equals() method n what are the reasons to override equals() method?*

*Anything else we have to do when overriding equals() method?*

*7)Have u worked with Collection then what is SortedMap?(d)*

*8)How SortedMap will do Ascending Order n Descending Order How it will works n is there anything to do by us?*

*9)What is Load Factor in Hash Map and How it Works?*

*Ans:* An instance of **HashMap** has two parameters that affect its performance: initial capacity and **load factor**. The capacity is the number of buckets in the **hash table**, and the initial capacity is simply the capacity at the time the **hash table** is created.

*10)What is Diff Between Error and Exception?*

*11)Have See anywhere out of memory error n how did u handle them?*

*12)Can u Expand NullPointerException?*

*13)What are the Other Exceptions you faced generally n what are they when did u get that exceptions n how did u handle?*

*14) Can we clone () an Object?*

*15) Can we define Marker Interface and what are the uses of marker interfaces?*

*16)What is the diff Synchronized method and Synchronized block?*

*17) Why we need Many-Many implementation in hibernate?*

*18) What is LazyLoading in Hibernate?*

*19)What is IOC in Spring?*

*20)What is DirtyReads in Hibernate?*

***Hibernate Interview Questions***

*HOME\_JAVA*

*This Blog provides java faq's and java openings*

*Java OpeningsCoreJava RealtimeCoding SQL FaqsJDBC Servlets JSPStrutsHibernate Spring*

*Hibernate*

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

*There are so many*

*1) Hibernate is data base independent, your code will work for all ORACLE,MySQL ,SQLServer etc.*

*In case of JDBC query must be data base specific. So hibernate based persistance logic is database independent persistance logic and JDBC based persistance logic is database dependent logic.*

*2) As Hibernate is set of Objects ,*

*3) No need to learn SQL language.You can treat TABLE as a Object . Only Java knowledge is need.*

*In case of JDBC you need to learn SQL.*

*3) Dont need Query tuning in case of Hibernate. If you use Criteria Quires in Hibernate then hibernate automatically tuned your query and return best result with performance.*

*In case of JDBC you need to tune your queries.*

*4) You will get benefit of Cache. Hibernate support two level of cache. First level and 2nd level. So you can store your data into Cache for better performance.*

*In case of JDBC you need to implement your java cache .*

*5) Hibernate supports Query cache and It will provide the statistics about your query and database status.*

*JDBC Not provides any statistics.*

*6) Development fast in case of Hibernate because you dont need to write queries*

*7) No need to create any connection pool in case of Hibernate. You can use c3p0.*

*In case of JDBC you need to write your own connection pool*

*8) In the xml file you can see all the relations between tables in case of Hibernate. Easy readability.*

*9) You can load your objects on start up using lazy=false in case of Hibernate.*

*JDBC Dont have such support.*

*10 ) Hibernate Supports automatic versioning of rows but JDBC Not.*

*2.What is Hibernate?*

*Hibernate is an open source, light weight Object Relational Mapping tool to develop the database independent persistence login in java and j2ee based applications.*

*Hibernate is a pure Java object-relational mapping (ORM) and persistence framework that allows you to map plain old Java objects to relational database tables using (XML) configuration and mapping files. Its purpose is to relieve the developer from a significant amount of relational data persistence-related programming tasks*

*3.What is ORM ?*

*ORM stands for object/relational mapping, means providing the mapping between class with table and member variables with columns is called ORM. ORM is the automated persistence of objects in a Java application to the tables in a relational database.*

*4.hat does ORM consists of ?*

*An ORM solution consists of the following four pieces:*

*API for performing basic CRUD operations*

*API to express queries referring to classes*

*Facilities to specify metadata*

*Optimization facilities : dirty checking,lazy associations fetching*

*5.What are the ORM levels ?*

*The ORM levels are:*

*Pure relational (stored procedure.)*

*Light objects mapping (JDBC)*

*Medium object mapping*

*Full object Mapping (composition,inheritance, polymorphism, persistence by reachability)*

*.*

*6.Why do you need ORM tools like hibernate?*

*The main advantage of ORM like hibernate is that it can develop the database independent persistence logic. Apart from this, ORM provides following benefits:*

*Improved productivity*

*High-level object-oriented API*

*Less Java code to write*

*No SQL to write*

*Improved performance*

*Sophisticated caching*

*Lazy loading*

*Eager loading*

*Improved maintainability*

*A lot less code to write*

*Improved portability*

*ORM framework generates database-specific SQL for you*

*7.What Does Hibernate Simplify?*

*Hibernate simplifies:*

*Saving and retrieving your domain objects*

*Making database column and table name changes*

*Centralizing pre save and post retrieve logic*

*Complex joins for retrieving related items*

*Schema creation from object model*

*8.What is the main difference between Entity Beans and Hibernate ?*

*1)In Entity Bean at a time we can interact with only one data Base. Where as in Hibernate we can able to establishes the connections to more than One Data Base. Only thing we need to write one more configuration file.*

*2) EJB need container like Weblogic, WebSphare but hibernate don't nned. It can be run on tomcat.*

*3) Entity Beans does not support OOPS concepts where as Hibernate does.*

*4) Hibernate supports multi level cacheing, where as Entity Beans doesn't.*

*5) In Hibernate C3P0 can be used as a connection pool.*

*6) Hibernate is container independent. EJB not.*

*9.What are the Core interfaces and classes of Hibernate framework?*

*The five core interfaces are used in just about every Hibernate application. Using these interfaces, you can store and retrieve persistent objects and control transactions.*

