Q. blocking queue.

Q. immutable class

Q. Future vs CompletableFuture

Q. cohesion vs coupling in java

Q. how many ways I can create object of class. New cloning , X

Q. question regarding immutable class.

Q. how many types objects can be created.

Q. volatile and AtomicInteger .

Q. cloning types .

Q. SOLID pattern.

Q. what is Serialization and Externalization , use of serialverionUUID

Q. Optional class java8

Q. use of default and static methods java 8

Q. Predicate vs BiPredicate

Q. Consumer vs ByConsumer

Q. List to Map conversion

Q. List<Integer> find square of all even numbers

Q. List<Student> find all male student whose age > 30 from computer department @

Q. Find average score of all student (gender age , depart , score ).

Q. how to convert any string to lowercase and uppercase .

Q. find first nonrepeatative character from String .

Q. what are the design pattern you have worked on and where

Q. Executable framework.

Q. what are the types of threadpools

Q. difference b/w fixed and cached threadpool

Q. blocking queue

Q. Synchronised vs reenterent lock

Q. fork and join =>what is fork join pool

Q. cycle barrier vs cowndown latch

Q. use of Enum in singleton class

Q. how many ways singleton class can break and how to prevent

Q. Hashmap changes to java 7 to java 8

Q. intermediet and ternary operations in java 8

<https://javaconceptoftheday.com/java-8-stream-intermediate-and-terminal-operations/>

Q. Finalize method in java

Q. Diff b/w Full GC and partial GC,and Major GC vs Minor GC

Q. java Heap memory management { Eden space and two survivor space , old GC , young generation GC}

Q. How to Define

size max and min.

Q. what is default GC Algo in java 8 . G1,G3

Q. Sequence Diagram , class diagram and UML diagram.

Q. class loader types in jvm.

Q. Map vs flateMap

Q. differeance between functional vs oops programming .

Q. Pollymorphysm real time example .

Q. Zenerics , Jacson library.

**Acid Property**

Following are the possible values for isolation level −

possible values for propagation types −

[volatile](http://javarevisited.blogspot.sg/2011/06/volatile-keyword-java-example-tutorial.html)  [race conditions](http://javarevisited.blogspot.sg/2012/02/what-is-race-condition-in.html) , [thread pools](http://javarevisited.blogspot.sg/2013/07/how-to-create-thread-pools-in-java-executors-framework-example-tutorial.html), [wait and notify](http://javarevisited.blogspot.sg/2015/07/how-to-use-wait-notify-and-notifyall-in.html), [callable and future](http://javarevisited.blogspot.sg/2015/06/how-to-use-callable-and-future-in-java.html),

[Producer-Consumer](http://javarevisited.blogspot.sg/2015/06/java-lock-and-condition-example-producer-consumer.html)

What is lock in java

reentrantLoc

[Producer-Consumer pattern using BlockingQueue](http://www.java67.com/2015/12/producer-consumer-solution-using-blocking-queue-java.html)

[CyclicBarrier](http://javarevisited.blogspot.sg/2012/07/cyclicbarrier-example-java-5-concurrency-tutorial.html#axzz5DmwFLA1K) [CountDowLatch,](https://javarevisited.blogspot.com/2012/07/countdownlatch-example-in-java.html#axzz5J9grtXLX)

[ConcurrentHashMap](https://javarevisited.blogspot.com/2017/08/top-10-java-concurrenthashmap-interview.html)

[Fork-Join Frameworks](http://javarevisited.blogspot.sg/2016/12/difference-between-executor-framework-and-ForkJoinPool-in-Java.html)

[MapReduce](http://www.java67.com/2016/09/map-reduce-example-java8.html)

[the Executor framework](http://javarevisited.blogspot.sg/2017/02/difference-between-executor-executorservice-and-executors-in-java.html)

ConcurrentHashMap

How to Make an Object Immutable in Java? Why Should You Make an Object Immutable

[Why String is Immutable in Java](http://javarevisited.blogspot.sg/2010/10/why-string-is-immutable-in-java.html)

**Which Design Patterns have You Used in Your Java Project?**

Solid principle

WeakHashMap

Which Two Methods HashMap key Object Should Implement

HashMap internal working

Memory management

 lambdas, fork-join pool, Phaser,

What is Busy Spinning? Why Should You Use It in Java?

