*<{{{{{*

*Spring-rest annotation*

*@RequestBody, @responseBody, @PathVariable,@RequestParam*

@RequestMapping(value="/{id}", method=RequestMethod.GET)

public ResponseEntity<Spittle> spittleById(@PathVariable long id) {

Spittle spittle = spittleRepository.findOne(id);

HttpStatus status = spittle != null ?

HttpStatus.OK : HttpStatus.NOT\_FOUND;

return new ResponseEntity<Spittle>(spittle, status);

}

Returns error and resp body

There’s no need to annotate

the method with @ResponseBody if it returns ResponseEntity.

*--* Catch(Exception e ) can not catch an error as both are siblings. So, in order to catch the errors we must use catch(Errors e);

+ Interview Questions:

-Difference between Object oriented language and procedural language

-iterate List using jstl.

<c:forEach items=”${controller\_returned\_value}” var=”any\_variable\_name”>

<c:out value=”${ any\_variable\_name.getXXX()}”/>

</c:forEach>

For List<Object>

<c:forEach items="${list}" var="item">

${item}<br>

</c:forEach>

Is similar to

for (Object item : list) {

System.out.println(item);

}

For List<Map<k,v>>

<c:forEach items="${list}" var="map">

<c:forEach items="${map}" var="entry">

${entry.key}<br>

${entry.value}<br>

</c:forEach>

</c:forEach>

Is similar to

for (Map<K, V> map : list) {

for (Entry<K, V> entry : map.entrySet()) {

System.out.println(entry.getKey());

System.out.println(entry.getValue());

}

}

1.Explain your project architecture?

2.Explain spring MVC flow?

3.List out all the annotations in spring?

@Bean, @Autowired, @Inject, @Named, @Resource,@Qualifier, @Primary, @component, @Repository, @Controller, @Service,@PathVariable, @RequestParam, @ModelAttribute, @Configuration, @EnableWebMvc, @ComponentScan, @RequestBody, @ResponseBody,

@PropertySource.

4.what are all the annotations you used in your project on restful services?

7.what is diff b/w ArrayList and LinkedList?

8.can you explain the internal flow of HashMap?

9.what is the diff b/w HashMap and Hashtable?

10.Diff b/w Array and ArrayList?

11.Diff b/w ArrayList and Vector?

12.In your project where you used cuncurrent hashmap?

13.What is java annoying?

14.Diff b/w callable interface and future interface in concurrent package?

Callable and Runnable both are interfaces whose instances are executed by a thread. But Callable is capable of returning some values whereas runnable is not. Runnable can be executed by Thread class as well as ExecutorService where as callable can be executed by ExecutorService only. The value returned after executorService.submit(new callable<Integer>) is Future<Integer>. Callable class method throws Checked Exception.

15.What is class loaders?

16.How can you take List into Map?

17.How can you take Map into List? List<Map<integer,string>>

18.when you will get ClassNotFoundException and NoClassDefFoundError?

19.How you implement exception handling in your project?

20.where you implement multi-threading in your project?

21.what are all the design patterens you observed in spring?

* Singleton Pattern: Singleton-scoped beans
* Factory Pattern: Bean Factory classes
* Prototype Pattern: Prototype-scoped beans
* Adapter Pattern: Spring Web and Spring MVC
* Proxy Pattern: Spring Aspect Oriented Programming support
* Template Method Pattern: *JdbcTemplate*, *HibernateTemplate,* etc.
* Front Controller: Spring MVC *DispatcherServlet*
* Data Access Object: Spring DAO support
* Model View Controller: Spring MVC

22.which design patters you used in your project?

23.what are all the critial situations you come across in your project?

24.why wait() placed in object class ? why not it is placed in Thread class?

9973947777

25.what is use of intern() in spring?

Intern checks whether this string object is coming from scp or not, if it’s not coming from scp it simply put it there and return the object. Let’s say:

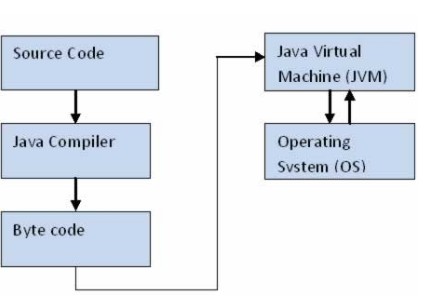
String a="a"; String b=new String("a");

In above (a==b) -🡪 false

But if we do String c=b.intern(); the (a==c)🡪 true

26.what is diff b/w String str="xyz"; and String str2= new String("xyz");

27.Expalin about java architectue?



-Difference between Collection and Collections?

Collection is root interface of collections framework. Collection represents the group of individual objects as a single unit.Collection is root interface for data structures like List, Set, Queue.Whereas Collections is utility class which work for collection. All the methods in Collections are Static. They are used as utitlity for collection like if we want to find maximum and minimum elements in collection or we want to sort the unsorted collection then we can use Collections methods.

28.Explain about jvm architecture?

29.Data base queries?

30.I have a company table in remote database. by using rest

i need to get the table data and print into a file?

31.how to read book pages on online library by using bookid or author id(by using restful services)?

32.I have a table in remote database, how to update the data in that table using rest?

34.Agile methodolgy?

35.How to create web-services project and spring project using mavan?

36.what is diff b/w throw and throws?

37.can you tell me java8 features?

38.what are all the contents in wsdl?

39.Refer regular expressions?

40.can i add elements to list , if it is defined as final

ex:final List<String> list= new ArrayList<>();?

41.if you pass duplicate key to map what will happen?

42.Diff b/w abstract class and interface?

43.Diff b/w comparator and comparable?

44.How to compare two database tables(clue: comparator, compare(),

you have to compare database objects.)?

45.How to set timeout for the browser?(clue: restful client api.)?

46.what workflow you used in your project?

47.why java? why not c & c++?

48.In written test they are asking sorting programs(bubblesort,quicksort,...)?

49.what is time complexity? if you are going to implement sorting by

your own which sorting you prefer? and why?

50.what is the use of volatile and synchronized?

51.what is serialization? have you implement serialization in your project?

52.programs on io streams?

53.why we are using @qualifier?

54.Diff b/w BeanFactory and ApplicationContext?

55.Explain about IOC container?

56.programs on string manipulations?(they are expecting solve by using regular expressions).

57.How you are implemented polymorphism in your project?

58.How you iterate map having key and list<values>?

59.Diff b/w Iterator and ListIterator and Enumarator?

60.what are all the collections are supporting ListIterator?

61.How to make non-synchronized map and list as synchronized(by using collection method)?

62.what is diff b/w collection and collections?

63.write the junit test case for the below senario..

-->read array of elements into list<>.

64.what are all the modifiers we can use inside method?(ans: only final)

65.what is diff b/w spring-jdbc and hibernate?

66.what are all the drawbacks of jdbc over hibernate?

67.what are all the problems with inheritance?

68.what is the use of hibernate session?

69.They given one query in sql and they are asking corresponding criteria api query?

70.why we are using @transient in hibernate?

71.what are all the inputs we are giving to SessionFactory?

72.what we are writing in hibernate-mapping file?

73.what we are writing in hibernate-configuration file?

74.senario: in jsp page with 2 buttons, one for addbook and another is for

showListOfBooks(by using spring and hibernate)?

75.what is use of @ComponentScan?

76.what is use of dispatcher servlet?

77.what are all the pre-processings tasks done by DispatcherServlet?

78.How to render excel and pdf view to the enduser(using poi and itext api's)?

79.How to validate valid username and password in spring?for validating

can i directly interact with dao without service?

80.By defalut servlet container will handle multi-threaded applications ,

then why you are implementing multi-threading in your application

*}}}}}}*

*Class Loading, Linking, and Initialization*

The Java Virtual Machine makes types available to the running program through a process of *loading*,

*linking*, and *initialization*. Loading is the process of bringing a binary form for a type into the Java

Virtual Machine. Linking is the process of incorporating the binary type data into the runtime state of

the virtual machine. Linking is divided into three sub-steps: *verification*, *preparation*, and *resolution*.

Verification: ensures the type is properly formed and fit for use by the Java Virtual Machine. Preparation:

involves allocating memory needed by the type, such as memory for any class variables. Resolution: is

the process of transforming symbolic references in the constant pool into direct references.

Implementations may delay the resolution step until each symbolic reference is actually used by the

running program. After verification, preparation, and (optionally) resolution are completed, the type is

ready for initialization. During initialization, the class variables are given their proper initial values. See

Figure 7-1 for a graphical depiction of this process.

Host:port/Path?QueryString#fragments

Ipaddress/contextpath/pathexpression/querystring/fragments

Host:port address of resource in network

http:*//example.com/customers?lastName=Burke&zipcode=02115*

A specific parameter name can be repeated in the query string. In this case, there are

multiple values for the same parameter.

