JAVA

Screening & Training [Interview Questions]

17 SEPTIEMBRE

NUBELITY LLC
Engineering Department

NUBELITY
IT Talent and Solutions

NUBELITY
IT Talent and Solutions

Document version

Ver.	Version Owner	Ver. Date	Reviewed by	Rev. Date	Approved by	Approval Date
1.1	Victor Ochoa	01/10/2023	Valentin Gonzalez	03/12/2022	Julio Negrete	03/12/2022
1.2	Victor Ochoa	02/09/2023	Valentin Gonzalez	03/13/2022	Julio Negrete	03/13/2022
1.3	Victor Ochoa	03/09/2023	Valentin Gonzalez	03/13/2022	Julio Negrete	03/13/2022
2.1	Valentin Gonzalez	04/06/2023	Victor Ochoa	03/10/2022	Julio Negrete	03/10/2022
2.2	Valentin Gonzalez	05/04/2023	Victor Ochoa	03/08/2022	Julio Negrete	03/08/2022
3.1	Julio Negrete	06/02/2023	Valentín Gonzalez	06/06/2023	Julio Negrete	06/06/2023
4.1	Marín Alvarado	12/19/2023	Valentín Gonzalez	12/20/2023	Julio Negrete	12/20/2023
5.1	Julio Negrete	03/19/2024	Israel Albino	03/19/2024	Julio Negrete	03/19/2024
6.0	Valentin Gonzalez	04/02/2024	Julio Negrete	04/02/2024	Julio Negrete	04/02/2024

NUBELITY
IT Talent and Solutions

Contents

Technical Screening Questions

Basic questions approach

1. **What are the main concepts of OOP? Explain each of them.**

Answer:

Object-oriented programming (OOP) is a programming paradigm based on the concept of objects, which can contain data and code. The basic concepts of OOP are:

1. ***Inheritance*:** Inheritance is the process where one class acquires the properties (methods and fields) of another class 2. With the use of inheritance, the information is made manageable in a hierarchical order.
2. ***Polymorphism*:** Polymorphism is the ability of an object to perform different actions (or, exhibit different behaviors) based on the context.
3. ***Abstraction*:** Abstraction is a process of hiding the implementation details from the user, only the functionality will be provided to the user. In other words, the user will have the information on what the object does instead of how it does it.
4. ***Encapsulation*:** Encapsulation is a mechanism for wrapping the data (variables) and code acting on the data (methods) together as a single unit. In encapsulation, a class's variables will be hidden from other classes and can be accessed only through their current class methods. Therefore, it is also known as data hiding.

2. **What is lambda expression?**

Answer: A lambda expression is a short block of code that can be used to create anonymous functions in programming languages such as Java, C#, and Python 12. A lambda expression can be used to define a function without a name, which can be passed as an argument to other functions or stored in a variable.

Lambda expressions are often used in functional programming to simplify the syntax of functions and make them more concise. They are particularly useful for working with collections and streams of data, where they can be used to filter, map, and reduce data in a more expressive way.

In this example, the lambda expression (a, b) -> a + b defines a function that takes two arguments a and b, and returns their sum.

3. **What is the difference between an abstract class and an Interface?**

Answer:

Abstract Class: An abstract class can have both abstract (methods without a body) and concrete methods (methods with an implementation). It is used when classes share a common structure or

behavior but also have their own unique implementations of some methods. An abstract class can have constructors and can also have fields that are non-final and non-static. A class can extend only one abstract class, which means Java supports single inheritance for classes.

Interface: An interface, on the other hand, is a contract for what a class can do, without saying anything about how the class will do it. Before Java 8, interfaces could only have abstract methods (no method body). From Java 8 onwards, interfaces can also have default and static methods with a body. Interfaces are used to achieve full abstraction and multiple inheritance in Java because a class can implement multiple interfaces.

Key Differences:

- **Inheritance vs Implementation:** A class extends an abstract class using the extends keyword, implying that it is an is-a relationship. A class implements an interface using the implements keyword, which means it agrees to perform certain behaviors (has-a relationship).
- **Constructor and State:** Abstract classes can have constructors and can hold state (instance variables), while interfaces cannot have constructors and cannot hold state (until Java 8, interfaces could not have static, final, or non-final variables).
- **Access Modifiers:** All methods in an interface are implicitly public, while methods in an abstract class can have any access modifier.
- **Multiple Inheritance:** Interfaces enable a form of multiple inheritance because a class can implement multiple interfaces. A class, however, can extend only one abstract class.
- **Use Cases:** Abstract classes are used when subclasses share method implementations. Interfaces are used to define a common protocol for classes that can belong to different class hierarchies.

4. **What are the main HTTP verbs and how can they be implemented in Spring boot?**

Answer:

The main HTTP verbs are GET, POST, PUT, DELETE, HEAD, OPTIONS, and PATCH. These verbs represent the different types of operations that can be performed on a resource.

In Spring Boot, these verbs can be implemented using the @RequestMapping annotation. For example, to implement a GET request.

5. **What is Spring Initializer?**

Answer:

Spring Initializr is a tool that helps you create Spring Boot applications from scratch or from existing Gradle or Maven projects. You can choose from various options and features, such as Spring Web, Webflux, GraphQL, Docker, Modulith, and more.

To use Spring Initializr, you can visit the official website and select the options you want for your project. You can choose the language, build system, packaging format, and dependencies you need for your project. Once you have selected your options, you can download a ZIP file containing your project skeleton, which you can then import into your favorite IDE.

6. What is CRUD?

Answer:

CRUD stands for Create, Read, Update, and Delete, which are the four basic operations that can be performed on data in a database or a data storage system. These operations are foundational for most web and software applications, as they allow the system to manage and manipulate stored data effectively.

7. Explain the @RestController annotation?

Answer:

The @RestController is a Spring annotation used to create RESTful web services. It is a convenient annotation that combines the @Controller and @ResponseBody annotations, which eliminates the need to annotate every request handling method of the controller class with the @ResponseBody annotation.

The @Controller annotation is used to define a class as a Spring MVC controller. We typically use @Controller in combination with a @RequestMapping annotation for request handling methods. On the other hand, the @ResponseBody annotation is used to bind the returned object to the HTTP response body.

In contrast, the @RestController annotation is a specialized version of the @Controller annotation that includes the @Controller and @ResponseBody annotations. As a result, it simplifies the controller implementation by automatically serializing return objects into the HTTP response.