**What is Read-Write Lock?**

ReentrantReadWritLock

Map vs flatMap

Q. best practices of restfull services .

Q. idempotent and non idempotent .

Q. Pagination versioning in rest

Q. Controller vs rest controller .

Q. @profile annotation .

Q. Junit annotations .

Implementing JWT (JSON Web Tokens) with Spring Security in Springboot App

What Is The Most Challenging Project You Have Worked On?

https://www.interviewbit.com/puzzles/

important questons ---

spring boot embedded database = embedded H2, HSQL and Derby databases. (in memory database)

difference between Yaml and Properties file

what is thymeleaf

application.properties => spring.application.name , spring.config.name ,server.port ,

Spring Boot Embedded Web Server : Tomcat , Jetty Server , Undertow Server

Maven what is ArtifactId and groupId

Mockito test framwork , Simple Single Sign-On with Spring Security OAuth2

JUnit4 test framwork

Spring Security JWT , oauth2

ci cd pipeline jenkins

devops ci/cd pipeline https://devops.com/how-to-implement-an-effective-ci-cd-pipeline/

Microservices

Docker framwork , Kubernate

GIT Framwork

MongoDB https://www.guru99.com/mongodb-tutorials.html ,

https://docs.mongodb.com/manual/core/databases-and-collections/

GIT - https://git-scm.com/docs/gittutorial ,https://www.tutorialspoint.com/git/git\_basic\_concepts.htm ,

https://www.vogella.com/tutorials/Git/article.html

Mockito Framwork -

Core java important questions

cross origin in spring .

Flutter developer amazon lamda mongodb S3 Git ", " Jenkins "

what is use of jProfiler .

what is swagger .

https://java2blog.com/java-collections-interview-questions/

How to print even and odd numbers using threads in java

List<Employee> emp ; find duplicate

string str="sgggfgdjdgf" find duplicate ;

Mock final class .

mock static methods .

test void method using mockito

Spring Class A is singlton inside class B that's prototype

Spring Class B is prototype inside class A thats Singlton { all works as singlton use ApplicationContextAware}

is spring bean thread safe .

SQL Query Questions-------

Employee Department

empId depId

name departmentName , empId

Q. find number of employee from it department , number of employee in each depertment

Q. which development model you are following : agile

Q. what are thread pools

Q. diff bet singlton design pattern and singlton bean

<bean id="bean1" class ="Employee" >

<bean id="bean2" class ="Employee" > how many objects will create{

A Spring Singleton does not work like a Java Singleton.

If we see the output of the program, we will understand that it will return two different instances, So in a container, th

}

db table design order product customer (multiple products types)

Q. class A - > class B - > class c - > class D got exeption and how to handle

Q. java 8 filter Employee based on age and salary

Q. How will you handle situation when live application down and find Outoffmemory eror : first stop server and then check all process running and kil them all clean server like some logs and and other restart again .

Q. what is connection pools ;

Q. what is threadProol given situation (miceroservice Front calling to A B C D services parallaly)

Q. String str = "aabbbcccddssaaddffvv" find duplicates and delete duplicates

Q. what is parallel stream java 8

Q. spring security jwt authenication. And how do you authorization request.

Q. time complexity for inserting an element in ArrayList

Q. time complexity for inserting an element in LinkedList

Q. time complexity for retrieving an element from LinkedList

Transaction vihavoirs

Q. how to create checked custom exception

Q. use of static block

maven scops

agile process

Where non static variables are stored in Java?

Image result for where static variables are stored in java

Non-Static variables and Non-Static methods are non-static components of a class. These are also called instance components of a class. Non-static components are stored inside the object memory. Each object will have their own copy of non-static components.

The memory for local variables is allocated on the thread's stack, the memory for instance and class variables reside in the corresponding object's space, which is in the heap. Keep in mind, that variables only store references to objects. They do not store the object!

where primitive varriables are stored : Only local primitive variables and references to object (i.e. variable declared in method) are stored in stack. Others are stored in heap

What is hascode : hashCode() is used for bucketing in Hash implementations like HashMap, HashTable, HashSet, etc.

The value received from hashCode() is used as the bucket number for storing elements of the set/map. This bucket number is the address of the element inside the set/map.

What is custom Exception in java : https://www.baeldung.com/java-new-custom-exception

1) Custom Checked Exception:

Checked exceptions are exceptions that need to be treated explicitly.

Let’s consider a piece of code that returns the first line of the file:

try (Scanner file = new Scanner(new File(fileName))) {

if (file.hasNextLine()) return file.nextLine();

} catch(FileNotFoundException e) {

// Logging, etc

}

The code above is a classic way of handling Java checked exceptions. While the code throws FileNotFoundException, it's not clear what the exact cause is — whether the file doesn't exist or the file name is invalid.

To create a custom exception, we have to extend the java.lang.Exception class.

Let’s see an example of this by creating a custom checked exception called IncorrectFileNameException:

public class IncorrectFileNameException extends Exception {

public IncorrectFileNameException(String errorMessage) {

super(errorMessage);

}

}

2) Custom Unchecked Exception

In our same example, let's assume that we need a custom exception if the file name doesn't contain any extension.

In this case, we'll need a custom unchecked exception similar to the previous one, as this error will only be detected during runtime.

