OOP questions

Software design

Error handling

What does 'fail fast' mean in terms of exception handling? Why is it a good practice?
The fail fast principle stands for stopping the current operation as soon as any unexpected error occurs.
In most cases, adhering to the fail fast principle means we should shut down the application (probably, with a polite apology) in case of any unexpected situation.
"a program that fails fast will throw an exception"

```
try
{
DoSomething();
}
catch (Exception ex)
{
Logger.Log(ex);
throw ;
}
```

Benefits of the fail fast practice
- Shortening the feedback loop. It is remarkable how quickly bugs are revealed when you stick to the fail fast principle. Even if the application is in production, you are still better off letting the end users know if something went wrong with it.
- The confidence the software works as intended.
- The protection of the persistence state. If we allow the software to continue working after an error occurs, it may come into an invalid state, and, more importantly, save that state to the database. This, in turn, leads to a bigger problem – data corruption – which can’t be solved just by restarting the application.

Computer Science

Data structures

How to find the middle element of singly linked list in O(n)?
Method 1: (time: O(n), space: O(1))
Traverse the whole linked list and count the no. of nodes. Now traverse the list again till count/2 and return the node at count/2.
Method 2: (time: O(n), space: O(1))
Traverse linked list using two pointers. Move one pointer by one and the other pointers by two. When the fast pointer reaches the end slow pointer will reach the middle of the linked list.

Given an array of integers going from 1 to 100 (both inclusive) there is a duplicated entry. How to find it?

```
        for(int i = 0; i < arr.length; i++) {
            for(int j = i + 1; j < arr.length; j++) {
                if(arr[i] == arr[j])
                    System.out.println(arr[j]);
```

```
            }
        }
```
Another solution is to have a data structure to count the number of iterations of each integer. This solution could be implemented either with an array or a hash table.

What is a linked list? How to find if a linked list has a loop?
A linked list is a linear data structure, in which the elements are not stored at contiguous memory locations. The elements in a linked list are linked using pointers. In simple words, a linked list consists of nodes where each node contains a data field and a reference(link) to the next node in the list.
Traverse the list one by one and keep putting the node addresses in a Hash Table. At any point, if NULL is reached then return false, and if the next of the current nodes points to any of the previously stored nodes in Hash then return true.

What is the Big O time complexity of the common operations in an ArrayList, LinkedList, HashMap? And of a bubble sort, quicksort, finding items in a Binary Search tree?
ArrayList:
add() – takes O(1) time; however, worst-case scenario, when a new array has to be created and all the elements copied to it, it's O(n)
add(index, element) – on average runs in O(n) time
get() – is always a constant time O(1) operation
remove() – runs in linear O(n) time. We have to iterate the entire array to find the element qualifying for removal.
indexOf() – also runs in linear time. It iterates through the internal array and checks each element one by one, so the time complexity for this operation always requires O(n) time.
contains() – implementation is based on indexOf(), so it'll also run in O(n) time.

LinkedList:
add() – appends an element to the end of the list. It only updates a tail, and therefore, it's O(1) constant-time complexity.
add(index, element) – on average runs in O(n) time
get() – searching for an element takes O(n) time.
remove(element) – to remove an element, we first need to find it. This operation is O(n).
remove(index) – to remove an element by index, we first need to follow the links from the beginning; therefore, the overall complexity is O(n).
contains() – also has O(n) time complexity

Hashmap:
containsKey() - O(1)
get() - O(1)
put - O(1)
remove - O(1)

Bubble sort: This algorithm has a worst-case time complexity of O(n2).The number of swaps in bubble sort equals the number of inversion pairs in the given array. When the array elements are few

and the array is nearly sorted, bubble sort is effective and efficient.
Quick sort : The average time complexity of quick sort is O(N log(N)).
Binary Search: O(log n)

How does HashMap work?
The basic idea behind hashing is to distribute key/value pairs across an array of placeholders or "buckets" in the hashmap.
A hashmap is typically an array of linked lists. When you want to insert a key/value pair, you first need to use the hash function to map the key to an index in the hash table. Given a key, the hash function can suggest an index where the value can be found or stored.
Since your hash map will probably be significantly smaller than the amount of data you're processing, hash collisions are unavoidable. There are two main approaches to handling collisions: chaining and open addressing. Chaining means that each key/value pair in the hashmap, the value is a linked list of data rather than a single cell.