8. What are common design patterns used on spring boot applications?

Answer:

1. **Singleton:** Ensures a class has only one instance and provides a global point of access to it.

Spring beans are singleton by default.

2. **Factory Method:** Defines an interface for creating an object but let's subclasses alter the type of objects that will be created. Spring uses this for bean creation.
3. **Proxy:** Provides a surrogate or placeholder for another object to control access to it. Used extensively in Spring for AOP, transactions, and remoting.
4. **Observer:** An object, known as a subject, maintains a list of its dependents, called observers, and notifies them of any state changes. Spring events and listeners follow this pattern.
5. **Template Method:** Defines the skeleton of an algorithm in a method, deferring some steps to subclasses. Spring uses it in JdbcTemplate, HibernateTemplate, etc.
6. **Decorator:** Adds new functionality to an object without altering its structure. In Spring, this can be seen using multiple decorators on data sources or streams.
7. **Adapter:** Allows incompatible interfaces to work together. Spring's JdbcTemplate is an example, adapting the JDBC interface to the Spring framework.
8. **Builder:** Allows the creation of complex objects step by step. Used in Spring Security for configuring authentication and authorization.

9. **What is a stored procedure?**

Answer:

A stored procedure is a set of SQL statements with an assigned name, stored in the database catalog. It can perform complex operations, is executed by the database server, and can be called from applications. Stored procedures improve performance by reducing network traffic and enhance security by abstracting data access. They support parameters for dynamic execution and can include programming constructs such as loops and conditions, facilitating modular and reusable code.

10. **What are the two types of Exceptions in Java? Which are the differences between them?**

Java has two types of exceptions: checked exceptions and unchecked exceptions.

1. **Unchecked exceptions **do not need to be declared in a method or a constructor's throws clause, if they can be thrown by the execution of the method or the constructor, and propagate outside the method or constructor boundary.
2. On the other hand, checked exceptions must be declared in a method or a constructor's throws clause.

11. ¿Para qué sirve la palabra reservada static en Java?

a) Para declarar variables que no puedan cambiar de valor
b) Para declarar métodos/variables que deban ser referenciados a través de una instancia
c) Para declarar métodos/variables de clase
d) Para crear bloques estáticos de código
e) Para declarar constantes

12. ¿Cuál de las siguientes aserciones aplica para polimorfismo?

a. ***Un objeto puede comportarse diferente en situaciones diferentes***
b. Un objeto hereda de dos clases
c. ***Se puede hacer por herencia***
d. ***Se puede hacer por implementación de interfaces***
e. Se puede hacer por sobreescritura, pero no por sobrecarga de métodos
f. Se puede hacer por sobrecarga, pero no por sobreescritura de métodos

13. ¿Cuál de las siguientes son interfaces de colecciones?

- HashMap
- ***Map***
- LinkedList
- HashSet
- ***List***

14. ¿Cuál es la mejor forma de concatenar cadenas?

- Usando la clase StringConcat
- Usando la clase StringBuffer
- ***Usando la clase StringBuilder***
- Concatenando con el operador +
- Utilizando StringUtils de apache commons

15. ¿Cuál de los siguienes enunciados es correcto referente a las clases ArrayList y LinkedList?

- Son estructuras diferentes debido a que se ArrayList se utiliza para listas ordenadas y LinkedList se utiliza para listas desordenadas.
- Se prefiere el uso de ArrayList debido a que LinkedList generalmente está en desuso y no se recomienda.
- ArrayList utiliza nodos para interconectar el nodo anterior y siguiete.

- LinkedList utiliza memoria contigua para crear los nodos o buckets donde almacenará sus objetos.
- ***La inserción y eliminación de objetos en un ArrayList es más lenta comparada con las mismas operaciones en un LinkedList.***

16. Formas de inyectar dependencias en Spring:

- Por reflexión, por constructor
- Por setter, por constructor
- Por getter, por setter
- ***Por reflexión, por constructor, por setter***
- Por reflexión, por getter , por setter

17. Son tipos de scopes en Spring:

- Request, application, factory
- Singleton, factory, application
- ***Session, prototype, request***
- Singleton, prototype, factory

18. ¿Para qué se utiliza la palabra reservada Synchronized?

- Esa palabra no existe en Java, solo aplica en Python.
- Se utiliza para permitir la ejecución concurrente de hilos, es decir, habilita que 2 o más hilos ejecuten el mismo fragmento de código.
- ***Se utiliza para sincronizar métodos estáticos o de clase que permitan la ejecución de un solo hilo a la vez.***
- Se utiliza para sincronizar métodos estáticos o de clase que permitan la ejecución de un solo hilo a la vez, solo si y, siempre y cuando, el programa se ejecute en modo debug con la opción -b.
- Synchronized habilita que dos ejecutores o más ejecuten hilos sobre el mismo método al mismo tiempo.

19. ¿Cuáles son los modificadores de acceso?

- final, public, private, protected
- ***public, protected, private, default***
- final, static, public, private, protected
- public protected, private, static

20. ¿Para qué se utiliza la palabra reservada “final” a nivel de clase?

- ***Para evitar que la clase sea sobre-escrita.***
- Para evitar que los métodos de la clase sean sobre-escritos.
- Para evitar que la clase sea extendida.
- No se puede aplicar la palabra reservada “final” a nivel de clase

Technical Theorical Questions

I. What is the difference between an Applet and a Java Application?

Answer:

- Applets are executed within a Java enabled browser, but a
- Java application is a standalone Java program that can be executed outside of a browser.

However, they both require the existence of a Java Virtual Machine (JVM). Furthermore, a Java application requires a main method with a specific signature, in order to start its execution. Java applets don't need such a method to start their execution. Finally, Java applets typically use a restrictive security policy, while Java applications usually use more relaxed security policies.

II. What is a JSP Page?

Answer:

A **Java Server Page (JSP)** is a text document that contains two types of text:

- static data
- and JSP elements.

Static data can be expressed in any text-based format, such as HTML or XML. JSP is a technology that mixes static content with dynamically-generated content.

III. What is a Servlet?

Answer:

The servlet is a Java programming language class used to process client requests and generate dynamic web content. Servlets are mostly used to process or store data submitted by an HTML form, provide dynamic content and manage state information that does not exist in the stateless HTTP protocol.