The last part of the URI is the fragment. It is delimited by a “#” character. The fragment

is usually used to point to a certain place in the document you are querying.

Not all characters are allowed within a URI string. Some characters must be encoded

using the following rules. The characters a–z, A–Z, 0–9, ., -, \*, and \_ remain the same.

The space character is converted to +. The other characters are first converted into a

sequence of bytes using a specific encoding scheme. Next, a two-digit hexadecimal

number prefixed by % represents each byte.

Implement asynchronous rest client call two rest in oneclient and process them asynchronously ( if resource class using drools if client call two services) in project

<collection internal working check bookmark>

-Difference between ClassNotFound and NoClassDefFound:

We get both the exceptions when jvm can not find the given class in class path. ClassNotFound is a checked exception which we get when we try to load the given class using fully qualified name, using class.forName() or ClassLoader.loadClass() or classLoader.findSystemClass().we get NoClassDefFound exception when we try to instantiate a class using new keyword or a method call, when jvm can not find this class in classpath. This exception occurs when compiler compiled the code successfully but while instantiating it can not locate the class in classpath.

-what is spring boot advantages of spring boot?

Spring boot is a part of springframework. Using springboot we can build production ready application in very less time. Also we don’t have to worry about transitive dependencies and compatible jars. With the use of spring boot starters we can get the compatible dependencies for our project (spring boot starters are dependency discriptors which contain set of dependencies for the particular technology ). Spring boot has also provided embedded servers to run our web applications.

<SPRING-BOOT>::

Q:What is Spring Boot?  
A: Over the years spring has become more and more complex as new functionalities have been added. Just visit the page-<https://spring.io/projects> and we will see all the spring projects we can use in our application for different functionalities. If one has to start a new spring project we have to add build path or add maven dependencies, configure application server, add spring configuration . So a lot of effort is required to start a new spring project as we have to currently do everything from scratch. Spring Boot is the solution to this problem. Spring boot has been built on top of existing spring framework. Using spring boot we avoid all the boilerplate code and configurations that we had to do previously. Spring boot thus helps us use the existing Spring functionalities more robustly and with minimum efforts.  
  
  
Q: What are advantages of Spring Boot ?  
A: The advantages of Spring Boot are

* Reduce Developement, Testing time and efforts.
* Use of JavaConfig helps avoid usage of XML.
* Avoid lots of maven imports and the various version conflicts.
* Provide Opinionated Development approach.
* Quick start to development by providing defaults.
* No Separate Web Server Needed. Which means that you no longer have to boot up Tomcat, Glassfish, or anything else.
* Requires less configuration-Since there is no web.xml file. Simply add classes annotated with@Configuration and then you can add methods annotated with @Bean, and Spring will automagically load up the object and manage it like it always has. You can even add @Autowired to the bean method to have Spring autowire in dependencies needed for the bean.
* Environment Based Configuration-Using these properties, you can pass into the application which environment you are using with: -Dspring.profiles.active={enviornment}. Spring will then load up the subsequent application properties file at (application-{environment}.properties) after loading up the main application properties file.

Q: Which build tool have you used to develop Spring Boot Application ?  
A: Spring Boot application can be developed using Maven as well as Gradle.   
[Spring Boot application using Maven](http://www.javainuse.com/spring/SpringBoot_HelloWorld)   
[Spring Boot application using Gradle](http://www.javainuse.com/spring/SpringBoot_HelloWorld_gradle)   
  
Q: What is JavaConfig?  
A: Spring JavaConfig is a product of the Spring community that provides a pure-Java approach to configuring the Spring IoC Container. It thus helps avoid using XML configurations. The advantages of using JavaConfig are  
The advantages of JavaConfig are

* Object-oriented configuration. Because configurations are defined as classes in JavaConfig, users can take full advantage of object-oriented features in Java. One configuration class may subclass another, overriding its @Bean methods, etc.
* Reduced or eliminated XML configuration. The benefits of externalized configuration based on the principles of dependency injection have been proven. However, many developers would prefer not to switch back and forth between XML and Java. JavaConfig provides developers with a pure-Java approach to configuring the Spring container that is conceptually similar to XML configuration. It is technically possible to configure the container using only JavaConfig configuration classes, however in practice many have found it ideal to mix-and-match JavaConfig with XML.
* Type-safe and refactoring-friendly. JavaConfig provides a type-safe approach to configuring the Spring container. Thanks to Java 5.0's support for generics, it is now possible to retrieve beans by type rather than by name, free of any casting or string-based lookups.

Q:How to reload my changes on Spring Boot without having to restart server?  
A: This can be achieved using DEV Tools. With this dependency any changes you save, the embedded tomcat will restart. Spring Boot has a Developer tools (DevTools) module which helps to improve the productivity of developers. One of the key challenge for the Java developers is to auto deploy the file changes to server and auto restart the server. Developers can reload changes on Spring Boot without having to restart my server. This will eliminates the need for manually deploying the changes every time. Spring Boot doesn’t have this feature when it has released it’s first version. This was a most requested features for the developers. The module DevTools does exactly what is needed for the developers. This module will be disabled in the production environment. It also provides H2-database console for better testing the application. The following dependency is used

<dependency>

<groupId>org.springframework.boot</groupId>

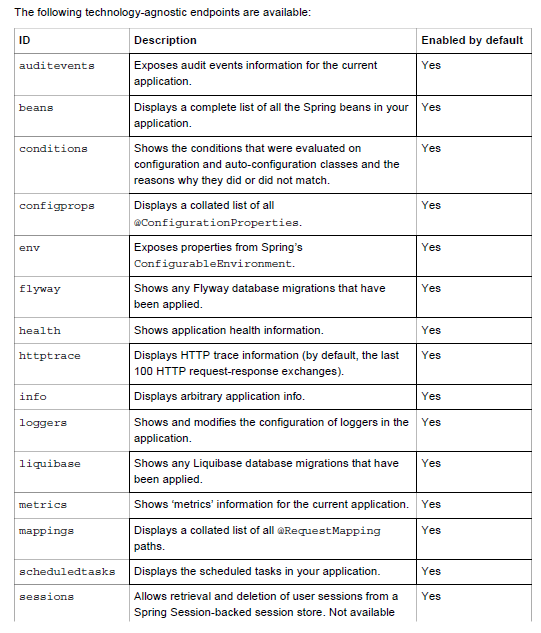
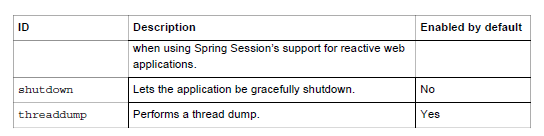
<artifactId>spring-boot-devtools</artifactId>

<optional>true</optional>

</dependency>

The DevTool dependency usage for autorestart and H2 DB

Q:What is Actuator in Spring Boot?  
A: Spring boot actuator is one of the important feature in spring boot framework. Spring boot actuator helps you to access the current state of the running application in production environment. There are several metrics that has to be checked and monitored in the production environment. Even some external applications may be using those services to trigger the alert message to concerned person. Actuator module exposes set of REST endpoints that can be directly accessed as a HTTP URL to check the status.

Q : How does Spring enable creating production ready applications in quick time?

Spring Boot aims to enable production ready applications in quick time. Spring Boot provides a few non functional features out of the box like caching, logging, monitoring and embedded servers.

* spring-boot-starter-actuator - To use advanced features like monitoring & tracing to your application out of the box
* spring-boot-starter-undertow, spring-boot-starter-jetty, spring-boot-starter-tomcat - To pick your specific choice of Embedded Servlet Container
* spring-boot-starter-logging - For Logging using logback
* spring-boot-starter-cache - Enabling Spring Framework’s caching support

-How to restart tomcat automatically? when u modify any class or method in spring boot?

-spring boot actuator?

-how many ways we can create spring boot project?

-What is Jhipster what is the purpose it, did you ever used it?

-How you are deploying your micro services.

-Why JSON why not XML? where we can integrate both XML and JSON from UI

-What are the Repositories in spring boot and how we will write if I have sample Entity Price give a sample.

-If in spring boot application.properties is there then why we need application.yml

</SPRING-BOOT>

<REST>

-Difference between request param and path variable?

-Difference between idempotent vs non idempotent?

-what are the annotations? that are used in rest?

How do you expose your rest services? write snippet of code (she gave a scenario) I have a requirement based on product name it will give me price just one input and one op

What is difference between @pathVariable and @RequestParm when we should choose which one (she asked it based on my previous question code)

- want to stop some process before off my server how can I achieve this ?