*Configuration class (org.hibernate.cfg package)*

*Session interface (org.hibernate package)*

*SessionFactory interface (org.hibernate package)*

*Transaction interface (org.hibernate package)*

*Query and Criteria interfaces (org.hibernate package)*

*10.What is the general flow of Hibernate communication with RDBMS?*

*The general flow of Hibernate communication with RDBMS is :*

*Load the Hibernate configuration file and create configuration object. It will automatically load all hbm mapping files because mapping file can be configured in configuration file.*

*Create session factory from configuration object*

*Get one session from this session factory*

*Create HQL Query*

*Execute query to get list containing Java objects.*

*11.What is the need for Hibernate mapping file?*

*Hibernate mapping file is used to provides the mapping between java class with table member variables with column names of the table. And also we can configure primary key generation algorithm, relations and so on. Typical mapping file look as follows:*

*12.What are the important tags of hibernate.cfg.xml?*

*This file can be used to provide the database information like driverclass name, url, database usename, database password, dialect, connection pooling mapping file and so on.*

*Following are the important tags of hibernate.cfg.xml:*

*13.What role does the Session interface play in Hibernate?*

*The main runtime interface between a Java application and Hibernate The Session interface is the primary interface used by Hibernate applications. It is a single-threaded, short-lived object representing a conversation between the application and the persistent store. It allows you to create query objects to retrieve persistent objects.*

*The main function of the Session is to offer create, read and delete operations for instances of mapped entity classes. Instances may exist in one of three states:*

*transient: never persistent, not associated with any Session*

*persistent: associated with a unique Session*

*detached: previously persistent, not associated with any Session*

*Session session = sessionFactory.openSession();*

*Session interface role:*

*Wraps a JDBC connection*

*Factory for Transaction*

*Holds a mandatory (first-level) cache of persistent objects, used when navigating the object graph or looking up objects by identifier*

*14.What role does the SessionFactory interface play in Hibernate?*

*SessionFactorys are immutable. The behaviour of a SessionFactory is controlled by properties supplied at configuration time. These properties are defined on Environment.*

*The application obtains Session instances from a SessionFactory. There is typically a single SessionFactory for the whole application—created during application initialization. The SessionFactory caches generate SQL statements and other mapping metadata that Hibernate uses at runtime. It also holds cached data that has been read in one unit of work and may be reused in a future unit of work*

*Implementors must be threadsafe.*

*SessionFactory sessionFactory = configuration.buildSessionFactory();*

*15.What are the most common ways to specify the Hibernate configuration properties?*

*The most common methods of Hibernate configuration are:*

*Programmatic configuration*

*By using setProperty(-) method of org.hibernate.cfg.Configuration.*

*XML configuration (hibernate.cfg.xml)*

*By using .properties file*

*By Using annotaions.(from Hibernate 3.3 on words)*

*16.How do you map Java Objects with Database tables?*

*First we need to write Java domain objects (beans with setter and getter).*

*Write hbm.xml, where we map java class to table and database columns to Java class variables.*

*Example :*

*not-null="true" type="java.lang.String"/>*

*not-null="true" type="java.lang.String"/>*

*17.How do you define sequence generated primary key algorithm in hibernate?*

*By using , tags we can configure the primary key and primary key generation algorithm.*

*Example:-*

*SEQ\_NAME*

*18.What is component mapping in Hibernate?*

*A component is an object saved as a value, not as a reference*

*A component can be saved directly without needing to declare interfaces or identifier properties*

*Required to define an empty constructor*

*Shared references not supported*

*19 . Difference between getCurrentSession() and openSession() in Hibernate ?*

*getCurrentSession() :*

*Obtains the current session. The "current session" refers to a Hibernate Session bound by Hibernate behind the scenes, to the transaction scope.*

*A Session is opened when getCurrentSession() is called for the first time and closed when the transaction ends. It is also flushed automatically before the transaction commits. You can call getCurrentSession() as often and anywhere you want as long as the transaction runs. Only the Session that you obtained with sf.getCurrentSession() is flushed and closed automatically.*

*openSession() :*

*If you decide to use manage the Session yourself the go for sf.openSession() , you have to flush() and close() it.*

*It does not flush and close() automatically.*

*Example :*

*Transaction tx =session.beginTransaction();*

*Session session = factory.openSession();*

*try {*

*tx.begin();*

*// Do some work*

*session.createQuery(...);*

*session.persist(...);*

*session.flush(); // Extra work you need to do*

*tx.commit();*

*}*

*catch (RuntimeException e) {*

*tx.rollback();*

*throw e; // or display error message*

*}*

*finally {*

*session.close(); // Extra work you need to do*

*}*

*20.What are the types of Hibernate instance states ?*

*Three types of instance states:*

*Transient -The instance is not associated with any persistence context*

*Persistent -The instance is associated with a persistence context*

*Detached -The instance was associated with a persistence context which has been closed – currently not associated*

*21.What are the types of inheritance models in Hibernate?*

*There are three types of inheritance models in Hibernate:*

*Table per class hierarchy*

*Table per subclass*

*Table per concrete class*

*22.What is Hibernate Query Language (HQL)?*

*Hibernate Query Language is query language which is used to develop the data independent query language in the application. This HQL queries are not related to any database. Hibernate offers a query language that embodies a very powerful and flexible mechanism to query, store, update, and retrieve objects from a database. This language, the Hibernate query Language (HQL), is an object-oriented extension to SQL.*