IV. What does the static keyword mean? Can you override private or static method in Java?

Answer:

The static keyword denotes that a member variable or method can be accessed, without requiring an instantiation of the class to which it belongs.

A user cannot override static methods in Java, because method overriding is based upon dynamic binding at runtime and static methods are statically binded at compile time. A static method is not associated with any instance of a class so the concept is not applicable.

V. What are pass by reference and pass by value?

Answer:

- When an object is passed by value, this means that a copy of the object is passed. Thus, even if changes are made to that object, it doesn't affect the original value.
- When an object is passed by reference, this means that the actual object is not passed, rather a reference of the object is passed. Thus, any changes made by the external method, are also reflected in all places.

VI. What are the basic interfaces of Java Collections Framework?

Answer:

Java Collections Framework provides a well designed set of interfaces and classes that support operations on a collections of objects. The most basic interfaces that reside in the Java Collections Framework are:

- **Collection**, which represents a group of objects known as its elements.
- **Set**, which is a collection that cannot contain duplicate elements.
- **List**, which is an ordered collection and can contain duplicate elements.
- **Map**, which is an object that maps keys to values and cannot contain duplicate keys.

VII. What is an Iterator?

Answer:

The Iterator interface provides a number of methods that are able to iterate over any Collection. Each Java Collection contains the Iterator method that returns an Iterator instance. Iterators are capable of removing elements from the underlying collection during the iteration.

VIII. How HashMap works in Java?

Answer:

A HashMap in Java stores key-value pairs. The HashMap requires a hash function and uses hashCode and equals methods, in order to put and retrieve elements to and from the collection respectively. When the put method is invoked, the HashMap calculates the hash value of the key and stores the pair in the appropriate index inside the collection. If the key exists, its value is updated with the new value. Some important characteristics of a HashMap are its capacity, its load factor and the threshold resizing.

IX. What differences exist between HashMap and Hastable?

Answer:

There are several differences between HashMap and Hashtable in Java:

1. Hashtable is synchronized, whereas HashMap is not. This makes HashMap better for non-threaded applications, as unsynchronized Objects typically perform better than synchronized ones.
2. Hashtable does not allow null keys or values. HashMap allows null keys and any number of null values.
3. One of HashMap's subclasses is LinkedHashMap, so if you'd want predictable iteration order (which is insertion order by default), you could easily swap out the HashMap for a LinkedHashMap. This wouldn't be as easy if you were using Hashtable.

X. What does System.gc() and Runtime.gc() methods do?

Answer:

These methods can be used as a hint to the JVM, in order to start a garbage collection. However, this it is up to the Java Virtual Machine (JVM) to start the garbage collection immediately or later in time.

XI. When does an Object becomes eligible for Garbage Collection?

Answer:

A Java object is subject to garbage collection when it becomes unreachable to the program in which it is currently used.

XII. What is the difference between Exception and Error in Java?

Answer:

An **Error** "indicates serious problems that a reasonable application should not try to catch."

- An **Exception** "indicates conditions that a reasonable application might want to catch."

XIII. What is the importance of finally block in exception handling?

Answer:

A finally block will always be executed, whether or not an exception is actually thrown. Even in the case where the catch statement is missing and an exception is thrown, the finally block will still be executed. Last thing to mention is that the finally block is used to release resources like I/O buffers, database connections, etc.

XIV. What is a Java Applet?

Answer:

A Java Applet is program that can be included in a HTML page and be executed in a java enabled client browser. Applets are used for creating dynamic and interactive web applications.

XV. What is JDBC?

Answer:

JDBC is an abstraction layer that allows users to choose between databases. JDBC enables developers to write database applications in Java, without having to concern themselves with the underlying details of a particular database.

XVI. What are Directives?

Answer:

What are the different types of Directives available in JSP ? Directives are instructions that are processed by the JSP engine, when the page is compiled to a servlet. Directives are used to set page-level instructions, insert data from external files, and specify custom tag libraries. Directives are defined between < %@ and % >.The different types of directives are shown below:

- **Include directive:** it is used to include a file and merges the content of the file with the current page.
- **Page directive:** it is used to define specific attributes in the JSP page, like error page and buffer.
- **Taglib:** it is used to declare a custom tag library which is used in the page.

XVII. Explain what is Binary Search

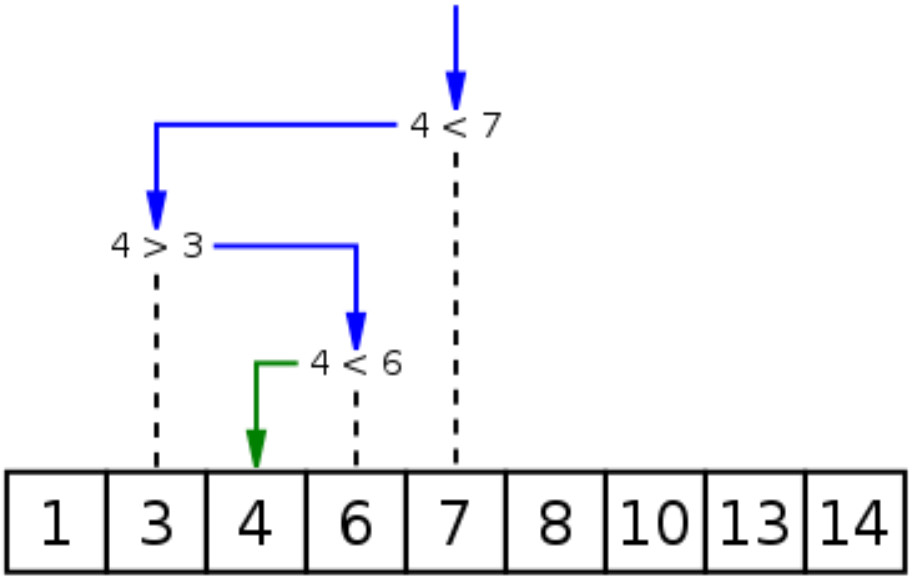
Answer:

When the list is sorted we can use the binary search (also known as half-interval search, logarithmic search, or binary chop) technique to find items on the list. Here's a step-by-step description of using binary search:

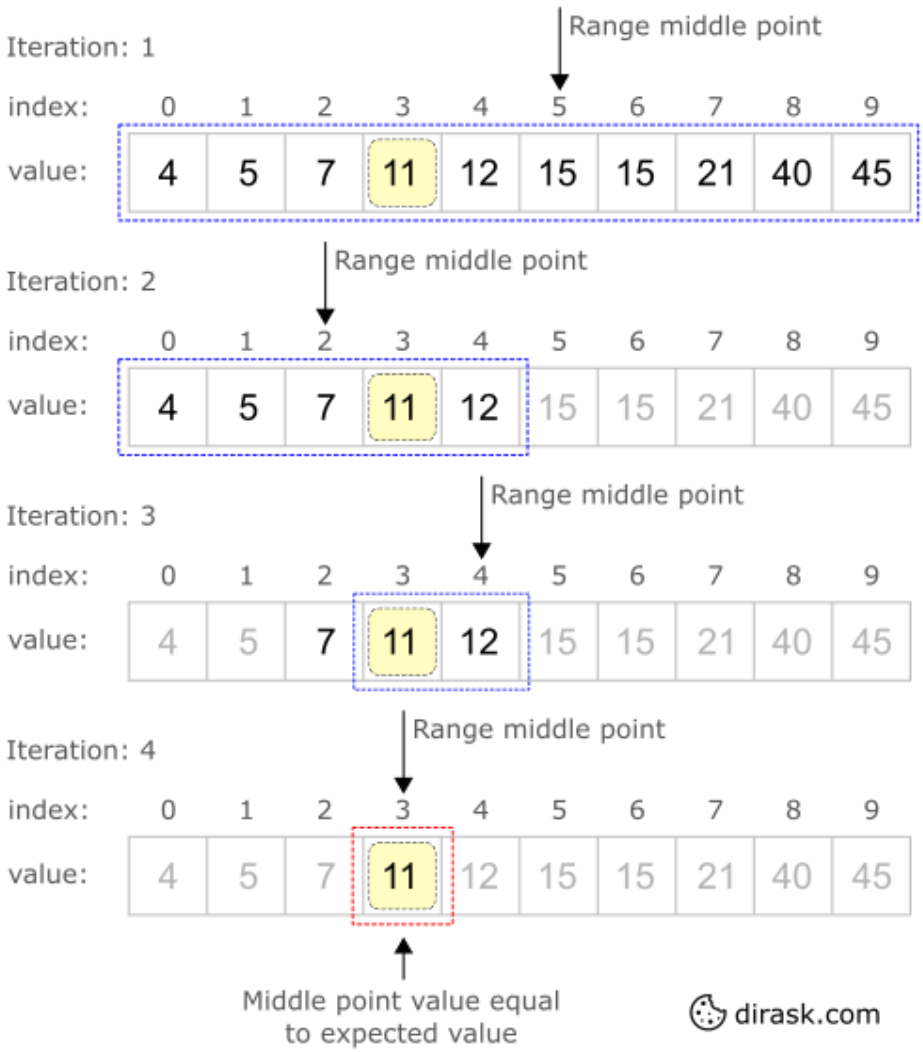
1. Let **min = 1** and **max = n.**
2. Guess the average of **max** and **min** rounded down so that it is an integer.
3. If you guessed the number, stop. You found it!
4. If the guess was too low, set min to be one larger than the guess.
5. If the guess was too high, set max to be one smaller than the guess.

6. Go back to step two.

In this example we looking for array item with value 4 :



When you do one operation in binary search we reduce the size of the problem **by half** (look at the picture below how do we reduce the size of the problem area) hence the complexity of binary search is $O(\log n)$. The binary search algorithm can be written either *recursively* or *iteratively*.



Complexity Analysis:

Time Complexity: O(log n) **Space Complexity**: O(log n)

Implementation:

JS

```
var binarySearch = function(array, value) {
    var guess,
        min = 0,
        max = array.length - 1;

    while(min <= max){
        guess = Math.floor((min + max) /2);
    if(array[guess] === value)
        return guess;
    else if(array[guess] < value)
        min = guess + 1;
    else
        max = guess - 1;
     }

     return -1;
```

Java

```
// binary search example in Java
/* here Arr is an of integer type, n is size of array
   and target is element to be found */

int binarySearch(int Arr[], int n, int target) {

    //set stating and ending index
    int start = 0, ending = n-1;

    while(start <= ending) {
        // take mid of the list
        int mid = (start + end) / 2;

        // we found a match
        if(Arr[mid] == target) {
            return mid;
        }
        // go on right side
        else if(Arr[mid] < target) {
            start = mid + 1;
        }
        // go on left side
        else {
            end = mid - 1;
        }
    }
    // element is not present in list
```

XVIII. What are the Data Types supported by Java? What is Autoboxing and Unboxing?

Answer:

The eight primitive data types supported by the Java programming language are:

- byte
- short
- int
- long
- float
- doub
- le
- boolean
- char

Autoboxing is the automatic conversion made by the Java compiler between the primitive types and their corresponding object wrapper classes. If the conversion goes the other way, this operation is called **unboxing**.

XIX. What is the difference between Interface and Abstract Class?

Answer:

Java provides and supports the creation both of **abstract** classes and **interfaces**. Both implementations share some common characteristics, but they differ in the following features:

- All methods in an interface are implicitly abstract. On the other hand, an abstract class may contain both abstract and non-abstract methods.
- A class may implement a number of Interfaces, but can extend only one abstract class.
- In order for a class to implement an interface, it must implement all its declared methods. However, a class may not implement all declared methods of an abstract class. Though, in this case, the sub-class must also be declared as abstract.
- Abstract classes can implement interfaces without even providing the implementation
- of interface methods. Variables declared in a Java interface is by default final. An abstract class may contain non-final variables.
- Members of a Java interface are public by default. A member of an abstract class can either be private, protected or public.
- An interface is absolutely abstract and cannot be instantiated. An abstract class also cannot be instantiated, but can be invoked if it contains a main method.

XX. What is the difference between processes and threads?

Answer:

The main difference between them is that

- a **Process** is a program which is executing some code and
- a **Thread** is an independent path of execution in the process.

A process can have more than one thread for doing independent task e.g. a thread for reading data from disk, a thread for processing that data and another thread for sending that data over the network.