[addShutdownHook](https://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html#addShutdownHook(java.lang.Thread))([Thread](https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.html) hook) check Runtime class Api

-What is the difference between Soap web services and rest web services , in which case we should go for which one

1. They gave small scenario to explain WSDL structure for overloaded method public Device getDeviceInfo(int deviceId);

Public Device getDeviceInfo (String deviceName);

</REST>

<CORE-JAVA>

OverRiding:

1)Providing new implementation to superclass method in subclass. Overriden method execution is resolved on object type. Whereas static method is resolved on referenced type as it doesn’t contain object.We can override only in subclass.we can not override final method.while overriding we don’t get ambiguous error.we must take help of super keyword if we want to access the super class method that is overridden in subclass. Provide runtime polymorphism.

OverLoading:

Creating new method with same method name but different parameters.Overloaded method execution is resolved based on reference type and argument type. we can overload the method in same class as well as in child class. Provide compile time polymorphism.

Is-A(Inheritence) We should establish Is-A relation between classes for grouping classes as one family to establish parent child relation. So that child class objects can be used wherever parent class reference variable is used. We can establish IS-A relation using implements and extends keyword.

HAS-A(Composition): We should establish Has-A relation between classes if one object can not exist without another object. This relation can be implemented by creating one class referenced variable as non static variable in another class.

USES-A(Aggregation): We should establish uses-A relation between classes if one object uses another objectfor performing some operation. To establish this relation we must create referenced variable of one class into method parameter of another class.

OOP Principles: Encapsulation,Inheritence,Polymorphism, Abstraction(supporting principle)

Encapsulation:The process of creating a class by hiding its internal data from direct access by outside classes and providing its access only through publicly exposed setters and getters methods. We develop encapsulation by declaring all its variables as private and to access them we write public setters and getters methods. By declaring variables as private we stop the direct access of this variable. Now through setters and getters we can access same private variable with proper validation. Like in a typical bank class example deposited money can not be negative.So while depositing the funds we can validate whether the data is negative or positive. So, using encapsulation we can achieve security to object data.

Inheritence:- The process of creating a new object by obtaining type,properties and behaviours of an existing object is called inheritance. Through inheritance we can reuse the existing object properties and behaviours with our new object reference as if they were defined in our class. The main benefit of inheritance is reusability and runtime polymorphism. Using, extends and implements we can implement inheritance.

Polymorphism:- the process of defining a method with multiple implementation exhibiting different behaviours of the object in same operation is called polymorphism. Polymorphism can be developed in two ways either by method-overloading or by method-overriding.

Abstraction: The process of defining a class by hiding implementation details of a method and providing only necessary details to invoke this method is called abstraction.Through abstraction we will first define abstract methods, later in subclasses we will implement their logic as per subclass requirement.

1. Where you used polymorphism
2. How is use of gc. Can we execute GC?

Garbage collection is algorithm to reclaim the unused memory in heap that are occupied by descarded objects. We can call gc using System.gc() or Runtime.getRuntime().gc(). Calling gc() doesn’t guarantee that garbage collection will happen, it just suggests the jvm that it’s good time for garbage collection; no, we can’t execute the gc even if we call System.gc() or Runtime.gc() garbage collection is not guaranteed. There is no difference between System.gc() and Runtime.gc() . System.gc()internally calls Runtime.gc(). The only difference is System.gc() is a class method whereas Runtime.gc()is an instance method. So, System.gc() is more convenient. Also, Runtime.gc() is a native method whereas System.gc() is non - native method which in turn calls the Runtime.gc()

1. How to propagate exception between multiple layer
2. Final classes in java:: System, All wrapper classes, string, string buffer,StringBuilder;
3. Static classes in java::
4. Hashcode of null is 0;
5. What is use of multi catch block

Prior to java 7 we need to write separate catch block for catching a particular exceptions.ex.

catch(IOException ie){

ie.printStackTrace();

}

Catch(SQLException se){

se.printStackTrace();

}

But from java 7 we can catch multiple exceptions in same catch block. Eg.

Catch(IOException | SQLException exc ){

Exc.printStackTrace();

}

1. When we have to use comparator

When we want the natural sorting order we go for comparable and for customized sorting we go for Comparator. Using comparator we can implement many sort sequence like if there’s a class Employee then we can sort it based on employee id, name, salary, joining date etc. but in case of comparator we can sort this object using any one of its field.

1. What is contract between hash code and equal method

1) Two objects can have same hashcode.

2) If the two objects are equal their hashcode must be same, but vice versa is not true.

So, whenever we override equals(), we must override hashcode.

1. How to improve performance of application
2. How stream api is working internally
3. What is use of functional interface
4. Difference between abstract class and interface

<http://www.thecoldsun.com/en/content/01-2009/abstract-classes-and-interfaces>

Autoboxing and UFFnboxing::

The process of converting the primitive data type to corresponding wrapper class Object is called Autoboxing (ie int to Integer, char to Character double to Double etc )and the process of converting Object type to corresponding primitive type is called unboxing.

Ex:

List<Integer> list =new ArrayList<>();

For(int i=0;i<10;i++){

list.add(i); //autoboxing

}

Serialization:::

The process of converting the object state into stream of bytes. Using serialization we can store the object state permanently in a destination e.g file or in remote computer. In order to serialize an object we need to implement our class from Serializable interface. Serializable is a marker interface that provides writeObject() special permission to serialize the object. If our class doesn’t implement serializable interface then writeObject(-) will throw NotSerializableException. Now using

ObjectOutputStream oos= new ObjectOutputStream(new FileOutputStream(new File(“file.ser”)));

oos.writeObject(obj); we can serialize the object.

The method writeObject store the object in a file with the given class name, serialVersionUid of that .class file all the non static non transient variables names and their data types, and their current modified value.

Deserialization is the process of converting stream of byte into original object.We can achieve deserialization using

ObjectInputStream ois=new ObjectInputStream(new FileInputStream(“file.ser”));

Object1 obj1=(Obj1)ois.readObject();

-During deserialization readObject() first first reads class name from the file and then using class.forName() it loads the original class. Here static variables are assigned with the original value and static blocks execute.

-now it compares the serialVersionUid of this class with the serialVersionUid stored in file.if both the serialVersionUid matches then it creates the object of current loaded class without using new keyword and populates the non transient variables values in the object. The static and transient variables are populated with their default value. But if the serialversionuid doesn’t matches it terminates deserialization by throwing java.io.InvalidClassException.

During deserialization the .class file must be available otherwise readObject() will throw ClassNotFoundException. Also we must use same version of class in deserialization that we have used during serialization.

In serialization the process of serialization is in hand of JVM. Also serialization is not fully secured if we want to serialize any sensitive data like card number or password. So, using Externalization we manually serialize the object. In externalization we implement our class from externalization and user writeExternal(ObjectOutput oo) for serialization also to deserialize the object we use readExternal(ObjectInput oi); inside writeExternal(ObjectOutput oo) and readExternal(ObjectInput oi) we use writeXXX() for serialization/deserialization.

public class User implements Externalizable {

@Override

public String toString() {

return "User [id=" + id + ", name=" + name + "]";

}

int id;

String name;

@Override

public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException {

this.id = in.readInt();

this.name = in.readUTF();

System.*out*.println("read........");

}

@Override

public void writeExternal(ObjectOutput out) throws IOException {

out.writeInt(2222);

out.writeUTF("xxxxx");

System.*out*.println("write.........");}}

…….. public class Test {

public static void main(String[] args) throws ClassNotFoundException, IOException {

User user=new User();

user.id=1;

user.name="r";

ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStream("/file.ser"));

oos.writeObject(user);

ObjectInputStream ois=new ObjectInputStream(new FileInputStream("/file.ser"));

User u=(User) ois.readObject();

System.*out*.println(u); }}

OP:: write.........

read........

User [id=2222, name=xxxxx]

1. How to find memory leakage in your application

[Java profilers](https://stackify.com/java-profilers-3-types/) as a good way to track down memory leaks and be able to run the garbage collector manually. You can use Java profilers to review how memory is being used that can easily show you which processes and classes are using too much memory when they should not be. You can also use JVM Performance Metrics, which gives you tons of data on garbage collection, thread counts, and memory usage.

We can use tools like JProfiler, **GC Viewer**

several ways to prevent memory leaks in Java, including:

* Release the session when it is no longer needed. Use the HttpSession.invalidate() to do this.
* Keep the time-out time low for each session.
* Store only the necessary data in your HttpSession.
* Avoid using string concatenation. Use StringBuffer’s append() method because the string is an unchangeable object while string concatenation creates a lot of unnecessary objects. A large number of temporary objects will slow down performance.
* As much as possible, you should not create HttpSession in your jsp page. You can do this by using the page directive <%@page session=”false”%>.
* If you are writing a query that is frequently executed, use PreparedStatement object rather than using Statement object. Why? PreparedStatement is precompiled while Statement is compiled every time your SQL statement is transmitted to the database.
* When using JDBC code, avoid using “\*” when you write your query. Try to use the corresponding column name instead.
* If you are going to use stmt = con.prepareStatement(sql query) within a loop, then be sure to close it inside that particular loop.
* Be sure to close the Statement and ResultSet when you need to reuse these.
* Close the ResultSet, Connection, PreparedStatement, and Statement in the finally block.