Why is it important for keys in a map to have an immutable type? (Consider String for example.)
Key’s hash code is used primarily in conjunction to its equals() method, for putting a key in map and then getting it back from map. Immutability allows you to get same hash code every time, for a key object.

Database

How can you connect your application to a database server? What are the possible ways?
- Advertising database connection address and port number on an open connection. This isn't advised, because it's totally insecure and hard to manage.
- Have a simple HTTP interface that accepts a login and lets the user enter SQL, and displays the results. This is more controlled, but not much.
- Create an API to the database, and have a server-side app that accepts the inbound API call, keeps connections to the DB, executes the queries appropriate to the API call, and returns results to the caller using a formatted outbound API. This approach is more secure.

What do you know about database normalization?
Normalization is the process of organizing data in a database. This includes creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency. Redundant data wastes disk space and creates maintenance problems. If data that exists in more than one place must be changed, the data must be changed in exactly the same way in all locations.
There are a few rules for database normalization. Each rule is called a "normal form." If the first rule is observed, the database

is said to be in "first normal form." If the first three rules are observed, the database is considered to be in "third normal form."
First normal form
Eliminate repeating groups in individual tables.
Create a separate table for each set of related data.
Identify each set of related data with a primary key.
Second normal form
Create separate tables for sets of values that apply to multiple records.
Relate these tables with a foreign key.
Third normal form
Eliminate fields that do not depend on the key.

Other

What is a garbage collector, in a nutshell?
A garbage collector is a piece of software that performs automatic memory management. Its job is to free any unused memory and ensure that no memory is freed while it is still in use.

Programming paradigms

Procedural

What is casting? What is the difference between up vs downcasting?
Casting is a process that converts a data type into another either manually by the programmer or automatically by the compiler. Upcasting is conversion to a parent type and downcasting is conversion to a child type. Upcasting is always allowed, but downcasting involves a type check and can throw a ClassCastException.

Which order should we catch the exceptions? Why?
The order is whatever matches first, gets executed. If the first catch matches the exception, it executes, if it doesn't, the next one is tried and on and on until one is matched or none are.

Object-oriented

What is a class?
A class is the blueprint or template for its objects.

What is an object?
An object is an instance of a class. Each object has its own state, behavior, and identity.

What is a constructor?
Constructors are special methods without any return value. Their name is always the same as the name of the class, but they can accept parameters that help set the object’s initial state before the application starts using it.
If we do not provide any constructor, JVM assigns a default constructor to the class. This default constructor does not accept

any parameter.

Do we require parameter for constructors?
We are able to define constructors with parameters, however when creating an object without giving parameters the compiler creates a default constructor that needs no paramters to create the object.

What is an interface?
Interface define contracts, which implementing classes need to honor. These contracts are essentially unimplemented methods. Java already has a keyword for unimplemented methods i.e. abstract. Java has the provision where any class can implement any interface, so all the methods declared in interfaces need to be public only.

What are access modifiers?
Java provides four access modifiers to set access levels for classes, variables, methods and constructors i.e. public, private, protected and default. These access level modifiers determine whether other classes can use a particular field or invoke a particular method.
public – accessible everywhere
protected – accessible in the same package and in sub-classes
default – accessible only in the same package
private – accessible only in the same class

public > protected > package-private (or default) > private

What is data hiding?
If a field is declared private in the class then it cannot be accessed by anyone from outside the class and hides field within the class. Therefore, it is also called data hiding.