XXI. What is Function Overriding and Overloading in Java?

Answer:

- Method **overloading** in Java occurs when two or more methods in the same class have the exact same name, but different parameters.

```
class Dog{
    public void bark(){
        System.out.println("woof ");
    }
    //overloading method
    public void bark(int num){
        for(int i=0; i<num; i++)
            System.out.println("woof ");
    }
}
```

- On the other hand, method **overriding** is defined as the case when a child class redefines the same method as a parent class. 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.

```
class Dog{
    public void bark(){
        System.out.println("woof ");
    }
}

class Hound extends Dog{
    public void sniff(){
        System.out.println("sniff ");
    }
    public void bark(){
        System.out.println("bowl");
    }
}

public class OverridingTest{
    public static void main(String [] args){
        Dog dog = new Hound();
        dog.bark();
    }
}
```

XXII. What do you know about the Big-O notation and can you give some examples with respect to different data structures?

Answer:

The Big-O notation simply describes how well an algorithm scales or performs in the worst case scenario as the number of elements in a data structure increases. The Big-O notation can also be used to describe other behavior such as memory consumption. Since the collection classes are actually data structures, we usually use the Big-O notation to chose the best implementation to use, based on time, memory and performance. Big-O notation can give a good indication about performance for large amounts of data.

XXIII. What is the purpose of Garbage Collection in Java, and when is it used?

Answer:

The purpose of garbage collection is to identify and discard those objects that are no longer needed by the application, in order for the resources to be reclaimed and reused.

XXIV. What will happen to the Exception object after exception handling?

Answer:

The Exception object will be garbage collected in the next garbage collection.

XXV. What is the design pattern that Java uses for all Swing components?

Answer:

The design pattern used by Java for all Swing components is the Model View Controller (MVC) pattern.

XXVI. What is the purpose of Class.forName method?

Answer:

This method is used to method is used to load the driver that will establish a connection to the database.

XXVII. How are the JSP requests handled?

Answer:

On the arrival of a JSP request, the browser first requests a page with a .jsp extension. Then, the Web server reads the request and using the JSP compiler, the Web server converts the JSP page into a servlet class. Notice that the JSP file is compiled only on the first request of the page, or if the JSP file has changed.The generated servlet class is invoked, in order to handle the browser's request. Once the execution of the request is over, the servlet sends a response back to the client. See how to get Request parameters in a JSP.

XXVIII. What are JSP Actions?

Answer:

JSP actions use constructs in XML syntax to control the behavior of the servlet engine. JSP actions are executed when a JSP page is requested. They can be dynamically inserted into a file, re-use JavaBeans components, forward the user to another page, or generate HTML for the Java plugin.Some of the available actions are listed below:

- jsp:include – includes a file, when the JSP page is requested. jsp:useBean – finds or instantiates a JavaBean. jsp:setProperty – sets the property of a JavaBean. jsp:getProperty – gets the property of a JavaBean. jsp:forward – forwards the requester to a new page. jsp:plugin – generates browser-specific code.

XXIX. What are Decalarations?

Answer:

Declarations are similar to variable declarations in Java. Declarations are used to declare variables for subsequent use in expressions or scriptlets. To add a declaration, you must use the sequences to enclose your declarations.

XXX. What are Expressions?

Answer:

A JSP expression is used to insert the value of a scripting language expression, converted into a string, into the data stream returned to the client, by the web server. Expressions are defined between <% = and %> tags.

XXXI. Explain the architechure of a Servlet.

Answer:

The core abstraction that must be implemented by all servlets is the javax.servlet.Servlet interface. Each servlet must implement it either directly or indirectly, either by extending javax.servlet.GenericServlet or javax.servlet.http.HTTPServlet. Finally, each servlet is able to serve multiple requests in parallel using multithreading.

XXXII. What's the difference between sendRedirect and forward methods?

Answer:

The sendRedirect method creates a new request, while the forward method just forwards a request to a new target. The previous request scope objects are not available after a redirect, because it results in a new request. On the other hand, the previous request scope objects are available after forwarding. FInally, in general, the sendRedirect method is considered to be slower compare to the forward method.

XXXIII. Explain Serialization and Deserialization.

Answer:

Java provides a mechanism, called object serialization where an object can be represented as a sequence of bytes and includes the object's data, as well as information about the object's type, and the types of data stored in the object. Thus, serialization can be seen as a way of flattening objects, in order to be stored on disk, and later, read back and reconstituted. Deserialisation is the reverse process of converting an object from its flattened state to a live object.

XXXIV. What is reflection and why is it useful?

Answer:

The name **reflection** is used to describe code which is able to inspect other code in the same system (or itself) and to make modifications at runtime.

For example, say you have an object of an unknown type in Java, and you would like to call a 'doSomething' method on it if one exists. Java's static typing system isn't really designed to support this unless the object conforms to a known interface, but using reflection, your code can look at the object and find out if it has a method called 'doSomething' and then call it if you want to.

```
Method method = foo.getClass().getMethod("doSomething", null);
method.invoke(foo, null);
```

XXXV. How does Garbage Collection prevent a Java application from going out of memory?

Answer:

It doesn't! Garbage Collection simply cleans up unused memory when an object goes out of scope and is no longer needed. However an application could create a huge number of large objects that causes an OutOfMemoryError.

XXXVI. What is a Constructor, Constructor Overloading and Copy-Constructor?

Answer:

A constructor gets invoked when a new object is created. Every class has a constructor. In case the programmer does not provide a constructor for a class, the Java compiler (Javac) creates a default constructor for that class. The constructor overloading is similar to method overloading in Java. Different constructors can be created for a single class. Each constructor must have its own unique parameter list. Finally, Java does support copy constructors like C++, but the difference lies in the fact that Java doesn't create a default copy constructor if you don't write your own.

XXXVII. What is the role of stub in RMI?

Answer:

A stub for a remote object acts as a client's local representative or proxy for the remote object. The caller invokes a method on the local stub, which is responsible for executing the method on the remote object. When a stub's method is invoked, it undergoes the following steps:

- It initiates a connection to the remote JVM containing the
- remote object. It marshals the parameters to the remote JVM.
- It waits for the result of the method invocation and execution.
- It unmarshals the return value or an exception if the method has not been
- successfully executed.

It returns the value to the caller.

XXXVIII. Can you access non static variable in static context?

Answer:

A static variable in Java belongs to its class and its value remains the same for all its instances. A static variable is initialized when the class is loaded by the JVM. If your code tries to access a non-static variable, without any instance, the compiler will complain, because those variables are not created yet and they are not associated with any instance.