**A quick word about Java profilers.** Java profiling helps you monitor different JVM parameters including object creation, thread execution, method execution, and, yes, garbage collection.

There are times when you have ruled out memory leaks as the reason for your application’s slow down, so you can use Java profiling tools to get a closer view of how your application is utilizing memory and other resources. Instead of going over your code to find the problems, you can just use these tools to do just that. It saves you the effort and the hours needed to ensure that your code is up to par.

These tools give you a comprehensive set of statistics and other information that you can use to trace your coding mistakes. They can also help you find what are really causing performance slowdown, multi-threading problems, and memory leaks. In short, they give you a more stable and scalable application. And the best part – Java profiling tools can give you a fine-grained analysis of every problem and how to solve them.

If you use these tools early into your project, and you use them regularly – particularly when you use them in conjunction with other [Java performance tools](https://stackify.com/java-performance-tools-8-types-tools-need-know/) – you can create efficient, high-performing, fast, and stable applications. It can also help you know critical issues before you deploy your app.

Some of the metrics you can find out using Java profiling tools include:

* A method’s CPU time
* Memory utilization
* Information on method calls
* What objects are created
* What objects are removed from memory or garbage collected

https://dzone.com/articles/what-to-do-about-java-memory-leaks-tools-fixes-and

The programming language that follows the Object concept but doesn’t follow all the oop principles are called object based programming language like javascript. The programming language that supports concept of Object and follow OOP principles like encapsulation,inheritance, polymorphism and abstraction is called object oriented programming language like java, .net.

-what is tree map? why we go for tree map

Treemap implements NavigableMap and the output is sorted in natural sorting order. doesn’t store null as key.

-TreeSet sorts in natural order. It compares the object before storing. So if we try to put integer and string both in same TreeSet, it will throw classcastexception.

-when to use equals method? and hash code?

There are two types of equality Object equality and reference equality. Say we have two Poffers

Public class POffers{

private String offerId;

private String offerName;

private String expiryDate;

//setters and getters

}

POffers p1=new POffers();

p1.setOfferId("a");

p1.setOfferName("off1");;

p1.setExpiryDate("march1");

POffers p2=new POffers();

p2.setOfferId("a");

p2.setOfferName("off1");

p2.setExpiryDate("march1");

As we have taken same attributes in both the objects. Both the objects have same properties but if we try to store in HashSet or HashMap they will act as different object as equals and hashCode method is coming from object class. In order to make both the objects equal we need to override equals method.

Generally we use hashcode() and equals() so that object can be smoothly stored in HashSet or HashMap. There’s a contract between hashCode and equals method. If two objects are equal/same then their hashCode must be same. But if two object’s have the same hashcode it need not be same object. Generally we override equals() and hashCode so that it can be stored in HashSet or HashMap smoothly. Let’s say we have a class called POffers. Two POffers will be same if they have same OfferId and expiry Date. Let, in this class we override the equals method.

Public class POffers{

Private String offerId;

Private String offerName;

Private String expiryDate;

Public boolean equals(Object obj){

If(this.offerId.equals(obj.offerId) && this. expiryDate.equals(obj. expiryDate)){

Return true;}

else{

Return false;} }

As we know set or map uses hashcode method to store the object in bucket. And when there is hashing collision, it uses equals method to differentiate the object. So, in above example both objects will be stored in different bucket even after both objects are same as their hashcode is still coming from object class. That’s why we need to override equals as well as hashCode(). Take the below example.

Public class POffers{

Private String offerId;

Private String offerName;

Private String expiryDate;

Public boolean equals(POffers obj){

If(this.offerId.equals(obj.offerId) && this. expiryDate.equals(obj. expiryDate)){

Return true;}

else{

Return false;}

public int hashCode() {

return this.offerId.hashCode()-this.expiryDate.hashCode();

}

}

Now both the objects are same and can be stored in HashSet or HashMap. If we insert p1 and then p2 then both the objects will be considered as same.

-Difference between comparable and comparator?

Both comparable and comparator are used to compare two objects or if want to create the custome sorting. These classes are also required when we want to put our object in TreeSet or treeMap.

In my application we have used Comparable as well as comparater for sorting.

We have class POffers, MOffers and GOffers

Public class POffers{

Private String offerId;

Private String offerName;

Private String expDate;

//setters and getters

}

We can compare these classes using Comparable or Comparater. Let’s first take Comparable

Public class POffers implements Comparable<POffers>{

Private String offerId;

Private String offerName;

Private String expDate;

//setters and getters

Public int compareTo(POffers offers){

return this.offerId.hashCode()-offers.getOfferId().hashCode();}

By this we can achieve natural sorting order.But if we need the Custom sorting the we need to implement our classes from Comparator.

Public class POffersComparator implements Comparator<POffers>{

Public int compare(POffers offers1, POffers offers2){

return offers1.expiryDate.hashCode()-offers2.expiryDate.hashCode();

}

Or

Public class POffersNameComparator implements Comparator<POffers>{

Public int compare(POffers offers1, POffers offers2){

return offers1.name.hashCode()-offers2.name.hashCode();

}}

The above class can be stored in TreeMap as

Map<String,POffers>=new TreeMap(Comparator<POffers>);

Map<String,POffers>=new TreeMap<>(new POffersNameComparator());

Or TreeSet<> ts=new TreeSet(new POffersNameComparator());

So, if we want the natural sorting order we can go for comparable interface given in java.lang package and override compareTo(Object o) but if we want custom sorting or multiple sorting then we can go for comparator of interface given in java.util packae and override compare(Object o1, Object o2). If we use Comparator our class doesn’t has to implement the interface. But there will be a class for each sorting. In case of Comparable there will be less classes but the class will be implementing Comparable interface.

2nd if we have the source code of the class then only we can use comparable as our class has to implement from comparable interface, but if we don’t have the source code of the class then comparator is the only option.

-Array list internal implementation?

ArrayList default capacity is zero(java 8). Before java 8 the default capacity of ArrayList was 10. So, array list creates an array with size 10. If the list becomes full and a new object is added. Array regrows by adding some more index in it. Then copy all the elements into this new ArrayList object and then add the new element in it. Now the Arraylist refer to this new Object and the old array list becomes eligible for garbage collection.

-hash map vs concurrent hash map?

HashMap is not thread safe where as ConurrentHashMap is thread safe and given in java 5 as part of java.util.concutrent package. ConurrentHashMap locks the particular entry when it is modified but when multiple threads try to read the particular entry from ConcurrentHashMap it doesn’t lock the entry. In order to make HashMap as Thread safe we need to use Collections.synchronizedMap(<Hashmap object>) but we don’t need to do anything for making concurrent Hashmap as thread safe.

-Array list vs linked list?

ArraList stores the elements in array where as LinkedList stores element in nodes. LinkedList implement List and DQuue. Where each node stares the reference of it’s next and previous link. Internally LinkedList uses its private class Entry to store the objects inside the link

Private class Entry{

E element;

Entry next;

Entry Previous;

}

-initial capacity if array list? Before java 8 it was 10 but from java 8 it’s zero.

-project packing jar or war?

For web application it is packaged as war.

Myclass.war

|-META-INF

|-WEB-INF

| -classes

|-lib

|-web.xml

-how to run jar? Go to classpath>>> java –jar <jar\_name>.jar <class\_nm>.class

-difference between executable and non executable jar?

-how to extract that jar and what is the structure of jar after extracting?

Extracting jar>>> jar xf <jar\_nm>.jar

Structure>>>

-a.class

-b.class

:

:

META-INF

|-manifest.mf

-how to create jar? Go to classpath and type in cmd jar –cvf <class\_name>.jar

Executable jar>> jar –cvmf manifest.mf <jar\_nm>.jar class\_nm.class

Inside manifest.mf write Main-Class: <class\_name without.class>

Synchronization:

Class-Level Synchronization

What will be the behavior for the below combinations of Synchronization.

I have two static methods having synchronized blocks at class-level, if two threads are trying to access the methods simultaneously, what is the behavior?

Let's take two threads accessing two methods m1() and m2() (of same class ) having synchronized block in both the methods. Assume Thread1 and Thread2 is calling m1() and m2() respectively. Let's assume Thread1 entered to first method and got and locked the class as m1() contains class level synchronized block. Assume Thread1 got suspended after entering the sync block( we can use Thread.sleep(1000)). As thread1 got suspended thread2 will get a chance to execute m2(). Thread2 will definetely enter to m2() but will not be able to execute the method as m2() also contain class level lock. Thread2 will remain suspended unless Thread1 releases the lock. Once Thread1 released the lock, Thread2 can further start Execution

I have two static methods having synchronized blocks at class-level, if two threads are trying to access the methods simultaneously on different object, what is the behavior?