Can a static method use non-static members?
A static method can only access static data members and static methods of another class or same class but cannot access non-static methods and variables.

What is the difference between hiding a static method and overriding an instance method?
Hiding a static method of a superclass looks like overriding an instance method of a superclass. The main difference can be seen at runtime in the following scenario.

When we override an instance method, we get the benefit of run-time polymorphism.
When we override a static method, we do not get the benefit of run-time polymorphism.

The distinction between hiding a static method and overriding an instance method has important implications:

The version of the overridden instance method that gets invoked is the one in the subclass.
The version of the hidden static method that gets invoked depends on

whether it is invoked from the superclass or the subclass.

Define the following terms: Instantiation, Attribute, Method
Instantiation in computer science refer to the creation of an object (or an “instance” of a given class) in an object-oriented programming (OOP) language.
Attributes are defined in the class template and represent the state of an object. Objects will have data stored in the attributes field. Class attributes belong to the class itself.
Methods are functions that are defined inside a class that describe the behaviors of an object. Each method contained in class definitions starts with a reference to an instance object. Additionally, the subroutines contained in an object are called instance methods. Programmers use methods for reusability or keeping functionality encapsulated inside one object at a time.

Could we access a static variable (or method) from a non-static method? Why?
Yes, a non-static method can access a static variable or call a static method in Java. There is no problem with that because of static members i.e. both static variable and static methods belongs to a class and can be called from anywhere, depending upon their access modifier.

Could we access a non-static variable (or method) from a static method? Why?
No.Each instance of a non-static variable has a different value and is created when the new() operator initializes an instance of an object. The static variables are created or initialized when the class loads into JVM
. We need an instance of an object for calling the non-static variable. We can create many objects by giving different values to that non-static or instance variable.

How many instances you have of a static variable of a given class?
one

Why is it not a good practice to write a lot of static methods?
Static methods inside a class are usually a good indication of a method that doesn’t belong to this particular class. It doesn’t use the state of the class or other non-static members that the class has, and therefore, they break the Single Responsibility Principle.
Static methods are not polymorphic. The definition of polymorphism is the use of a single interface for entities of different types. So, by definition, static is not polymorphic. The Static method belongs to the class and not to the class instance, therefore you can’t achieve polymorphism with static.
Static methods can’t be used for abstraction and inheritance. You can’t declare a static method in an interface or static abstract method in an abstract class. A static method cannot access non-static class level members, not its own, nor its base class. (Even though in TypeScript and Java, a derived class inherits its base class static members, it still doesn’t fit well as mentioned).

Static methods are bad for testability. Since static methods belong to the class and not a particular instance, mocking them becomes difficult and dangerous. Overriding a static method is not that simple for some languages. Even if you succeed, it will affect other tests that rely on the original implementation and lead to mutations that you didn’t expect to be.

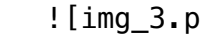
What are the features of static attributes and static methods of a class? What are the benefits, when to use them?
Variable:
The static variable can be used to refer to the common property of all objects (which is not unique for each object), for example, the company name of employees, college name of students, etc.
The static variable gets memory only once in the class area at the time of class loading.
It makes your program memory efficient (i.e., it saves memory).
Since static variables belong to a class, we can access them directly using class name. So, we don't need any object reference.
We can only declare static variables at the class level.
We can access static fields without object initialization.
Finally, we can access static fields using an object reference (such as ford.numberOfCars++). But we should avoid this because it becomes difficult to figure out if it’s an instance variable or a class variable. Instead, we should always refer to static variables using class name (Car.numberOfCars++).
When the value of the variable is independent of objects
When the value is supposed to be shared across all objects

Method:
A static method belongs to the class rather than the object of a class.
A static method can be invoked without the need for creating an instance of a class.
A static method can access static data member and can change the value of it.
static methods in Java are resolved at compile time. Since method overriding is part of Runtime Polymorphism, static methods can't be overridden.
Abstract methods can't be static.
static methods can't use this or super keywords.
To access/manipulate static variables and other static methods that don't depend upon objects
static methods are widely used in utility and helper classes.