XXXIX. Why Collection doesn't extends Cloneable and Serializable interfaces?

Answer:

The Collection interface specifies groups of objects known as elements. Each concrete implementation of a Collection can choose its own way of how to maintain and order its elements. Some collections allow duplicate keys, while some other collections don't. The semantics and the implications of either cloning or serialization come into play when dealing with actual implementations. Thus, the concrete implementations of collections should decide how they can be cloned or serialized.

XL. What differences exist between Iterator and ListIterator?

Answer:

The differences of these elements are listed below:

- An Iterator can be used to traverse the Set and List collections, while the ListIterator can be used to iterate only over List .
- The Iterator can traverse a collection only in forward direction, while the ListIterator can traverse a List in both directions.
- The ListIterator implements the Iterator interface and contains extra functionality, such as adding an element, replacing an element, getting the index position for previous and next elements, etc

XLI. What is the tradeoff between using an unordered array versus an ordered array?

Answer:

The major advantage of an ordered array is that the search times have time complexity of O(log n), compared to that of an unordered array, which is O (n). The disadvantage of an ordered array is that the insertion operation has a time complexity of O(n), because the elements with higher values must be moved to make room for the new element. Instead, the insertion operation for an unordered array takes constant time of O(1).

XLII. What is structure of Java Heap?

Answer:

The JVM has a heap that is the runtime data area from which memory for all class instances and arrays is allocated. It is created at the JVM start-up. Heap memory for objects is reclaimed by an automatic memory management system which is known as a garbage collector. Heap memory consists of live and dead objects. Live objects are accessible by the application and will not be a subject of garbage collection. Dead objects are those which will never be accessible by the application, but have not been collected by the garbage collector yet. Such objects occupy the heap memory space until they are eventually collected by the garbage collector.

XLIII. What is Comparable and Comparator interface? List their differences.

Answer:

Java provides the Comparable interface, which contains only one method, called compareTo. This method compares two objects, in order to impose an order between them. Specifically, it returns a negative integer, zero, or a positive integer to indicate that the input object is less than, equal or greater than the existing object. Java provides the Comparator interface, which contains two methods, called compare and equals. The first method compares its two input arguments and imposes an order between them. It returns a negative integer, zero, or a positive integer to indicate that the first argument is less than, equal to, or greater than the second. The second method requires an object as a parameter and aims to decide whether the input object is equal to the comparator. The method returns true, only if the specified object is also a comparator and it imposes the same ordering as the comparator.

XLIV. When is the finalize() called? What is the purpose of finalization?

Answer:

The finalize method is called by the garbage collector, just before releasing the object's memory. It is normally advised to release resources held by the object inside the finalize method.

XLV. What is the difference between throw and throws?

Answer:

The **throw** keyword is used to explicitly raise a exception within the program. On the contrary, the **throws**
clause is used to indicate those exceptions that are not handled by a method. Each method must explicitly specify which exceptions does not handle, so the callers of that method can guard against possible exceptions. Finally, multiple exceptions are separated by a comma.

XLVI. Explain the life cycle of an Applet.

Answer:

An applet may undergo the following states:

- Init: An applet is initialized each time is
- loaded. Start: Begin the execution of an applet.
- Stop: Stop the execution of an applet.
- Destroy: Perform a final cleanup, before unloading the applet.

XLVII. What happens when an Applet is loaded?

Answer:

First of all, an instance of the applet's controlling class is created. Then, the applet initializes itself and finally, it starts running.

XLVIII. What are the restrictions imposed on Java applets?

Answer:

Mostly due to security reasons, the following restrictions are imposed on Java applets:

- An applet cannot load libraries or define native methods.
- An applet cannot ordinarily read or write files on the
- execution host. An applet cannot read certain system properties.
- An applet cannot make network connections except to the host that it came from.
- An applet cannot start any program on the host that's executing it.

XLIX. What are untrusted applets?

Answer:

Untrusted applets are those Java applets that cannot access or execute local system files. By default, all downloaded applets are considered as untrusted.

L. What is the applet security manager, and what does it provide?

Answer:

The applet security manager is a mechanism to impose restrictions on Java applets. A browser may only have one security manager. The security manager is established at startup, and it cannot thereafter be replaced, overloaded, overridden, or extended.

LI. Which Swing methods are thread-safe?

Answer:

There are only three thread-safe methods:

- repaint,
- revalidate,
- invalidate.

LII. What is the relationship between an event-listener interface and an event-adapter class?

Answer:

An event-listener interface defines the methods that must be implemented by an event handler for a particular event. An event adapter provides a default implementation of an event-listener interface.

LIII. Explain the role of Driver in JDBC.

Answer:

The JDBC Driver provides vendor-specific implementations of the abstract classes provided by the JDBC API. Each driver must provide implementations for the following classes of the java.sql package: Connection, Statement, PreparedStatement, CallableStatement, ResultSet and Driver.

LIV. What is the difference between GenericServlet and HttpServlet?

Answer:

GenericServlet is a generalized and protocol-independent servlet that implements the Servlet and ServletConfig interfaces. Those servlets extending the GenericServlet class shall override the service method. Finally, in order to develop an HTTP servlet for use on the Web that serves requests using the HTTP protocol, your servlet must extend the HttpServlet instead. Check Servlet examples here.

LV. Explain the life cycle of a Servlet.

Answer:

On every client's request, the Servlet Engine loads the servlets and invokes its init methods, in order for the servlet to be initialized. Then, the Servlet object handles all subsequent requests coming from that client, by invoking the service method for each request separately. Finally, the servlet is removed by calling the server's destroy method.

LVI. What is the difference between doGet() and doPost() ?

Answer:

- **doGET**: The GET method appends the name-value pairs on the request's URL. Thus, there is a limit on the number of characters and subsequently on the number of values that can be used in a client's request.
  Furthermore, the values of the request are made visible and thus, sensitive information must not be passed in that way.
- **doPOST**: The POST method overcomes the limit imposed by the GET request, by sending the values of the request inside its body. Also, there is no limitations on the number of values to be sent across. Finally, the
  sensitive information passed through a POST request is not visible to an external client.