If one thread has acquired the lock, it will not allow other threads to execute the synchronized block/ method of other object as well.

I have two non-static methods, in this case can i apply class-level synchronized block? If two threads are accessing the methods what is the behavior?

Yes. We can apply class level synchronized block in non static method. If we do so, if one method acquires the lock unless it release the lock 2nd method will not get a chance to execute.

In below example two threads are trying to access two different methods having synchronized block. Unless 1 thread release the lock 2nd thread can not execute it’s method.

Object-Level Synchronization

I have two non-static methods, i applied synchronization at object level, what will happen if two threads tried accessing those two methods with two objects of the class.

Both the threads will be able to execute at the same time. There will be no blocking for another thread due to 1st thread.

I have two non-static methods, i applied synchronization at object level, what will happen if two threads tried accessing those two methods with one object of that class.

If one of the threads acquired the lock, it will not allow other threads to execute any other synchronized block/method unless it releases the lock.

What is immutable? What’s the use of it . Any use cases?

If the state of an object is not allowed to change then such an object is called immutable. All the enums and String objects are immutable. Consider a messaging server which needs the port address and urls,of other servers to send the message. As the port number and urls of other servers remain same. The port no, urls can be stored as immutable object in messaging servers.

-What are the main oop principles and where u used in your project and why

-Which java version u familiar just give a brief on it about the new features

-java7 features

Jax-ws was added, jaxp, jaxb and stax was enhanced.

Try with resource

Try(InputStream is=new FileInputStream(src); OutputStream os=new FileOutputStream(dest)){

//code }

Multiple exception catching

Try{//some code

}catch(FileNotFoundException | SQLException exc)

{//handeling code}

Diamond operators

The code List<String> l=new ArrayList<String>()

Became List<String>l=new ArrayList<>();

-java 8 features?

* forEach() method in Iterable interface.
* default and static methods in Interfaces.
* Functional Interfaces and Lambda Expressions.
* Java Stream API for Bulk Data Operations on Collections.
* Java Time API.
* Collection API improvements.
* Concurrency API improvements.
* Java IO improvements.

Java 9 Features:

Interface can have private methods

interface InterfaceWithPrivateMethods {

    private static String staticPrivate() {

        return "static private";

    }

    private String instancePrivate() {

        return "instance private";

    }

-Will java stream API will give better performance as compare with java 7 approach can you proof it? (She gave laptop configure with jdk 9)

-What are the method presents in Object class, why 3 wait () overloaded method what is the purpose of it.

Clone(), finalize(), getClass()

Equals(), hashCode(), toString();

wait(), wait(long timeout) wait(long timout, int nanos)

notify(), notifyAll(),

1. Are you aware of Concurrent API? what are the new DS introduced in concurrent API just give a brief and purpose .
2. Did you aware on Executor Framework if yes what is the difference between execute() and submit() method?

ExecuterService.execute(runnabe<T>) return type is void. It doesn’t return anything. Whereas ExecuterService.submit(runnabe<T>)/ ExecuterService.submit(callable<T>) returns the Future<T>.

Execute(-) takes only runnable as method parameter whereas submit takes both runnable well as callable as method param.

1. What is Type generic in Java? Did you used as part of your project if yes. Where and its purpose

Type generic is used for type safety. I have used Generics in Collection classes as well as in Future<POffersComparable> and in comparators.

Where you used Set and List in your project just tell me the exact scenario , after my explanation He asked suppose am fetching bunch of Employee records from Table and I want to store them in Set is this right approach or not just give point to point answer.

difference between throw and throws?

Customized exception program

Difference between coucuranthashmap and synchronized hashmap?

A Java Bean is a java class that should follow following conventions:

It should have a no-arg constructor. It should be Serializable. It should provide methods to set and get the values of the properties, known as getter and setter methods.

Java Bean Why use Java Bean?

|  |
| --- |
| According to Java white paper, it is a reusable software component. A bean encapsulates many objects into one object, so we can access this object from multiple places. Moreover, it provides the easy maintenance. |

* About project.
* What is method overloading ?

It is one of the concept of polymorphisim. It defines more than one method have same name but different argument . In method overloading method resolution takes care by the compiler based on reference type.

* If an exception occurred before coming to the try-catch, then what will happen and how to resolve?

Abnormal termination.

* Why String is immutable?

For security, efficiency.

To create immutable class in java, you have to do following steps.

1. Declare the class as final so it can’t be extended.
2. Make all fields private so that direct access is not allowed.
3. Don’t provide setter methods for variables
4. Make all mutable fields final so that it’s value can be assigned only once.
5. Initialize all the fields via a constructor performing deep copy.
6. Perform cloning of objects in the getter methods to return a copy rather than returning the actual object reference.
7. What is connection polling?

 a connection pool is a cache of database connections maintained so that the connections can be reused when future requests comes to the database . Connection pools are used to enhance the performance of executing commands on a database.

* What is finally?

Java finally block is always executed whether exception is handled or not. Java finally block follows try or catch block.

* How u will show the result in jsp , fetching data from database? program
* Difference between HashMap and HashTable?

-HashTable doesn,t store null(neither as key nor as value).Where as null can be stored in hashMap (both as key and value).

-HashTable:insertion order is not preserved, in HashMap also insertion order is not preserved.

-HashTable is synchronized where as HashMap is not.

* What will be the output if we access private method? Program
  + Compiler throws exception <method> has private access in <class>
* How to resolve the OutOfmemory problem.(heap memeory)
* Is it possible to create object in interface and why?
* Exception program .and what will be output?
* If there are 5 methods in a interface, Is child class implements 2 method from 5 ?and why? Program…

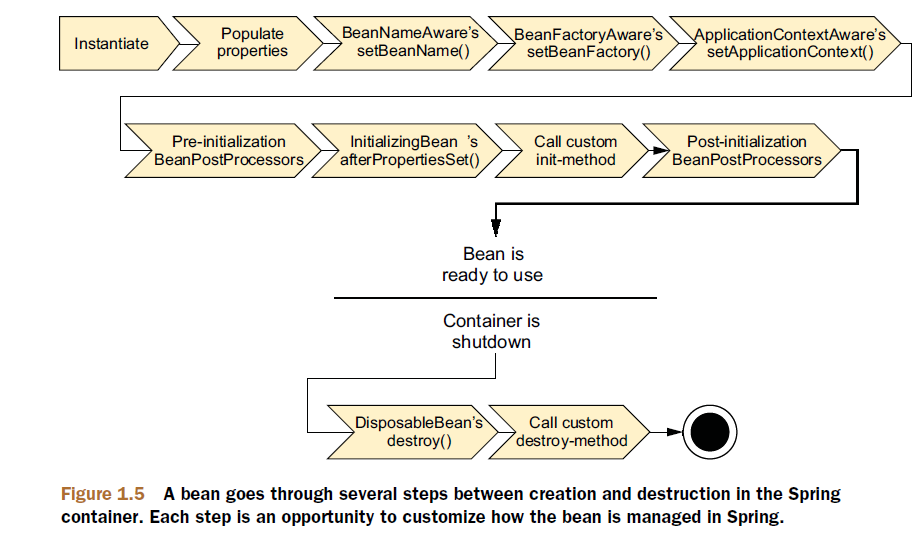
1. Abstraction vs interface
2. Linked list program(last element of list will come to first and after all are same)
3. Difference between sql date and util date?
4. Difference between sterilization and externalization?

* What is lambda Expression und uses?
* What is marker interface?
* Where u use marker interface on your project? Example ?
* What is oops and the benefit of oops comparing with other language?
* What is class?
* What is difference between collection and collections?
* What is hashing?
* Singletone design patern?

</CORE-JAVA>

<Spring>

-Bean life cycle



As you can see, a bean factory performs several setup steps before a bean is ready to

use. Let’s break down the figure in more detail:

1 Spring instantiates the bean.

2 Spring injects values and bean references into the bean’s properties.

3 If the bean implements BeanNameAware, Spring passes the bean’s ID to the setBeanName()

method.

4 If the bean implements BeanFactoryAware, Spring calls the setBeanFactory()

method, passing in the bean factory itself.

5 If the bean implements ApplicationContextAware, Spring calls the set-

ApplicationContext() method, passing in a reference to the enclosing application

context.

6 If the bean implements the BeanPostProcessor interface, Spring calls its post-

ProcessBeforeInitialization() method.

7 If the bean implements the InitializingBean interface, Spring calls its after-

PropertiesSet() method. Similarly, if the bean was declared with an initmethod,

then the specified initialization method is called.

8 If the bean implements BeanPostProcessor, Spring calls its postProcess-

AfterInitialization() method.

9 At this point, the bean is ready to be used by the application and remains in the

application context until the application context is destroyed.

10 If the bean implements the DisposableBean interface, Spring calls its

destroy() method. Likewise, if the bean was declared with a destroy-method,

the specified method is called.