To create a custom unchecked exception, we need to extend the java.lang.RuntimeException class:

public class IncorrectFileExtensionException

extends RuntimeException {

public IncorrectFileExtensionException(String errorMessage, Throwable err) {

super(errorMessage, err);

}

}

Static block :

Java supports a special block, called static block (also called static clause) which can be used for static initializations of a class. This code inside static block is executed only once: the first time the class is loaded into memory. For example, check output of following Java program.

// filename: Main.java

class Test {

static int i;

int j;

// start of static block

static {

i = 10;

System.out.println("static block called ");

}

// end of static block

}

class Main {

public static void main(String args[]) {

// Although we don't have an object of Test, static block is

// called because i is being accessed in following statement.

System.out.println(Test.i);

}

}

static block called

10

Also, static blocks are executed before constructors. For example, check output of following Java program.

If a class has static members that require complex initialization, a static block is the tool to use. Suppose you need a static map of some kind (the purpose is irrelevant here). You can declare it in-line like this:

public static final Map<String, String> initials = new HashMap<String, String>();

However, if you want to populate it once, you can't do that with an in-line declaration. For that, you need a static block:

public static final Map<String, String> initials = new HashMap<String, String>();

static {

initials.put("AEN", "Alfred E. Newman");

// etc.

}

SOLID Design principles ,CORE Java, J2EE (Spring, Spring Boot) , Web services (REST), Ant/Maven,

Code Quality: Unit Testing ( Junit 4/5 + Mockito), Code quality check Tools ( SonarQube, Sona type)

cloud watch , ec2 instances

https://www.javainuse.com/webseries/spring-security-jwt/chap7

https://www.javainuse.com/webseries/spring-security-jwt/chap4

<https://www.bezkoder.com/spring-boot-refresh-token-jwt/>

Below are the Big O performance of common functions of different Java Collections.

List | Add | Remove | Get | Contains | Next | Data Structure

---------------------|------|--------|------|----------|------|---------------

ArrayList | O(1) | O(n) | O(1) | O(n) | O(1) | Array

LinkedList | O(1) | O(1) | O(n) | O(n) | O(1) | Linked List

CopyOnWriteArrayList | O(n) | O(n) | O(1) | O(n) | O(1) | Array

Set | Add | Remove | Contains | Next | Size | Data Structure

----------------------|----------|----------|----------|----------|------|-------------------------

HashSet | O(1) | O(1) | O(1) | O(h/n) | O(1) | Hash Table

LinkedHashSet | O(1) | O(1) | O(1) | O(1) | O(1) | Hash Table + Linked List

EnumSet | O(1) | O(1) | O(1) | O(1) | O(1) | Bit Vector

TreeSet | O(log n) | O(log n) | O(log n) | O(log n) | O(1) | Red-black tree

CopyOnWriteArraySet | O(n) | O(n) | O(n) | O(1) | O(1) | Array

ConcurrentSkipListSet | O(log n) | O(log n) | O(log n) | O(1) | O(n) | Skip List

Queue | Offer | Peak | Poll | Remove | Size | Data Structure

------------------------|----------|------|----------|--------|------|---------------

PriorityQueue | O(log n) | O(1) | O(log n) | O(n) | O(1) | Priority Heap

LinkedList | O(1) | O(1) | O(1) | O(1) | O(1) | Array

ArrayDequeue | O(1) | O(1) | O(1) | O(n) | O(1) | Linked List

ConcurrentLinkedQueue | O(1) | O(1) | O(1) | O(n) | O(n) | Linked List

ArrayBlockingQueue | O(1) | O(1) | O(1) | O(n) | O(1) | Array

PriorirityBlockingQueue | O(log n) | O(1) | O(log n) | O(n) | O(1) | Priority Heap

SynchronousQueue | O(1) | O(1) | O(1) | O(n) | O(1) | None!

DelayQueue | O(log n) | O(1) | O(log n) | O(n) | O(1) | Priority Heap

LinkedBlockingQueue | O(1) | O(1) | O(1) | O(n) | O(1) | Linked List

Map | Get | ContainsKey | Next | Data Structure

----------------------|----------|-------------|----------|-------------------------

HashMap | O(1) | O(1) | O(h / n) | Hash Table

LinkedHashMap | O(1) | O(1) | O(1) | Hash Table + Linked List

IdentityHashMap | O(1) | O(1) | O(h / n) | Array

WeakHashMap | O(1) | O(1) | O(h / n) | Hash Table

EnumMap | O(1) | O(1) | O(1) | Array

TreeMap | O(log n) | O(log n) | O(log n) | Red-black tree

ConcurrentHashMap | O(1) | O(1) | O(h / n) | Hash Tables

ConcurrentSkipListMap | O(log n) | O(log n) | O(1) | Skip List