LVII. What is a Server Side Include (SSI)?

Answer:

Server Side Includes (SSI) is a simple interpreted server-side scripting language, used almost exclusively for the Web, and is embedded with a servlet tag. The most frequent use of SSI is to include the contents of one or more files into a Web page on a Web server. When a Web page is accessed by a browser, the Web server replaces the servlet tag in that Web page with the hyper text generated by the corresponding servlet.

LVIII. Does Java support multiple inheritance?

Answer:

No, Java does not support multiple inheritance. Each class is able to extend only on one class, but is able to implement more than one interfaces.

LIX. Explain different ways of creating a thread. Which one would you prefer and why?

Answer:

There are three ways that can be used in order for a Thread to be created:

- A class may extend the Thread class.
- A class may implement the Runnable interface.
- An application can use the Executor framework, in order to create a thread pool.

The Runnable interface is preferred, as it does not require an object to inherit the Thread class. In case your application design requires multiple inheritance, only interfaces can help you. Also, the thread pool is very efficient and can be implemented and used very easily.

LX. What's a deadlock?

Answer:

A condition that occurs when two processes are waiting for each other to complete, before proceeding. The result is that both processes wait endlessly.

LXI. What is difference between fail-fast and fail-safe?

Answer:

The Iterator's fail-safe property works with the clone of the underlying collection and thus, it is not affected by any modification in the collection. All the collection classes in java.util package are fail-fast, while the collection classes in java.util.concurrent are fail-safe. Fail-fast iterators throw a ConcurrentModificationException, while fail- safe iterator never throws such an exception.

LXII. What is the importance of hashCode() and equals() methods?

Answer:

A HashMap in Java uses the hashCode and equals methods to determine the index of the key-value pair. These methods are also used when we request the value of a specific key. If these methods are not implemented correctly, two different keys might produce the same hash value and thus, will be considered as equal by the collection. Furthermore, these methods are also used to detect duplicates. Thus, the implementation of both methods is crucial to the accuracy and correctness of the HashMap.

LXIII. What is the difference between Array and ArrayList? When will you use Array over ArrayList?

Answer:

The Array and ArrayList classes differ on the following features:

- Arrays can contain primitive or objects, while an ArrayList can contain
- only objects. Arrays have fixed size, while an ArrayList is dynamic.
- An ArrayList provides more methods and features, such as addAll, removeAll, iterator, etc.
- For a list of primitive data types, the collections use autoboxing to reduce the coding effort. However, this approach makes them slower when working on fixed size primitive data types.

LXIV. What is the difference between ArrayList and LinkedList?

Answer:

Both the ArrayList and LinkedList classes implement the List interface, but they differ on the following features:

- An ArrayList is an index based data structure backed by an Array. It provides random access to its elements with a performance equal to O(1). On the other hand, a LinkedList stores its data as list of elements and every element is linked to its previous and next element. In this case, the search operation for an element has execution time equal to O(n).
- The Insertion, addition and removal operations of an element are faster in a LinkedList compared to an ArrayList, because there is no need of resizing an array or updating the index when an element is added in some arbitrary position inside the collection.
- A LinkedList consumes more memory than an ArrayList, because every node in a LinkedList stores two references, one for its previous element and one for its next element.

LXV. What's the difference between Iterator and interfaces?

Answer:

Enumeration is twice as fast as compared to an Iterator and uses very less memory. However, the Iterator is much safer compared to Enumeration, because other threads are not able to modify the collection object that is currently traversed by the iterator. Also, Iterators allow the caller to remove elements from the underlying collection, something which is not possible with Enumeration.

LXVI. If an object reference is set to null, will the Garbage Collector immediately free the memory held by the object?

Answer:

No, the object will be available for garbage collection in the next cycle of the garbage collector.

LXVII. How does the finally block differ from finalize() method?

Answer:

A finally block will be executed whether or not an exception is thrown and is used to release those resources held
by the application. `Finalize` is a protected method of the Object class, which is called by the Java Virtual
Machine (JVM) just before an object is garbage collected.

LXVIII. What is the advantage of PreparedStatement over Statement?

Answer:

PreparedStatements are precompiled and thus, their performance is much better. Also, PreparedStatement objects can be reused with different input values to their queries.

LXIX. What are Scriptlets?

Answer:

In Java Server Pages (JSP) technology, a scriptlet is a piece of Java-code embedded in a JSP page. The scriptlet is everything inside the tags. Between these tags, a user can add any valid

scriplet.

LXX. What is meant by JSP implicit objects and what are they?

Answer:

JSP implicit objects are those Java objects that the JSP Container makes available to developers in each page. A developer can call them directly, without being explicitly declared. JSP Implicit Objects are also called pre-defined variables.The following objects are considered implicit in a JSP page:

- application page
- request
- response
- session
- exception out
- config
- pageContext

LXXI. What is the difference between an Applet and a Servlet?

Answer:

An Applet is a client side java program that runs within a Web browser on the client machine. On the other hand, a servlet is a server side component that runs on the web server.An applet can use the user interface classes, while a servlet does not have a user interface. Instead, a servlet waits for client's HTTP requests and generates a response in every request.

LXXII. What are the steps involved to make work a RMI program?

Answer:

The following steps must be involved in order for a RMI program to work properly:

- Compilation of all source files.
- Generatation of the stubs using rmic. Start the rmiregistry.
- Start the RMIServer. Run the client program.

LXXIII. What are the differences between == and equals?

Answer:

As a reminder, it needs to be said that generally, `==` is NOT a viable alternative to `equals`. When it is, however
(such as with enum), there are two important differences to consider:

1. `==` never throws `NullPointerException`

```
enum Color { BLACK, WHITE };

Color nothing = null;
if (nothing == Color.BLACK);        // runs fine
if (nothing.equals(Color.BLACK)); // throws NullPointerEx
```

2. `==` is subject to type compatibility check at compile time

```
enum Color { BLACK, WHITE };
enum Chiral { LEFT, RIGHT };

if (Color.BLACK.equals(Chiral.LEFT)); // compiles fine
if (Color.BLACK == Chiral.LEFT);      // DOESN'T COMPILE!!! Incompatible types!
```

LXXIV. Is there anything like static class in Java?

Answer:

Java has **no way** of making a top-level class static but you can simulate a static class like this:

- Declare your class final - Prevents extension of the class since extending a static class makes no sense
- Make the constructor private - Prevents instantiation by client code as it makes no sense to instantiate a static class
- Make all the members and functions of the class static - Since the class cannot be instantiated no instance methods can be called or instance fields accessed
- Note that the compiler will not prevent you from declaring an instance (non-static) member. The issue will only show up if you attempt to call the instance member

LXXV. How threadsafe is enum in Java?

Answer:

Creation of an **enum** is guaranteed to be threadsafe. However, the methods on an **enum** are not necessarily threadsafe.