What is the ‘this’ reference?
this keyword automatically holds the reference to current instance of a class. It is very useful in scenarios where we are inheriting a method from parent class into child class, and want to invoke method from child class specifically.

What are base class, subclass and superclass?
A class that is derived from another class is called a subclass (also a derived class, extended class, or child class). The class from which the subclass is derived is called a superclass (also a

base class or a parent class).

Draw an object oriented family (as entities, with relations) on the whiteboard.


Difference between overloading and overriding?
When two or more methods in the same class have the same name but different parameters, it’s called Overloading.
When the method signature (name and parameters) are the same in the superclass and the child class, it’s called Overriding.
-Overriding implements Runtime Polymorphism whereas Overloading implements Compile time polymorphism.
-The method Overriding occurs between superclass and subclass. Overloading occurs between the methods in the same class.
-Overriding methods have the same signature i.e. same name and method arguments. Overloaded method names are the same but the parameters are different.
-With Overloading, the method to call is determined at the compile-time. With overriding, the method call is determined at the runtime based on the object type.
-If overriding breaks, it can cause serious issues in our program because the effect will be visible at runtime. Whereas if overloading breaks, the compile-time error will come and it’s easy to fix.

What are the Object Oriented Principles? Explain the concepts with realistic examples!
SOLID principles:
- Single-responsiblity: a class should have only one job.
- Open-closed: entities should be open for extension but closed for modification.
- Liskov Substitution: every derived class should be substitutable for their base class.
- Interface Segregation: clients shouldn’t be forced to depend on methods they do not use.
- Dependency Inversion: entities must depend on abstractions, not on concretions.

abstraction, encapsulation, inheritance, and polymorphism
Abstraction is the process of exposing the essential details of an entity, while ignoring the irrelevant details, to reduce the complexity for the users.
For example, when we drive our car, we do not have to be concerned with the exact internal working of the car. What we are concerned with is interacting with the car via its interfaces like steering wheel, brake pedal, accelerator pedal, etc. Here the knowledge we have of the car is abstract.

Encapsulation is the process of bundling data and operations on the data together in an entity. Wrapping data and methods within classes in combination with implementation hiding (through access control) is often called encapsulation. The result is a data type with characteristics and behaviors.

School bag is one of the most real examples of Encapsulation. School bag can keep our books, pens, etc.

Inheritance derives a new type from an existing class, thereby establishing a parent-child relationship.
For instance, we are humans. We inherit certain properties from the class 'Human' such as the ability to speak, breathe, eat, drink, etc.

Polymorphism lets an entity take on different meanings in different contexts.
Like a man at the same time is a father, a husband, an employee. So the same person possesses different behavior in different situations.

What is method overloading?
If a class has multiple methods having same name but different in parameters, it is known as Method Overloading.

What is method overriding?
Method overriding means subclass had defined an instance method with the same signature and return type as the instance method in the superclass. In such a case, method of the superclass is overridden (replaced) by the subclass.

Explain how object oriented languages attempt to simplify memory management for Programmers.
Garbage collection is the process in which programs try to free up memory space that is no longer used by objects.
Most modern OOP languages use automated garbage collectors, that attempts to reclaim memory which was allocated by the program, but is no longer referenced. This relieves the programmer from performing manual memory management.

Explain the “Single Responsibility” principle!
It is one of the Solid principles of OOP class design. It emphasizes that one class should have one and only one responsibility.
In other words, we should write, change, and maintain a class for only one purpose. This will give us the flexibility to make changes in the future without worrying about the impacts of changes for another entity.