*23.What are the ways to express joins in HQL?*

*HQL provides four ways of expressing (inner and outer) joins:-*

*An implicit association join*

*An ordinary join in the FROM clause*

*A fetch join in the FROM clause.*

*A theta-style join in the WHERE clause.*

*24 . Transaction with plain JDBC in Hibernate ?*

*If you don't have JTA and don't want to deploy it along with your application, you will usually have to fall back to JDBC transaction demarcation. Instead of calling the JDBC API you better use Hibernate's Transaction and the built-in session-per-request functionality:*

*To enable the thread-bound strategy in your Hibernate configuration:*

*set hibernate.transaction.factory\_class to org.hibernate.transaction.JDBCTransactionFactory*

*set hibernate.current\_session\_context\_class to thread*

*Session session = factory.openSession();*

*Transaction tx = null;*

*try {*

*tx = session.beginTransaction();*

*// Do some work*

*session.load(...);*

*session.persist(...);*

*tx.commit(); // Flush happens automatically*

*}*

*catch (RuntimeException e) {*

*tx.rollback();*

*throw e; // or display error message*

*}*

*finally {*

*session.close();*

*}*

*25 . What are the general considerations or best practices for defining your Hibernate persistent classes?*

*1.You must have a default no-argument constructor for your persistent classes and there should be getXXX()and setXXX() methods for all your persistable instance variables.*

*2.You should implement the equals() and hashCode() methods based on your business key and it is important not to use the id field in your equals() and hashCode() definition if the id field is a surrogate key (i.e. Hibernate managed identifier). This is because the Hibernate only generates and sets the field when saving the object.*

*3. It is recommended to implement the Serializable interface. This is potentially useful if you want to migrate around a multi-processor cluster.*

*4.The persistent class should not be final because if it is final then lazy loading cannot be used by creating proxy objects.*

*26 . Difference between session.update() and session.lock() in Hibernate ?*

*The session.update method is used to update the persistence object in the in the database.*

*The session.lock() method simply reattaches the object to the session without checking or updating the database on the assumption that the database in sync with the detached object. It is the best practice to use either session.update(..) or session.saveOrUpdate(). Use session.lock() only if you are absolutely sure that the detached object is in sync with your detached object or if it does not matter because you will be overwriting all the columns that would have changed later on within the same transaction.*

*27.What are the Collection types in Hibernate ?*

*Set*

*List*

*Array*

*Map*

*Bag*

*28.What is the difference between sorted and ordered collection in hibernate?*

*sorted collection vs. order collection :-*

*sorted collection*

*order collection*

*A sorted collection is sorting a collection by utilizing the sorting features provided by the Java collections framework. The sorting occurs in the memory of JVM which running Hibernate, after the data being read from database using java comparator.*

*Order collection is sorting a collection by specifying the order-by clause for sorting this collection when retrieval.*

*If your collection is not large, it will be more efficient way to sort it.*

*If your collection is very large, it will be more efficient way to sort it .*

*29.What are the ways to express joins in HQL?*

*HQL provides four ways of expressing (inner and outer) joins:-*

*An implicit association join*

*An ordinary join in the FROM clause*

*A fetch join in the FROM clause.*

*A theta-style join in the WHERE clause.*

*30.What do you mean by Named – SQL query?*

*Named SQL queries are defined in the mapping xml document and called wherever required.*

*Example:*

*SELECT emp.EMP\_ID AS {emp.empid},*

*emp.EMP\_ADDRESS AS {emp.address},*

*emp.EMP\_NAME AS {emp.name}*

*FROM Employee EMP WHERE emp.NAME LIKE :name*

*Invoke Named Query :*

*List people = session.getNamedQuery("empdetails")*

*.setString("TomBrady", name)*

*.setMaxResults(50)*

*.list();*

*31.How do you invoke Stored Procedures?*

*{ ? = call selectAllEmployees() }*

*32.Explain Criteria API*

*The interface org.hibernate.Criteria represents a query against a particular persistent class. The Session is a factory for Criteria instances. Criteria is a simplified API for retrieving entities by composing Criterion objects. This is a very convenient approach for functionality like "search" screens where there is a variable number of conditions to be placed upon the result set.*

*Example :*

*List employees = session.createCriteria(Employee.class)*

*.add(Restrictions.like("name", "a%") )*

*.add(Restrictions.like("address", "Boston"))*

*.addOrder(Order.asc("name") )*

*.list();*

*33.What’s the difference between load() and get()?*

*load()*

*get()*

*Only use the load() method if you are sure that the object exists.*

*If you are not sure that the object exists, then use one of the get() methods.*

*load() method will throw an exception if the unique id is not found in the database.*

*get() method will return null if the unique id is not found in the database.*

*load() just returns a proxy by default and database won’t be hit until the proxy is first invoked.*

*get() will hit the database immediately.*

*34.What is the difference between and merge and update ?*

*Use update() if you are sure that the session does not contain an already persistent instance with the same identifier, and merge() if you want to merge your modifications at any time without consideration of the state of the session.*

*35.Define cascade and inverse option in one-many mapping?*

*cascade - enable operations to cascade to child entities.*

*cascade="all|none|save-update|delete|all-delete-orphan"*

*inverse - mark this collection as the "inverse" end of a bidirectional association.*

*inverse="true|false"*

*Essentially "inverse" indicates which end of a relationship should be ignored, so when persisting a parent who has a collection of children, should you ask the parent for its list of children, or ask the children who the parents are?*