LXXVI. Compare the sleep() and wait() methods in Java?

Answer:

- `sleep()` is a blocking operation that keeps a hold on the monitor / lock of the shared object for the specified number of milliseconds.

- wait(), on the other hand, simply pauses the thread until either (a) the specified number of milliseconds have elapsed or (b) it receives a desired notification from another thread (whichever is first), without keeping a hold on the monitor/lock of the shared object.

`sleep()` is most commonly used for polling, or to check for certain results, at a regular interval.

`wait()` is generally used in multithreaded applications, in conjunction with notify() / notifyAll(), to achieve synchronization and avoid race conditions.

SELF EXAMINATION

Arquitectura y cultura de desarrollo.

A. CLEAN Architecture (Onion, Hex model)

   https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html

B. Principios SOLID (https://profile.es/blog/principios-solid-desarrollo-software-calidad/)

The SOLID principles are a set of rules and best practices to follow when designing a class structure in object-oriented programming.

These principles help us understand the need for certain design patterns and software architecture in general. The five SOLID principles are:

- *Single Responsibility Principle*: A class should have only one reason to change.
- *Open-Closed Principle*: Software entities (classes, modules, functions, etc.) should be open for extension but closed for modification.
- *Liskov Substitution Principle*: Derived classes must be substitutable for their base classes.
- *Interface Segregation Principle*: Clients should not be forced to depend on interfaces they do not use.
- *Dependency Inversion Principle*: High-level entities should not depend on low-level entities. Both should depend on abstractions.

C. Diagramado UML

D. Agile methodology: SCRUM

E. TDD (Test-Driven Development): TDD stands for Test-Driven Development. It is a software development methodology that emphasizes writing automated tests before writing the actual code. The TDD process typically follows these steps:

   - **Write a test**: The developer writes a test case that defines the expected behavior of a small piece of functionality. The test initially fails since the functionality hasn't been implemented yet.

- **Run the test**: The developer runs all the tests, including the newly created one. Since the new test fails, it confirms that the expected behavior is not yet implemented.
- **Write the code**: The developer writes the minimum amount of code necessary to make the test pass. The goal is to implement the functionality required by the test, and nothing more.
- **Run the tests**: The developer reruns all the tests, including the new one. If all the tests pass, it indicates that the newly implemented code works correctly, and no existing functionality has been broken.
- **Refactor**: The developer refactors the code if needed, improving its design, performance, or readability, while ensuring that all the tests continue to pass.
- **Repeat**: The developer writes the next test case for the next piece of functionality and repeats the process, ensuring that each test fails initially before implementing the corresponding code.

By following this cycle of writing, a failing test, implementing the code to make it pass, and then refactoring, TDD helps developers create code that is modular, reliable, and easier to maintain. The emphasis on automated tests also provides a safety net for detecting and preventing regressions as the codebase evolves over time.

Evaluación Técnica

JAVA.

A. Que es una Clase?
B. Que es un Objeto?
C. Cuales son los niveles e acceso de los métodos dentro de una clase y subclases?
D. Que es un CONSTRUCTOR?
E. Cual es la diferencia entre: Double vs double?
F. Que es la Herencia?
G. Cual es a diferencia entre una Interfaz y una clase abstracta?
H. Diferencia entre spring y springboot
I. Que es la abstracción?
J. Diferencias entre variable local y variable de instancia
K. Diferencia entre un array y un arraylist
L. Diferncia entre hash table y hashmou
M. Como definirias un Collection en JAVA
N. como maneja las excepciones?
O. para que sirve el Finally?
P. Que es la serialización y la des sercialización?
Q. What are the basic interfaces of Java Collections Framework?

Answer:

Java Collections Framework provides a well designed set of interfaces and classes that support operations on a collections of objects. The most basic interfaces that reside in the Java Collections Framework are:

- **Collection**, which represents a group of objects known as its elements.
- **Set**, which is a collection that cannot contain duplicate elements.
- **List**, which is an ordered collection and can contain duplicate elements.
- **Map**, which is an object that maps keys to values and cannot contain duplicate keys.

SPRINGBOOT

A. Cual es el ambito del bean?
B. Cual es el ciclo de vida del bean?
C. Que son los objetos Starters?
D. Cuales son los principales verbos HTTP que se pueden implementar en springboot?
E. differencia entre get y post
F. que es boot?
G. que es un patch?
H. diffrencia entre controllers

I. What is BEAN?

In Spring Boot, a bean is an object that is managed by the Spring IoC container. The Spring IoC container is responsible for instantiating, assembling, and managing these beans. The term bean is a key concept in the Spring Framework. It is used to describe the objects that form the backbone of your application and are managed by the Spring IoC container.

In simple terms, a bean is an object that is created, configured, and managed by the Spring IoC container. The Spring IoC container is responsible for creating the bean, injecting its dependencies, and managing its lifecycle.

J. What is Spring Cloud?

Spring Cloud is a collection of tools and frameworks that help developers build and deploy distributed systems and microservices on the cloud. It provides a set of libraries and tools that help developers quickly build common patterns in distributed systems, such as configuration management, service discovery, circuit breakers, messaging, and more.

K. What is Eureka

Eureka is a service registry and discovery service that is part of the Spring Cloud Netflix project. It provides a way for microservices to register themselves with a service registry and discover other services registered with the same registry.

In other words, Eureka is a client-side service discovery mechanism that allows services to find and communicate with each other without hard-coding the hostname and port 1. Eureka provides a REST API for registering and discovering services, as well as a web-based user interface for monitoring the health of the registered services.

ANGULAR.

A. que es un componente?
B. Que es una clase?
C. El modulo va dentro del modulo o el modulo dentro del componente?
D. Que es SPA?

DATA BASE

A. Que es un store procedure?
B. Que es un trigger?
C. A que se refiere PERFORMANCE en las DBs?
D. Cuales son las las mejores prácticas para tener rendimientos optimos en BDs?
E. BD Relacionales & PLSQL https://www.techonthenet.com/oracle/index.php