What is an object oriented program? Explain, show.
Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods).
A feature of objects is that an object's own procedures can access and often modify the data fields of itself (objects have a notion of this or self). In OOP, computer programs are designed by making them out of objects that interact with one another. OOP languages are diverse, but the most popular ones are class-based, meaning that objects are instances of classes, which also determine their types.

How do you make a class immutable? What do you need to watch out for?
An immutable class is one whose state can not be changed once created.
Don’t provide “setter” methods — methods that modify fields or objects referred to by fields.
This principle says that for all mutable properties in your class, do not provide setter methods. Setter methods are meant to change the state of an object and this is what we want to prevent here.

Make all fields final and private
This is another way to increase immutability. Fields declared private will not be accessible outside the class and making them final will ensure the even accidentally you can not change them.

Don’t allow subclasses to override methods
The simplest way to do this is to declare the class as final. Final classes in java can not be extended.

Special attention when having mutable instance variables
Always remember that your instance variables will be either mutable or immutable. Identify them and return new objects with copied content for all mutable objects. Immutable variables can be returned safely without extra effort.

How many instances can be created for an abstract class?
0

Programming languages

Java

What is autoboxing and unboxing?
Autoboxing is the automatic conversion of the primitive types into their corresponding wrapper class. For example, you can assign a int value to Interger class reference.

Unboxing happens when the conversion happens from wrapper class to its corresponding primitive type. It means we can pass or assign a wrapper object to an argument or reference accepting primitive type.

If you have a variable, that shall store a positive whole number between 0 and 200, what primitive type would you use to store it?
short

What is the "golden rule" of variable scoping in Java? What is the lifetime of variables?
The golden rule is that static code cannot access non-static members by their simple names. Static code is not executed in the context of an object, therefore the references this and super are not available. An object has knowledge of its class, therefore, static members are always accessible in a non-static context.
Member Variables (Class Level Scope):These variables must be

declared inside class (outside any function). They can be directly accessed anywhere in class. Let’s take a look at an example:
Member variables can be accessed outside a class with rules
Local Variables (Method Level Scope): Variables declared inside a method have method level scope and can’t be accessed outside the method. Local variables don’t exist after method’s execution is over.
Loop Variables (Block Scope): A variable declared inside pair of brackets “{” and “}” in a method has scope within the brackets only.

What is the purpose of the ‘equals()’ method?
In java equals() method is used to compare equality of two Objects. The equality can be compared in two ways:

Shallow comparison: The default implementation of equals method is defined in Java.lang.Object class which simply checks if two Object references (say x and y) refer to the same Object. i.e. It checks if x == y. Since Object class has no data members that define its state, it is also known as shallow comparison.
Deep Comparison: Suppose a class provides its own implementation of equals() method in order to compare the Objects of that class w.r.t state of the Objects. That means data members (i.e. fields) of Objects are to be compared with one another. Such Comparison based on data members is known as deep comparison.

What is the difference between '==' and 'equals()'?
The main difference between the .equals() method and == operator is that one is a method, and the other is the operator.
We can use == operators for reference comparison (address comparison) and .equals() method for content comparison. In simple words, == checks if both objects point to the same memory location whereas .equals() evaluates to the comparison of values in the objects.
If a class does not override the equals method, then by default, it uses the equals(Object o) method of the closest parent class that has overridden this method. See Why to Override equals(Object) and hashCode() method ? in detail.

What does the ‘static’ keyword mean?
The static keyword in Java is mainly used for memory management. The static keyword in Java is used to share the same variable or method of a given class. The users can apply static keywords with variables, methods, blocks, and nested classes. The static keyword belongs to the class than an instance of the class. The static keyword is used for a constant variable or a method that is the same for every instance of a class.
When a member is declared static, it can be accessed before any objects of its class are created, and without reference to any object.