*36.Define HibernateTemplate?*

*org.springframework.orm.hibernate.HibernateTemplate is a helper class which provides different methods for querying/retrieving data from the database. It also converts checked HibernateExceptions into unchecked DataAccessExceptions.*

*37.What are the benefits does HibernateTemplate provide?*

*The benefits of HibernateTemplate are :*

*HibernateTemplate, a Spring Template class simplifies interactions with Hibernate Session.*

*Common functions are simplified to single method calls.*

*Sessions are automatically closed.*

*Exceptions are automatically caught and converted to runtime exceptions.*

*38. How do you switch between relational databases without code changes?*

*Using Hibernate SQL Dialects , we can switch databases. Hibernate will generate appropriate hql queries based on the dialect defined.*

*39.If you want to see the Hibernate generated SQL statements on console, what should we do?*

*By using “show\_sql” property of the hibernate configuration file*

*In Hibernate configuration file set as follows:*

*true*

*40.What are derived properties?*

*The properties that are not mapped to a column, but calculated at runtime by evaluation of an expression are called derived properties. The expression can be defined using the formula attribute of the element.*

*41.Define cascade and inverse option in one-many mapping?*

*cascade - enable operations to cascade to child entities.*

*cascade="all|none|save-update|delete|all-delete-orphan"*

*inverse - mark this collection as the "inverse" end of a bidirectional association.*

*inverse="true|false"*

*Essentially "inverse" indicates which end of a relationship should be ignored, so when persisting a parent who has a collection of children, should you ask the parent for its list of children, or ask the children who the parents are?*

*42 . Explain about transaction file?*

*Transactions denote a work file which can save changes made or revert back the changes. A transaction can be started by session.beginTransaction() and it uses JDBC connection, CORBA or JTA. When this session starts several transactions may occur.*

*43 . Difference between session.save() , session.saveOrUpdate() and session.persist()?*

*All methods are used to store the data in to database*

*session.save() : save() method uSave does an insert and will fail if the primary key is already persistent.*

*session.saveOrUpdate() : saveOrUpdate() insert the data in the database if that primary key data not available and it update the data if primary key data not availabt*

*session.persist() :it is the same like session.save(). But session.save() return Serializable object but session.persist() return void.*

*For Example :*

*if you do :-*

*System.out.println(session.save(question));*

*This will print the generated primary key.*

*if you do :-*

*System.out.println(session.persist(question));*

*Compile time error because session.persist() return void.*

*44 . Explain about the id field?*

*This id field is used to configure the primary key in the mapping file, and also we can configure primary key generation algorithm.*

*45.What is the use of dynamic-insert and dynamic-update attributes in a class mapping?*

*Criteria is a simplified API for retrieving entities by composing Criterion objects. This is a very convenient approach for functionality like "search" screens where there is a variable number of conditions to be placed upon the result set.*

*dynamic-update (defaults to false): Specifies that UPDATE SQL should be generated at runtime and contain only those columns whose values have changed*

*dynamic-insert (defaults to false): Specifies that INSERT SQL should be generated at runtime and contain only the columns whose values are not null.*

*46.What is automatic dirty checking?*

*Automatic dirty checking is a feature that saves us the effort of explicitly asking Hibernate to update the database when we modify the state of an object inside a transaction.*

*47.What are Callback interfaces?*

*Callback interfaces allow the application to receive a notification when something interesting happens to an object—for example, when an object is loaded, saved, or deleted. Hibernate applications don't need to implement these callbacks, but they're useful for implementing certain kinds of generic functionality.*

*48.What is Hibernate proxy?*

*The proxy attribute enables lazy initialization of persistent instances of the class. Hibernate will initially return CGLIB proxies which implement the named interface. The actual persistent object will be loaded when a method of the proxy is invoked.*

*49.How can Hibernate be configured to access an instance variable directly and not through a setter method ?*

*By mapping the property with access="field" in Hibernate metadata. This forces hibernate to bypass the setter method and access the instance variable directly while initializing a newly loaded object.*

*50.How can a whole class be mapped as immutable?*

*Mark the class as mutable="false" (Default is true),. This specifies that instances of the class are (not) mutable. Immutable classes, may not be updated or deleted by the application.*

*51 . Explain about transparent persistence of Hibernate?*

*Transparent persistence is provided for Plain old Java objects or POJOs. For proper functioning of the applications importance should be given to the methods equals () and hash Code methods (). It has a requirement which should be strictly followed in the applications which is a no-argument constructor.*

*52 . Explain about the dirty checking feature of Hibernate?*

*Dirty checking feature of the Hibernate allows users or developers to avoid time consuming data base write actions. This feature makes necessary updations and changes to the fields which require a change, remaining fields are left unchanged or untouched.*

*Q)what is the difference between String hashcode and normal hash code?*

*A:Here the hashcode method of object class is overridden in String class hence ,both are same,they are overridden hashcode in String class in such a way that it returns 0 for empty String and returns int number for the existing String object.*

*Q)why JVM gives unique hashcode to the objects?*

*A:*

*Hashcode technique:*

*S[0]\*31^(n-1)+ S[1]\*31^(n-2)+…….s[n-1]*

*Feb 11 2017*

*ServiceNow Interview Questions*

1. *I have given 99 numbers from 1 to 100 then find the missed number?*

*Ans)sum of 100 numbers – sum of 99 numbers*

1. *I have given an array of numbers {3,7,2,7,5,6,15,12,4,9} ,write a logic to find highest product of two numbers and print those two numbers as well*