Now you know how to create and load a Spring container. But an empty container

isn’t much good by itself; it doesn’t contain anything unless you put something in it.

To achieve the benefits of Spring DI, you must wire your application objects into the

Spring container. We’ll go into bean wiring in more detail in chapter 2.

First, let’s survey the modern Spring landscape to see what the Spring Framework

is made up of and what the latest versions of Spring have to offer.

-Spring Security?

-Spring with rest integration? they asked about rest controller.

-Spring cloud, pivotal cloud foundry

-where u r hosting your application ? Spring cloud or google cloud?

-Have you worked in Spring Security if yes which approach u used xml or configure if configure then why 3 overloaded method provided in SpringSecurityConfigureAdapter what is the purpose of each one (As I said Configure approach that’s why this question asked)

1. What are the all spring modules you work just give me comparison whether that think can’t you achieve in simple J2EE?
2. Give a Real world example of Spring Transaction which we all are using our daily life
3. What is jsp:useBean?

The jsp:useBean action tag is used to locate or instantiate a bean class. If bean object of the Bean class is already created, it doesn't create the bean depending on the scope. But if object of bean is not created, it instantiates the bean.

* Difference between @Service, @component, @Controller, @Repository.
* @Component | generic stereotype for any Spring-managed component |
* | @Repository| stereotype for persistence layer |
* | @Service | stereotype for service layer |
* | @Controller| stereotype for presentation layer (spring-mvc)
* Is it possible to use @Repository behalf of @Controller? explain.

Yes.but it will be wrong as per convention. @repository, @controller, @service, @component, all serves the same purpose. initializing that bean into container. But they have been given names as per business tiers to avoid the confusion. So that we can use choose the annotation names as per application layers.

1. “helloWord” program using spring, json Rest Api(coding).

What type of annotation use in hibernate inheritance?

1. Dao layer program of Spring mvc connection with Hibernate.
2. How to connect database using hibernate with spring mvc ?
3. How and where u make transcation management in spring mvc with hibernate?

</spring>

<WEB-SERVICE>

-web services security? Apache cxf wss4j interceptors?

-Dynamic method dispatch?

-Which part you worked Provider or consumer? Can you write WSDL?

-Are you aware of WSDL? if yes what are the components of WSDL and its purpose?

</WEB-SERVICE>

-What is hystrix circuit breaker design pattern?

What is JMS?

<PROJECT>

Project Description with end to end only the part which u worked

-Sort intro about you not project and technology all

</PROJECT>

<HIBERNATE>

-What is Data Modeling? He Gave one Scenario and tell me to approach data modeling for same

What is NoSQL? Why people are using it .is there any performance improvement?

What is Aerospike did u used it?

How can I configure Data source in such a manner that I it will not tightly coupled with my DB. assume today am using Oracle in future I want to move MySQL so that time it should not impact to my business.

* What is Session Factory and difference between Session factory and Session?
* Difference between merge () and update();

many annotation u uses in spring?

-What is invert tag in hibernate?

* What is configuration in hibernate? program
* If I want to connect two database in hibernate , then how I will connect?
* What is dialect in hibernate ?

-How to insert query to the data source?

1. Difference between save and merge method in hibernate?

-Difference between first and second level cache?

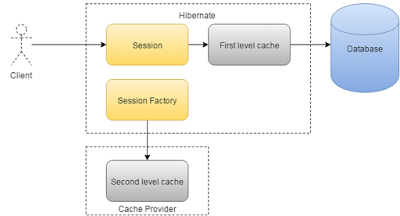
If you have used Hibernate in past then you know that one of the strongest points of Hibernate framework is caching, which can drastically improve the performance of Java application's persistence layer if configured and used correctly. Hibernate provides caching at many levels e.g. first level cache at Session level, second level cache at the SessionFactory level, and query cache to cache frequently executed [SQL queries](http://javarevisited.blogspot.sg/2017/02/top-6-sql-query-interview-questions-and-answers.html). The first level cache minimizes database access for the same object. For example, if you call the get() method to access Employee object with id = 1 from one session, it will go the database and load the object into memory, but it will also cache the object in the first level cache.  
  
When you will call the get() method again for the same object from the same session, even after doing some updates on the object, it will return the object from the cache without accessing the database. You can confirm this from Hibernate's log file by observing how many queries are executed. This is also one of the frequently asked [Hibernate Interview Questions](http://www.java67.com/2016/02/top-20-hibernate-interview-questions.html), so it will not only help to improve the performance of your Java application but also help you to do well on your next interview.  
  
This session level cache greatly improves the performance of Java application by minimizing database roundtrips and executing less number of queries. For example, if an object is modified several times within the same transaction, then Hibernate will only generate one SQL UPDATE statement at the end of the transaction, containing all the modification.

But, since this cache is associated with the Session object, which is a short-lived object in Hibernate, as soon as you close the session, all the information held in the cache is lost. So, if you try to load the same object using the get() method, Hibernate will go to the database again and fetch the record.

The second level cache is maintained at the SessionFactory level, which is used to open sessions, hence every session is linked to SessionFactory. This cache is opposite to first level cache which is by default enabled in Hibernate, this one is by default disabled and you need to configure the second level cache in Hibernate configuration file to enable it.  
The second level cache is provided with the help of caching providers e.g. EhCache and OSCache. If you look at the cache package in Hibernate, you can see the implementation of Caching related interfaces by these providers. Depending upon which cache you want to use, you can configure them in the Hibernate Configuration file.  
  
Once configured, every request for an object will go to the second level cache if it is not found in the [first level cache](http://www.java67.com/2017/02/2-best-books-to-learn-hibernate-for-Java-Developers.html). It won't hit the database without consulting second level cache, which means improved performance.  
  
It's very important for a Java and Hibernate developer to know about Caching in Hibernate. It's not just important from Interview point of view but also from the application development and performance improvement point of view. You will often face performance related challenges in a real world application which contain millions of records, by correctly configuring Hibernate sessions and writing code which make use of caching, your Java application can float above water even in the case of a significant load  
Now that we know what is first level and second level cache in Hibernate, let's revise some key differences between them from interview point of view.  
  
Scope  
First level cache is associated with Session Object, while the Second level cache is associated with the SessionFactory object. This means first level cache's scope is limited to session level while second level cache's scope is at the application level. Since Session object is created on demand from the SessionFactory and it's destroyed once the session is closed, the same query if run from two different sessions will hit the database twice if the second level cache is not configured. On the other hand, second level cache remains available throughout the application's life-cycle.  
  
  
Configuration  
First level cache is by default enabled in Hibernate, while the second level cache is optional. If you need it then you need to explicitly enable the second level cache on Hibernate configuration file i.e. the hibernate.cfg.xml file.  
  
You can use the hibernate.cache.provider\_class and hibernate.cache.use\_second\_level\_cache properties to enable the second level cache in Hibernate. The first one is the name of the class which implements Second level cache and could be different, depending upon which cache you use e.g. EhCache or OSCache.  
  
By default,  hibernate.cache.provider\_class is set to org.hibernate.cache.NoCacheProvider class, which means the second level cache is disabled. You can enable it by setting something like org.hibernate.cache.EhCacheProvider if you want to use EhCache as the second level cache.  
  
Here is a sample configuration to configure Second level cache with EhCache:  
  
Availability  
First level cache is available only until the session is open, once the session is closed, the first level cache is destroyed. On the other hand, second level cache is available through the application's life-cycle, it is only destroyed and recreated when you restart your application.  
  
  
Order  
If an entity or object is loaded by calling the [get()](http://javarevisited.blogspot.sg/2012/07/hibernate-get-and-load-difference-interview-question.html) method then Hibernate first checked the first level cache, if it doesn't found the object then it goes to the second level cache if configured. If the object is not found then it finally goes to the database and returns the object, if there is no corresponding row in the table then it return [null](http://javarevisited.blogspot.sg/2014/12/9-things-about-null-in-java.html). When an object is loaded from the database is put on both second level and first level cache, so that other session who request the same object can now get it from the second level cache.  
  
In case if the object is not found in the first level cache but found in the second level cache because some other sessions have loaded the object before then it is not only returned from first level cache but also cached at first level cache, so that next time if your code request the same object, it should be returned from 1st level cache rather than going to the 2nd level cache.

<prop key="hibernate.cache.use\_second\_level\_cache">true</prop>

<prop key="hibernate.cache.provider\_class">org.hibernate.cache.EhCacheProvider</prop>

Don't forget to include hibernate-ehcache.jar into your classpath. This class comes from that JAR. You can also see [Java Persistence with Hibernate](https://www.amazon.com/Java-Persistence-Hibernate-Christian-Bauer/dp/1617290459?tag=javamysqlanta-20), 2nd edition to learn more about other configuration options available to second level cache.  
  