Why is the main() method declared as static? Explain.
To understand this, let suppose we do not have the main method as static. Now, to invoke any method you need an instance of it. Java can have overloaded constructors, we all know. Now, which one

should be used, and from where the parameters for overloaded constructors will come. These are just more difficult questions, which helped Java designers to make up their minds and to have the main method as static.
Ans) It is because the object is not required to call a static method. If it were a non-static method, JVM creates an object first then call main() method that will lead the problem of extra memory allocation.

What is the default access modifier in a class?
Package private - access in the same package

What is the JVM?
Java Virtual machine (JVM) is the virtual machine that runs the Java bytecodes. You get this bytecode by compiling the .java files into .class files. .class files contain the bytecodes understood by the JVM.The JVM is called virtual because it provides a machine interface that does not depend on the underlying operating system and machine hardware architecture.

What is the difference between the JRE and the JDK?
JRE = JVM + libraries to run Java application.
The Java Runtime Environment (JRE) is a software package which bundles the libraries (jars) and the Java Virtual Machine, and other components to run applications written in the Java. JVM is just a part of JRE distributions.

JDK = JRE + tools to develop Java Application.
JDK is a superset of JRE. JDK contains everything that JRE has along with development tools for developing, debugging, and monitoring Java applications. You need JDK when you need to develop Java applications.

What is the difference between long and Long?
long - primitive type
Long - wrapper class

Can a long store bigger numbers than a Long?
no,
Long wrapper class' field:
static long MAX_VALUE
A constant holding the maximum value a long can have, 2^(63)-1.

What kind of packages do you know under java.util.* ? Bring at least 3 examples.
ArrayList, Arrays, List, HashSet...

What are the access modifiers in Java? Which one could we use for class?
Private: The access level of a private modifier is only within the class. It cannot be accessed from outside the class.
Default: The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do

not specify any access level, it will be the default.
Protected: The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.
Public: The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.
Default and public.

Can an “enum” contain methods in Java? Explain.
Remember that enum is basically a special class type, and can have methods and fields just like any other class. You can add methods which are abstract as well as concrete methods as well. Both methods are allowed in enum.

When would you use a private/protected/public attribute? What is the difference?
If the class member declared as public then it can be accessed everywhere.
If the class members declared as protected then it can be accessed only within the class itself and by inheriting child classes.
If the class members declared as private then it may only be accessed by the class that defines the member.

How do you prevent developers from subclassing a class?
final/static class

How do you prevent developers from overriding a method in a subclass?
final/static method

How do you prevent developers from changing the value of a variable?
final (static) variable

Think about money ;) How would you store a currency value, that shall support decimal parts? Think it through again, and try to think outside of the box!
Bigdecimal

What happens if you try to call something, that you have no access to, because of data hiding?
Cannot call. "Cannot resolve symbol ."

What happens if you try to delete/drop an item from an array, while you are iterating over it?
Java doesn't have a direct method for deleting array element. So in a custom implementation we must be wary, not to run into an IndexOutOfBoundsException.

What happens if you try to delete/drop/add an item from a List, while you are iterating over it?
We can do all those in the basic for loop with an iterating variable

without trouble, however modifying the list while using a forEach loop causes
ConcurrentModificationException.

What happens if you try to add an item to the end of an array, while you are iterating over it?
The loop won't consider the newly added element.

If you need to access the iterator variable after a for loop, how would you do it?
```
int x;
for (x = 0; x < 4; x++){
    System.out.println(x);
}

System.out.println(x);
```

Which interfaces extend the Collection interface in Java. Which classes?
Set, List and Queue interfaces extend it. Then there are many other classes in these three branches.
SortedSet, HashSet, TreeSet, ArrayList, LinkedList,

What is the connection between equals() and hashCode()? How are they used in HashMap?
hashCode() and equals() methods have been defined in Object class which is parent class for all java classes. For this reason, all java objects inherit a default implementation of these methods.
If two objects are equal according to the equals() method, then calling the hashCode() on each of the two objects must produce the same integer result.