*Ans) Take first two then start iterating loop from the third numbers in the loop. Lets say*

*{3,7,2,7,5,6,15,12,4,9}*

*First=3*

*Second=7 ‘*

*]*

*Product=First\*Second=3\*7=21..*

*It will compare the fourth and fifth product lets say 2\*7=14.*

*Now it will compare the first two numbers of first product with iterating set of two numbers if any one of two iterating numbers is greater then any one of first two numbers then it will take that maximum number*

*Then 3\*7=21*

*Iteration 2:- 5,6=5\*6=30,*

*Then 6\*7=42*

*Iteration 3:- 15,12=15\*12=180*

*Then 15\*12=180*

*Iteration 4\*9=36.*

*Then 15\*12=180..*

*3)Class.forName(“jdbc:oracle:thin:localhost:1521:XE”);*

*How we can load it and execute it?*

*4)*

*Prokrama by Rajesh*

*1)Try with Resources added in java7*

**package** com.interview;

**import** java.io.BufferedReader;

**import** java.io.FileReader;

**import** java.io.IOException;

//Added in Java 7

**public** **class** TryWithResourcesDemo {

**public** **static** **void** main(String[] args) {

**try** (BufferedReader br = **new** BufferedReader(**new** FileReader(

"F:/New/ram.txt"))) {

System.*out*.println(br.readLine());

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

*2)Array and ArrayList?*

*Ans)*

**Difference between Array and ArrayList in Java with Example**  
  
**1. Resizable :**   Array is static in size that is fixed length data structure, One can not change the length after creating the Array object.  
ArrayList is dynamic in size . Each ArrayList object  has instance variable *capacity* which indicates the size of the ArrayList. As elements are added to an ArrayList its capacity grows automatically.  
  
**2. Performance :** Performance of Array and ArrayList depends on the operation you are performing :  
  
*resize() opertation :* Automatic resize of ArrayList will slow down the performance as it will use temporary array to copy elements from the old array to new array.  
ArrayList is internally backed by Array during resizing  as it calls the native implemented method System.arrayCopy(src,srcPos,dest,destPos,length) .  
  
  
  
*add() or get() operation :* adding an element or retrieving an element from the array or arraylist object has almost same  performance , as for ArrayList object these operations  run in constant time.  
  
**3. Primitives :**  ArrayList can not contains primitive data types (like int , float , double) it can only contains Object while Array can contain both primitive data types as well as objects.  
One get a misconception that we can store primitives(int,float,double) in ArrayList , but it is not true    
  
Suppose we have ArrayList object ,

ArrayList  arraylistobject = new ArrayList();  
arraylistobject.add(**23**);  // try to add 23 (primitive)

JVM through Autoboxing(converting primitives to equivalent objects internally) ensures that only objects are added to the arraylist object.   
thus , above step internally works like this :

arraylistobject.add( **new Integer(23)**);         
// Converted int primitive to Integer object and added to arraylistobject

[](http://1.bp.blogspot.com/-WsUtI3c7wrQ/Uu9tohX5HLI/AAAAAAAAAJw/9PMdWlBvu-s/s1600/java-collections-tutorial.png)

**4. Iterating the values :** We can use iterator  to iterate through ArrayList . The iterators returned by the ArrayList class's iterator and listiterator method are [fail-fast](http://javahungry.blogspot.ca/2014/04/fail-fast-iterator-vs-fail-safe-iterator-difference-with-example-in-java.html).  We can use for loop or for each loop to iterate through array .    
  
**5. Type-Safety :**In Java , one can ensure Type Safety through Generics. while Array is a homogeneous data structure , thus it will contain objects of specific class or primitives of specific  data type. In array if one try to store the different data type other than the specified while creating the array object , ArrayStoreException is thrown.  
  
for example :

String temp[] =  new String[2];  // creates a string array of size 2  
temp[0] = new Integer(12); // throws ArrayStoreException, trying to add Integer object in String[]

**6. Length :**Length of the ArrayList is provided by the size() method while Each array object has the length variable which returns the length of the array.  
  
for example :

Integer arrayobject[] = new Integer[3];  
arraylength= arrayobject.length   ;  //uses arrayobject length variable

ArrayList  arraylistobject = new ArrayList();  
arraylistobject.add(12);   
arraylistobject.size();   //uses arraylistobject size method

**7. Adding elements :** We can insert elements into the arraylist object using the add() method while  in array we insert elements using the assignment operator.  
  
  
for example :

Integer addarrayobject[] = new Integer[3];  
addarrayobject[0]= new Integer(8)   ;  //new object is added to the array object

**8. Multi-dimensional :**Array can be multi dimensional , while ArrayList is always single dimensional.  
  
example of multidimensional array:

Integer addarrayobject[][] = new Integer[3][2];  
addarrayobject[0][0]= new Integer(8)

    
**Example of Array and ArrayList**    
 

**import** **java.util.ArrayList**;

**import** **java.util.Iterator**;

**public** **class** **ArrayArrayListExample** {

**public** **static** **void** **main**(String[] args) {

// ArrayList Example

 ArrayList<String> arrlistobj = **new** ArrayList<String>();

arrlistobj.add("Alive is awesome");

arrlistobj.add("Love yourself");

  Iterator it = arrlistobj.iterator();

System.out.print("*ArrayList object output :*");

  while(it.hasNext())

  System.out.print(it.next() + " ");

  // Array Example

String[] arrayobj = new String[3];

arrayobj[0]= "Love yourself";

arrayobj[1]= "Alive is awesome";

arrayobj[2]= "Be in Present";

System.out.print("*Array object output :*");

  for(int i=0; i < arrayobj.length ;i++)

  System.out.print(arrayobj[i] + " ");

 }

}

**Output :***ArrayList object output :****{ Alive is awesome , Love yourself }***  
                 *Array object output :****{ Love yourself , Alive is awesome, Be in present}***  
      
  
  
**Similarities Between Array and ArrayList**  
  
**1. add and get method :**Performance of Array and ArrayList are similar for the add and get operations .Both operations runs in constant time.  
  