In general. When an object is pass to [save()](http://www.java67.com/2016/01/difference-between-save-saveorupdate-and-persist-in-Hibernate.html), update(), or [saveOrUpdate()](http://javarevisited.blogspot.sg/2012/09/difference-hibernate-save-vs-persist-and-saveOrUpdate.html) method and retrieved by load(), get(), list(), iterate(), or scroll() method, that object is added to the internal cache of the Session and when the flush() is subsequently called, the state of the object is sychronized with the database.  
  
Second level cache can also be configured on a per-class and per-collection basis, which means it can cache a class or a collection. You can use class-cache and colleection-cache elements in hibernate.cfg.xml to specify which class or collection to cache at 2nd level cache. You should remember that second level cache by default doesn't cache any entitty until you configure it.  
  
You can also use JPA Annoation @Cacheable to specify which [entity](http://javarevisited.blogspot.sg/2016/01/why-jpa-entity-or-hibernate-persistence-should-not-be-final-in-java.html) is cacheable. and Hibernate annoation @Cache to specify caching startegy e.g. CacheConcurrencyStrategies like READ\_WRITE or READ\_ONLY to tell Hibernate how the second level cache should behave.  
[](http://www.shareasale.com/m-pr.cfm?merchantID=53701&userID=880419&productID=546412722)

What is the use of cascade and inverse in Hibernate?

</HIBERNATE>

1. Why spring boot why not spring .give point to point answer
2. Assume I have an interface ECurrency with getCurrencyInfo (Locale locale) and I have multiple concert implementation I have like INCurrencyFactory, USCurrencyFactory like this I injected both in my services .but I want to inject both as a single instance how can I approach to this use case.

<servlet>

JSP implicit Objects::



**Out**: This is used for writing content to the client (browser).out.print(); out.println();

**Request**: The main purpose of request implicit object is to get the data on a JSP page which has been entered by user on the previous JSP page. While dealing with login and signup forms in JSP we often prompts user to fill in those details, this object is then used to get those entered details on an another JSP page (action page) for validation and other purposes. Request.getParameter(“userId”); request.getParameter(“password”);

**Response**: It is basically used for modfying or delaing with the response which is being sent to the client(browser) after processing the request.response.setContentType(“-”),response.sendRedirect(“-”);

**Session:** It is most frequently used implicit object, which is used for storing the user’s data to make it available on other JSP pages till the user session is active.

**Application:** This is used for getting application-wide initialization parameters and to maintain useful data across whole JSP application. Application.getAttribute(“-”), application.setAttribute(“-”).

**Exception:** Exception implicit object is used in exception handling for displaying the error messages. This object is only available to the JSP pages, which has isErrorPage set to true.

**Page:** Page implicit object is a reference to the current Servlet instance (Converted Servlet, generated during translation phase from a JSP page). We can simply use this in place of it. I’m not covering it in detail as it is rarely used and not a useful implicit object while building a JSP application.

**pageContext**: It is used for accessing page, request, application and session attributes.

**Config:** This is a Servlet configuration object and mainly used for accessing getting configuration information such as servlet context, servlet name, configuration parameters etc.

</servlet>

1. What is the different between ServletConfig and ServletContext?

Ans=

* ServletConfig available in javax.servlet.\*; package
* ServletConfig object is one per servlet class.
* Object of ServletConfig will be created during initialization process of the servlet
* This Config object is public to a particular servlet only
* Scope: As long as a servlet is executing, ServletConfig object will be available, it will be destroyed once the servlet execution is completed.
* We should give request explicitly, in order to create ServletConfig object for the first time
* In web.xml – <init-param> tag will be appear under <servlet-class> tag

Here's how it looks under web.xml : ([Source](http://javapapers.com/servlet/difference-between-servletconfig-and-servletcontext/))

<servlet>

<servlet-name>ServletConfigTest</servlet-name>

<servlet-class>com.stackoverflow.ServletConfigTest</servlet-class>

<init-param>

<param-name>topic</param-name>

<param-value>Difference between ServletConfig and ServletContext</param-value>

</init-param>

</servlet>

* ServletContext available in javax.servlet.\*; package
* ServletContext object is global to entire web application
* Object of ServletContext will be created at the time of web application deployment
* Scope: As long as web application is executing, ServletContext object will be available, and it will be destroyed once the application is removed from the server.
* ServletContext object will be available even before giving the first request In web.xml .if we configure load on start up – <context-param> tag will be appear under <web-app> tag

Here's how it looks under web.xml :

<context-param>

<param-name>globalVariable</param-name>

<param-value>com.stackoverflow</param-value>

</context-param>

1. What is jsp include directive?

The include directive is used to include the contents of any resource it may be jsp file, html file or text file. The include directive includes the original content of the included resource at page translation time (the jsp page is translated only once so it will be better to include static resource).For Code Reusability purpose. <%@ include file="resourceName" %>

1. What is jsptaglib directive?

The JSP taglib directive is used to define a tag library that defines many tags. We use the TLD (Tag Library Descriptor) file to define the tags. In the custom tag section we will use this tag so it will be better to learn it in custom tag. <%@ taglib uri="uriofthetaglibrary" prefix="prefixoftaglibrary" %>

* How to find the 3rd largest salary from employee?
* join of two tables
* After joining two salary table and employee table getting 4th highest paid salary of employee
* How to change the port no. of server in eclipse.
* WAP occurrence of character in String.
* WAP to Swap two variables without using 3rd variable.
* WAP to for circular linked list.
* WAP for producer consumer relation.
* WAP for String Anagram.

1. Matrix anti clock wise 90 degree program in java
2. Different between application object and session object?
3. WAP I array to retrieve largest 3 no. from array.
4. Spring mvc with rest web service insert, update program
5. What report u use on your project?
6. In which format u use jasper report?
7. Wap occurrences of character in string.

AricentTechnolopgy:

*1st Round: Test:*

Written and progrominc test

*2st Round: Test:*

* Can we write query in hibernate?
* What are type of mvc model?
* What are the folder structure of spring mvc?
* There is table , query for finding all null value of specific coloumn and null of all table?
* Find index of first non-repeated character in a string.
* In an array there are lot of floating –ve and +ve value, find the sum of value which is equivalent to zero.

Without using any inbuilt function and collection.

*<<<<*

Accenture Interview Question:

1. Where you used polymorphism
2. How is use of gc
3. Can we execute GC
4. How to propagate exception between multiple layer
5. What is use of multi catch block
6. When we have to use comparator
7. What is contract between hash code and equal method
8. How to improve performance of application
9. How stream api is working internally
10. What is use of functional interface.
11. What is order if creation object (listener, filter, Servlet context, Servlet config)
12. How to pass additional data between multiple web pages
13. How to invalidate session.
14. How to configure error page in Servlet
15. Different between L1 and L2 cache
16. What is DI and IOC
17. Spring bean life cycle
18. Spring MVC flow

Infogain Interview Question:

1. Write the query to find duplicate value in related column.
2. Write the query to give grant for particular table
3. What is difference between primary key and unique key
4. What is DML and DDL
5. What is use of sequence
6. What is Rollback policy?
7. Difference between truncate and delete and drop
8. How to validate form using javascript
9. Write a program based on selection one should be enables and one disabled
10. How to read form value using javascript.
11. Write the program to disable all filed
12. How to send disable form field value to next page
13. Write a program to retrieve a data from controller and display in jsp page
14. What is safe method is rest api
15. Difference between http1 and http2
16. How to implement cache in Rest api.
17. What is use ETAG is rest
18. Difference between BASIC and DIGEST security.
19. How you will provide method level security in rest.
20. How to handle transaction between multiple database
21. Give typical architecture of your project
22. Which SDLC model you are following
23. How you are getting requirement
24. How you can make your rest api as thread safe

Capgemini Interview Question:

1. Online amcat written test( Java, Spring, Hibernate technical question)
2. Write a immutable class without using final keyword
3. Write a program to explain all oops concept
4. What will happen if we store custom object in tree set or tree map.
5. Why string class is immutable not string buffer.
6. Difference b/w IO stream NIO Stream
7. What is disadvantage of IO stream
8. What is service locator design pattern
9. What are design pattern you have used in your project
10. Difference between Factory design pattern and abstract Factory pattern
11. What is proxy design pattern
12. What all are the pattern used by spring
13. What is pull mechanism and push mechanism
14. What is disadvantage of hibernate
15. When we have native sql in hibernate why we have to use jdbc
16. How to call procedure in hibernate
17. How to resolve n+1 problem in hibernate.
18. What is asynchronous and synchronous communication?
19. Difference b/w absolute path and relative path
20. What if order of logging level
21. How to configure connection pool in server. How you get object of that
22. Difference between flat transaction and distributed transaction.
23. What is trace method in http.
24. How to raise a event in your application.
25. When are working multiple environment so how your managing DB credentials in your application.
26. How to resolve conflict in git.
27. How you are making your war file how you are deploying.
28. What is your rollback policy if deployment got failed?
29. What all are your step to resolve production issue.
30. How you are managing you team.
31. Have you involved in design.