While retrieving the value by key, first index location is found using key’s hashcode. Then all elements are iterated in the linkedlist and correct value object is found by identifying the correct key using it’s equals() method.

What is the difference between checked exceptions and unchecked exceptions? Could you bring example for each?
Java checked exceptions are those exceptions, as the name suggests, which a method must handle in its body or throw to the caller method so the caller method can handle it.
Checked exceptions are checked by the Java compiler so they are called compile time exceptions.
Java compiler forces us to handle these exceptions in some manner in the application code. We must handle these exceptions at a suitable level inside the application so that we can inform the user about the failure and ask him to retry or come later.
Note that all checked exceptions are subclasses of Exception class.
For example,
ClassNotFoundException
IOException
SQLException

Unchecked exceptions are not checked by the compiler. These are called runtime exceptions.
Unchecked exceptions will come into life and occur in the program, once any buggy code is executed.
In Java, a member method is not forced by the compiler to declare the unchecked exceptions into the method declaration. Generally, such methods almost always do not declare them.
Unchecked Exceptions are subclasses of RuntimeException class.

NullPointerException
ArithmeticException
ArrayStoreException
ClassCastException

What is Error in Java and how does it relate to Exception?
In java, both Errors and Exceptions are the subclasses of java.lang.Throwable class. Error refers to an illegal operation performed by the user which results in the abnormal working of the program. Programming errors often remain undetected until the program is compiled or executed. Some of the errors inhibit the program from getting compiled or executed. Thus errors should be removed before compiling and executing. It is of three types:
Compile-time
Run-time
Logical
Whereas exceptions in java refer to an unwanted or unexpected event, which occurs during the execution of a program i.e at run time, that disrupts the normal flow of the program’s instructions.
Error Exception
Recovering from Error is not possible. We can recover from exceptions by either using try-catch block or throwing exceptions back to the caller.
All errors in java are unchecked type. Exceptions include both checked as well as unchecked type.
Errors are mostly caused by the environment in which program is running. Program itself is responsible for causing exceptions.
Errors can occur at compile time as well as run time. Compile Time: eg Syntax Error Run Time: Logical Error.
All exceptions occurs at runtime but checked exceptions are known to the compiler while unchecked are not.

When does 'finally' block run? What it is used for? Could you give an example from your projects when you would use 'finally'?
An optional finally block gives us a chance to run the code which we want to execute EVERYTIME a try-catch block is completed – either with errors or without any error.
The finally block statements are guaranteed of execution even if the we fail to handle the exception successfully in catch block.
Example: close file

What is the largest number you can work with in Java?
The int primary type has a max of 2,147,483,647 and long can go till 9,223,372,036,854,775,807.
However by importing the BigInteger class, the largest number

depends on how much RAM you have.

When you use method overriding, can you change the access level of the method, from protected to public? Why?When you use method overriding, can you change the access level of the method, from public to protected? Why?
An overridden method can have a different access modifier but it cannot lower the access scope. That means methods declared protected in a superclass must either be protected or public in subclasses and methods declared public in a superclass also must be public in all subclasses.

Can the main method be overridden? Explain your answer!
No, we cannot override main method of java because a static method cannot be overridden.
The static method in java is associated with class whereas the non-static method is associated with an object. Static belongs to the class area, static methods don't need an object to be called. Static methods can be called directly by using the classname (classname.static_method_name()).
So, whenever we try to execute the derived class static method, it will automatically execute the base class static method.
Therefore, it is not possible to override the main method in java.

When you use method overriding, can you throw fewer exceptions in the subclass than in the parent class? Why?
Yes, in accordance with the Liskov Substitution Principle.

When you use method overriding, can you throw more exceptions in the subclass than in the parent class? Why?
No, in accordance with the Liskov Substitution Principle.

What does "final" mean in case of a variable, method or a class?
final keyword is used in different contexts. First of all, final is a non-access modifier applicable only to a variable, a method or a class. Following are different contexts where final is used.
variable - to create a constant variable
method - to prevent method overriding
class - to prevent inheritance

What is the super keyword?
It always hold the reference to parent class of any given class. Using super keyword, we can access the fields and methods of parent class in any child class.