**2. Duplicate elements :**Both array and arraylist can contain duplicate elements.  
  
**3. Null Values :**Both can store null values and uses index to refer to their elements.  
  
**4. Unordered :**  Both does not guarantee ordered  elements.  
  
  
  
  
**Recap : Difference between Array and ArrayList in Java** 

|  |  |  |
| --- | --- | --- |
|  | **Array** | **ArrayList** |
|  |  |  |
| Resizable | No | Yes |
|  |  |  |
| Primitives | Yes | No |
|  |  |  |
| Iterating values | for, for each | Iterator , for each |
|  |  |  |
| Length | length variable | size method |
|  |  |  |
| Performance | Fast | Slow in comparision |
|  |  |  |
| Multidimensional | Yes | No |
| Add Elements | Assignment operator | add method |

In case you have any doubts regarding the difference between array and arraylist  in java then please mention in comments.

1. Diff bw spring jdbc and hibernate and which is better?

Hibernate is a really huge solution with data persistence and ORM including JPA implementation. Also there are defined many ways how to manage entities in Hibernate, how to persist, transactions, etc. In hibernate you can use SQL, HQL or java annotations. JDBC template is just a simple tool that helps you to manage SQL queries and transactins. Probably it is better JDBC wrapper or helper. Also if you prefer managing database queries (SQL) yourself or if you are a beginner, try Spring JdbcTemplate to understand, how it works. Even if you are working on a bigger application, think about using Hibernate. However it needs some time to know Hibernate.

|  |
| --- |
| [share](http://stackoverflow.com/a/17301317)[improve this answer](http://stackoverflow.com/posts/17301317/edit) |

1. What changes u do if I give Java 1.5 project to convert into Java 7 or 8

Ans) 1)check wrapper classes

2)switch with string

3)ArrayList generics added in 1.7

4)Try with resources in 1.7

5)

1. Questions on exception handling and Custom exception handling program ?

Ans)

**package** com.interview;

**class** InvalidAgeException **extends** Exception{

InvalidAgeException(String s){

**super**(s);

}

}

**public** **class** TestCustomException1{

**static** **void** validate(**int** age)**throws** InvalidAgeException{

**if**(age<18)

**throw** **new** InvalidAgeException("not valid");

**else**

System.*out*.println("welcome to vote");

}

**public** **static** **void** main(String args[]){

**try**{

*validate*(13);

}**catch**(Throwable m){System.*out*.println("Exception occured: "+m);}

System.*out*.println("rest of the code...");

}

}

1. Can we write catch(throwable e) ?

Ans)Yes

1. Hibernate configuration in spring XML file?

Ans)

1. <bean id="dataSource" **class**="org.apache.commons.dbcp.BasicDataSource">
2. <property name="driverClassName"  value="oracle.jdbc.driver.OracleDriver"></property>
3. <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"></property>
4. <property name="username" value="system"></property>
5. <property name="password" value="oracle"></property>
6. </bean>
8. <bean id="mysessionFactory"  **class**="org.springframework.orm.hibernate3.LocalSessionFactoryBean">
9. <property name="dataSource" ref="dataSource"></property>
11. <property name="mappingResources">
12. <list>
13. <value>employee.hbm.xml</value>
14. </list>
15. </property>
17. <property name="hibernateProperties">
18. <props>
19. <prop key="hibernate.dialect">org.hibernate.dialect.Oracle9Dialect</prop>
20. <prop key="hibernate.hbm2ddl.auto">update</prop>
21. <prop key="hibernate.show\_sql">**true**</prop>
23. </props>
24. </property>
25. </bean>
27. <bean id="template" **class**="org.springframework.orm.hibernate3.HibernateTemplate">
28. <property name="sessionFactory" ref="mysessionFactory"></property>
29. </bean>
31. <bean id="d" **class**="com.javatpoint.EmployeeDao">
32. <property name="template" ref="template"></property>
33. </bean>
34. How many ways u can iterate hash map?

Ans)

Looping over a Map in Java. In this post we look at four different ways we can iterate through a map in Java. We can use for-each loop as well as the iterator class.