SoftVision Interview questions:

1. How to read session attribute in spring mvc
2. What is difference between request param and path param
3. What is use of @ModelAttribute annotation
4. What is use of @Initbinder annotation
5. Diffrencce between @Rest Controller and @ Controller
6. Difference b/w @Component and @Bean
7. Explain all view resolver and controller in spring
8. What is difference between xml configuration and annotation based configuration in spring
9. What is difference between request and session scope
10. If depended been have prototype scope if we call getbean method what will happen.
11. What is aware injection
12. What is dependency pool and dependency lookup
13. What is bean post processor
14. What is namaparamer jdbc template and jdbc template.
15. Jdbc template is thred safe or not
16. how to integrate spring with ORM tool write the program
17. how to configure multiple property file in spring.
18. how to execute batch in spring jdbc
19. difference between rowmapper and resultsetextraror
20. what is difference between ref and idref tag
21. How to handle exception in spring mvc
22. Difference between put and post method
23. What is resource class and resource method in rest
24. What is use of @Context annotation in rest
25. What is difference between url and Uri.
26. What is difference between itrate and execute update method in hibernate
27. What is QBC and criterion object in hibernate
28. What is order class and restriction class in hibernate.
29. What happen if we call get uniqueResult() method in hibernate if query giving multiple result.
30. Difference between merge and update method in hibernate.
31. What of component mapping in hibernate.
32. How to configure composite key In hibernate.
33. What is group id and artifact id in maven
34. Maven life cycle. And maven project architecture
35. Diffrence between mvn clean and maven install
36. How to resolve conflict in maven
37. How to configure your test case in maven
38. What is difference between after class and before class in junit
39. How to test void method in junit.
40. How to test you rest api using junit

*>>>>>*

Altimetric:

*1st Round: System Test:*

* 40 objective from java, c++, data structure , java script, sql
* 2 programing:
* One scenario where there are lot of parents having lot of children. Have to find the siblings of particular parent .

Example: input1 =5; length of the araay.

int[] input2={1, 3, 5, 6,7}

input3= siblings of particular no.

siblings will consider as n+1 and n+2

here siblings of 5 are 6,7

* There are 20 students in a class. Each student having different points for which they can able stand in a queue according to their points from highest point to lowest point. If any student having less point who stand ahead of student who has more point, how to track this things.

Int[] input1, input2

Where I<j and P[i]>p[j] then …

*2nd round:*

Find index of first non-repeated character1.Given a List of N number a1, a2, a3........an, you have to find smallest number from the List that is repeated in the List exactly K number of times.

Input Format

First Line of Input Contain Single Value N, Size of List

Second Line of Input Contain N Space Separated Integers

Third Line of Input Contain Single Value K

Output Format

Smallest Integer Value That is Repeated Exactly K Number of Time

Constraints

0 < N < 100001

0 <= K < 100001

0 <= ai< 100001

NOTE

There Will Be At least One Variable Which Is Repeated K Times

2. Class A has to keep generating random numbers and it has to notify class B when it generates a composite number. Class B has to print the number whenever it receives a notification. (Provide code for same)

3. The program has to search for a word in a data structure of multimillion entries what libraries/mechanism within java you recommend to minimize the time of search

4.Given a List of Distinct N number a1, a2, a3........an.

Find The Position Of Number K In The Given List.

Input Format

First Line Take Input Value of N

Second Line Take Input N Space Separated Integer Value

Third Line Take Input Value of K

Output Format

Position of K in the Given List

Constraints

0 < N < 100001

0 <ai< 1010

0 < K < 1010

NOTE:

Array Indexing Starts From 0

SAMPLE INPUT

5

1 2 3 4 5

4

SAMPLE OUTPUT

3

Time Limit: 1.0 sec(s) for each input file.

5. Complete the merge String function .It has 2 parameters:

1. A String a.

2. A String b.

Your function must merge string a and b and then return a single merged string. A merge operation between 2 string describe below

* Append alternating character from a and b , respectively, to some new String, merged string
* Once all of the characters in one of the strings have been merged, append the remaining characters in the other string to merged string
* Use only one single iteration to merge string

Sample input: s1=ab s2=zsd output: azbsd

in a string.

Accenture Interview Question:

1. Where you used polymorphism
2. How is use of gc
3. Can we execute GC
4. How to propagate exception between multiple layer
5. What is use of multi catch block
6. When we have to use comparator
7. What is contract between hash code and equal method
8. How to improve performance of application
9. How stream api is working internally
10. What is use of functional interface
11. What is order if creation object (listener, filter, Servlet context, Servlet config)
12. How to pass additional data between multiple web pages
13. How to invalidate session.
14. How to configure error page in Servlet
15. Different between L1 and L2 cache
16. What is DI and IOC
17. Spring bean life cycle
18. Spring MVC flow

Infogain Interview Question:

1. Write the query to find duplicate value in related column.
2. Write the query to give grant for particular table
3. What is difference between primary key and unique key
4. What is DML and DDL
5. What is use of sequence
6. What is Rollback policy?
7. Difference between truncate and delete and drop
8. How to validate form using javascript
9. Write a program based on selection one should be enables and one disabled
10. How to read form value using javascript.
11. Write the program to disable all filed
12. How to send disable form field value to next page
13. Write a program to retrieve a data from controller and display in jsp page
14. What is safe method is rest api
15. Difference between http1 and http2
16. How to implement cache in Rest api.
17. What is use ETAG is rest
18. Difference between BASIC and DIGEST security.
19. How you will provide method level security in rest.
20. How to handle transaction between multiple database
21. Give typical architecture of your project
22. Which SDLC model you are following
23. How you are getting requirement
24. How you can make your rest api as thread safe

Capgemini Interview Question:

1. Online amcat written test( Java, Spring, Hibernate technical question)
2. Write a immutable class without using final keyword
3. Write a program to explain all oops concept
4. What will happen if we store custom object in tree set or tree map.
5. Why string class is immutable not string buffer.
6. Difference b/w IO stream NIO Stream
7. What is disadvantage of IO stream
8. What is service locator design pattern
9. What are design pattern you have used in your project
10. Difference between Factory design pattern and abstract Factory pattern
11. What is proxy design pattern
12. What all are the pattern used by spring
13. What is pull mechanism and push mechanism
14. What is disadvantage of hibernate
15. When we have native sql in hibernate why we have to use jdbc
16. How to call procedure in hibernate
17. How to resolve n+1 problem in hibernate.
18. What is asynchronous and synchronous communication?
19. Difference b/w absolute path and relative path
20. What if order of logging level
21. How to configure connection pool in server. How you get object of that
22. Difference between flat transaction and distributed transaction.
23. What is trace method in http.
24. How to raise a event in your application.
25. When are working multiple environment so how your managing DB credentials in your application.
26. How to resolve conflict in git.
27. How you are making your war file how you are deploying.
28. What is your rollback policy if deployment got failed?
29. What all are your step to resolve production issue.
30. How you are managing you team.
31. Have you involved in design.

SoftVision Interview questions:

1. How to read session attribute in spring mvc
2. What is difference between request param and path param
3. What is use of @ModelAttribute annotation
4. What is use of @Initbinder annotation
5. Diffrencce between @Rest Controller and @ Controller
6. Difference b/w @Component and @Bean
7. Explain all view resolver and controller in spring
8. What is difference between xml configuration and annotation based configuration in spring
9. What is difference between request and session scope
10. If depended been have prototype scope if we call getbean method what will happen.
11. What is aware injection
12. What is dependency pool and dependency lookup
13. What is bean post processor
14. Spring bean life cycle method.
15. What is namaparamer jdbc template and jdbc template.
16. Jdbc template is thred safe or not
17. how to integrate spring with ORM tool write the program
18. how to configure multiple property file in spring.
19. how to execute batch in spring jdbc
20. difference between rowmapper and resultsetextraror
21. what is difference between ref and idref tag
22. How to handle exception in spring mvc
23. Difference between put and post method
24. What is resource class and resource method in rest
25. What is use of @Context annotation in rest
26. What is difference between url and Uri.
27. What is difference between itrate and execute update method in hibernate
28. What is QBC and criterion object in hibernate
29. What is order class and restriction class in hibernate.
30. What happen if we call get uniqueResult() method in hibernate if query giving multiple result.
31. Difference between merge and update method in hibernate.
32. What of component mapping in hibernate.
33. How to configure composite key In hibernate.
34. What is group id and artifact id in maven
35. Maven life cycle. And maven project architecture
36. Diffrence between mvn clean and maven install
37. How to resolve conflict in maven
38. How to configure your test case in maven
39. What is difference between after class and before class in junit
40. How to test void method in junit.
41. How to test you rest api using junit