### Method #1: Iterating over entries using For-Each loop

Java



|  |  |
| --- | --- |
| 1  2  3  4 | Map<Integer, Integer> map = new HashMap<Integer, Integer>();  for (Map.Entry<Integer, Integer> entry : map.entrySet()) {      System.out.println("Key = " + entry.getKey() + ", Value = " + entry.getValue());  } |

### Method #2: Iterating over keys or values using For-Each loop

Java



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | Map<Integer, Integer> map = new HashMap<Integer, Integer>();    //iterating over keys only  for (Integer key : map.keySet()) {      System.out.println("Key = " + key);  }    //iterating over values only  for (Integer value : map.values()) {      System.out.println("Value = " + value);  } |

*3)*

### Method #3: Iterating using Iterator.

Using Generics:



|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Map<Integer, Integer> map = new HashMap<Integer, Integer>();  Iterator<Map.Entry<Integer, Integer>> entries = map.entrySet().iterator();  while (entries.hasNext()) {      Map.Entry<Integer, Integer> entry = entries.next();      System.out.println("Key = " + entry.getKey() + ", Value = " + entry.getValue());  } |

Without Generics:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | Map map = new HashMap();  Iterator entries = map.entrySet().iterator();  while (entries.hasNext()) {      Map.Entry entry = (Map.Entry) entries.next();      Integer key = (Integer)entry.getKey();      Integer value = (Integer)entry.getValue();      System.out.println("Key = " + key + ", Value = " + value);  } |

*5)*

### Method #4: Iterating over keys and searching for values



|  |  |
| --- | --- |
| 1  2  3  4  5 | Map<Integer, Integer> map = new HashMap<Integer, Integer>();  for (Integer key : map.keySet()) {      Integer value = map.get(key);      System.out.println("Key = " + key + ", Value = " + value);  } |

1. Diff between outofmemory and heap exception?

### *9)* What Are Memory Leaks?

Let’s start by describing how memory leaks happen in Java. Java implements automatic garbage collection (GC), and once you stop using an object you can depend on the garbage collector to collect it. While additional details of the collection process are important when tuning GC performance, for the sole purpose of fixing a memory leak we can safely ignore them.

When is memory eligible for GC? Let’s take a look at an example:

We don't have to do anything special to make an object eligible for GC—we just eliminate any references to it, and it "magically" disappears and stops using memory. That's why we say that Java performs "automatic" GC.  
  
Why "eligible" for GC? Because objects are not collected immediately. GC is not instantaneous and comes with some performance impacts. Consequently, Java doesn't immediately collect every object that is eligible for collection; it typically postpones collection until a more convenient time later on. The way to think about GC in Java is that it's a "lazy bachelor" that hates taking out the trash and typically postpones the process for some period of time. However, if the trash can begins to overflow, Java immediately takes it out. In other words, if memory becomes scarce, Java immediately runs GC to free memory.  
  
Since we don't need to do anything special in order to dispose of objects in Java, how do memory leaks happen in Java? Memory leaks occur when a program never stops using an object, thus keeping a permanent reference to it.  
  
Let's take a look at an example that helps illustrate this point. The following code will cause all available memory in the JVM to be exhausted:  
  
  
  
When no more memory is remaining, an OutOfMemoryError alert will be thrown and generate an exception like this:  
  
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space at  
MemoryLeakDemo.main(MemoryLeakDemo.java:14)  
  
In the example above, we continue adding new elements to the list memoryLeakArea without ever removing them. In addition, we keep references to the memoryLeakArea, thereby preventing GC from collecting the list itself. So although there is GC available, it cannot help because we are still using memory. The more time passes the more memory we use, which in effect requires an infinite amount memory for this program to continue running.  
  
This is an example of unbounded memory leak—the longer the program runs, the more memory it takes. So even if the memory size is increased, the application will still run out of memory at a later date.

## Meat & Potatoes

### Fixing Memory Leaks

As illustrated by the flowchart below, the process of fixing memory leaks is fairly simple:

Memory leaks are misunderstood creatures. Just getting an OutOfMemoryError alert doesn’t necessary mean that you're suffering from a memory leak. So, before you dive into "fixing" the problem, you must first find out whether or not a memory leak actually exists. If a memory leak does in fact exist, the next step is to determine which objects are leaking and uncover the source of the memory leak. Then, you fix it.  
  
We'll skip past the initial steps and dive right into diagnosing whether or not the problem is a memory leak.

### Is My Program Leaking Memory?

Not every OutOfMemoryError alert indicates that a program is suffering from a memory leak. Some programs simply need more memory to run. In other words, some OutOfMemoryError alerts are caused by the load, not by the passage of time, and as a result they indicate the need for more memory in a program rather than a memory leak.

To distinguish between a memory leak and an application that simply needs more memory, we need to look at the "peak load" concept. When program has just started no users have yet used it, and as a result it typically needs much less memory then when thousands of users are interacting with it. Thus, measuring memory usage immediately after a program starts is not the best way to gauge how much memory it needs! To measure how much memory an application needs, memory size measurements should be taken at the time of peak load—when it is most heavily used.  
  
The graph below shows the memory usage in a healthy Java application that does not suffer from memory leaks, with the peak load occurring around 10 AM and application usage drastically decreasing at 5 PM. Naturally, the peak load on business applications often correlates with normal business hours.  
  
  
  
The application illustrated by the chart above reaches its peak load around 10 AM and needs around 900MB of memory to run. This is normal behavior for an application suffering from no memory leaks; the difference in memory requirements throughout the day is caused solely by the user load.  
  
Now, let's suppose that we have a memory leak in the application. The primary characteristic of memory leaks is that memory requirements increase as a function of time, not as a function of the load. Let's see how the application would look after running for a few days with a memory leak and the same peak user loads reached around 10 AM every day:

Because peak loads on the system are similar every morning but memory usage is growing over a period of a few days, this picture indicates a strong possibility of memory leaks. If the program eventually started suffering from OutOfMemory exceptions, it would be a very strong indication that there’s a problem with memory leaks. The picture above shows a memory leak of about 100MB per day.

Note that the key to this example is that the only thing changing is the amount of time the system is up—the system peak load doesn’t change over time. This is not the case for all businesses. For example, the peak load for a tax preparation service is seasonal, as there are likely more users on the system in April than July.

There is one special case that should be noted here: a program that needs to be restarted periodically in order to prevent it from crashing with an OutOfMemoryError alert. Imagine that on the previous graph the max memory size was 1100MB. If the program started with about 900MB of memory used, it would take about 48 hours to crash because it leaks about 100MB of memory per day. Similarly, if the max memory size was set to 1000MB, the program would crash every 24 hours. However, if the program was regularly restarted more often than this interval, it would appear that all is fine.

Regularly scheduled restarts may appear to help, but also might make “upward sloping memory use” (as shown in the previous graph) more difficult to notice because the graph is cut short before the pattern emerges. In a case like this, you’ll need to look more carefully at the memory usage, or try to increase the available memory so that it’s easier to see the pattern.

### Monitoring Memory in Java

As you are already aware, you need to measure memory that is free and used inside of the JVM, not memory that the JVM process is using. In other words, the top/Task Manager/Activity Monitor will measure how much memory your Java process is using, but that’s not what you need. You need a tool that can look inside the JVM process and tell you how much memory is available for your program running inside the JVM.

In addition, keep in mind that Java GC is not a constant process—it runs in intervals. The memory usage that you see in the JVM is usually higher then what your program needs at the moment, as GC hasn’t yet run. Remember, lazy bachelors usually have some trash in their apartments waiting to be taken out.

*10)ApplicationContext and BeanFactory?*

**Bean Factory**

* Bean instantiation/wiring

**Application Context**

* Bean instantiation/wiring
* Automatic BeanPostProcessor registration
* Automatic BeanFactoryPostProcessor registration
* Convenient MessageSource access (for i18n)
* ApplicationEvent publication

So if you need any of the points presented on the Application Context side, you should use ApplicationContext.

Nisum Technologies ,Gachibowli, Feb 13 2017

1)what is immutable in java?

Ans)The string is an immutable object .once the object is created it wont be changed…

If I add data to the existing string it will create new object in the memory rather then adding to the existing string object…

2)Write immutable class ?

Ans)

3)What is use of ResponseBody annotation in spring mvc ?

4)What is the use of @Produces and @Consumes Annotations in Restful web services?

5)Which one is secure SOAP or Rest?

6)SOAP Protocol uses which

7)Write Program on Singleton Design Pattern?

8)Diff HashSet and TreeSet?

9)What is the Diff Comparable and Comparator?

10)Can we have customized employee object in the key of HashMap?

11)What is Spring IOC Container?

@Autowired

Product product;

Prokarma Software Technologies On Feb 18 2017

1)I have HTML page having css files and Js files included .How Browser will read this is the right synax for css class like syntax:- .classs

{

}

How Browser Engine will work?.

2)How you are consuming SOAP web service in your project?

3)What is the Return type of Map.put(“key”,”value”)?

4)How Map removes Duplicate Keys ?

5)I have one jsp,servlet and web.xml is there how you can convert it into Spring MVC application?

6) I have client and app server .one request is passed which will take it directly servlet or app server?

7)Are you good in Angular JS –Said just stared training as part of new project Reqmnt.

8)What is the return type of Set.add() ?

9) I have Javascript method like below im passing One Object as a parameter in that function .

In that method definition you just check if that object is null or undefined then return empty string or else

Return same object .. write the code for it?(**Without using if conditions(any conditional statements or Ternary Operator**)

10)What is SPY in Mockito Framework?

11)In My spring core java application im loading spring configuration file im having Spring ApplicationContext object I can get the bean lets say Demo demo=applicationContext.getBean(“beanid”); But here I don’t know my bean id lets assume .Without knowing Bean Id how can we load that Bean here….

12)What is the Diff Set and List ?

13) I have HashSet collection added three objects

HashSet s=new HashSet();

s.add(“a”);

s.add(“b”);

s.add(“c”);

Now Write one method in java class takes this Set collection and return second object from the Set. Write the code for it?(**Without using Iterator and counters and for loops or any for each**)

14)What are the methods of Set Collections?

15)Tell me one Singleton class in Java API?

16)Write the code of Singleton Design Pattern?

17)What is the use of volatile keyword and can we apply volatile to all like methods or classes and variables ?

18)Can you write the JUNIT or TestNG Testclass for Singleton class?

19)Let us assume in testclass one method is taking different objects like creditcard object or deposit object or etc i.e passing as argument based on the object it return different value .How we can write the test class and what are the annotations we can use either in Junit or TestNG?

20) I have Spring bean is there Demo having

Class Demo

{

Void m1()

{

Sop(“Parasuram”);

}

Void m2()

{

Sop(“Hello”);

}

Void m3()

{

Sop(“Yerramsetty”);

}

Main class

{

Psvm()

{

Demo demo=applicationContext.getBean(“demo”);

Demo.m1();

}

Spring XML Bean is created like this

<bean id=”demo” class=”Demo”>

</bean>

Without Changing the Java code ,you can change the spring configuration code to execute all the three methods and print the output :-

21) Who will Marshall and Unmarshall the Object in SOAP?

22) What is XSD and why we can change it in SOAP?

23)Which Version Spring Jars are using ?

24)Which Java version using and tell me Java 8 Features and Lamda Expression?

25)SOAP and REST Uses which Protocols ?

26) Did you work on Jenkins

27)Do you know Sonar Cubes ?

28)How cucumber works ?

29)