What are "generics"? When to use? Show examples.
Generics means parameterized types. The idea is to allow type (Integer, String, … etc., and user-defined types) to be a parameter to methods, classes, and interfaces. Using Generics, it is possible to create classes that work with different data types. An entity such as class, interface, or method that operates on a parameterized type is a generic entity.
The Object is the superclass of all other classes, and Object

reference can refer to any object. These features lack type safety. Generics add that type of safety feature.

What is the benefit of having “generic” containers?
Code that uses generics has many benefits over non-generic code:

Stronger type checks at compile time.
A Java compiler applies strong type checking to generic code and issues errors if the code violates type safety. Fixing compile-time errors is easier than fixing runtime errors, which can be difficult to find.

Elimination of casts.
    The following code snippet without generics requires casting:
    List list = new ArrayList();
    list.add("hello");
    String s = (String) list.get(0);
    When re-written to use generics, the code does not require casting:
    List<String> list = new ArrayList<String>();
    list.add("hello");
    String s = list.get(0);   // no cast
Enabling programmers to implement generic algorithms.
By using generics, programmers can implement generic algorithms that work on collections of different types, can be customized, and are type safe and easier to read.

Given two Java programs on two different machines. How can you communicate between the two? What are the possible ways?
- If you need only inter-process synchronous messaging (remote procedure call), RMI should be easiest.

- If you need asynchronous inter-process messaging, JMS should be easiest.

- If you need inter-process shared memory, use mapped files.

- If you need all the above, Terracotta is the easiest way: Java programs on different JVMs on the same or even different computers see each other as if they were executed inside one JVM on one machine.

What is an annotation? What can be annotated and how? Show examples.
Annotations are used to provide supplemental information about a program. They start with ‘@’.
Classes, methods, variables, parameters and Java packages can all be annotated.
Two annotations I commonly use are:
@Override - to mark that a certain method is overriding another
@Test - to mark that the current method is a junit test method

C#

Explain the purpose of IL and how does it relate to CLR?
What does “managed code” mean?
What is an assembly?
What is the difference between an EXE and a DLL?
What is the difference between `dotnet build` and `dotnet restore`?
What is strong-typing versus weak-typing? Which is preferred? Why?
What is a namespace?
Explain sealed class in C#?
What is explicit vs. implicit conversion? Give examples of both of them.
Is a struct stored on the heap or stack?
Can a struct have methods?
Can DateTimes be null?
List out the differences between Array and ArrayList in C#?
How is the using() pattern useful? What is IDisposable? How does it support deterministic finalization?
How can you make sure that objects using dedicated resources (database connection, files, hardware, OS handle, etc.) are released as early as possible?
Why to use keyword “const” in C#? Give an example.
What is the difference between “const” and “readonly” variables in C#?
What is a property in C#?
List out two different types of errors in C#?
What is the difference between “out” and “ref” parameters in C#?
Can we override private virtual method in C#?
What's the difference between IEquatable and just overriding Object.Equals()?
Explain the differences between public, protected, private and internal. Explain access modifier – “protected internal” in C#!
What’s the difference between using `override` and `new` keywords when defining method in child class?
Explain StringBuilder class in C#!
How we can sort the array elements in descending order in C#?
Can you use a value type as a generic type argument in C#? For example when implementing an interface like (IEquatable).
What are Nullable Types in C#?
Conceptually, what is the difference between early-binding and late-binding?
What is delegate, event, callback, multicast delegate?
What is enum in C#?
What is null-conditional operator?
What is null-coalescing operator?
What is serialization?
What is the difference between Finalize() and Dispose() methods?
How do you inherit a class from another class in C#?
What is difference between “is” and “as” operators in C#?
What are indexers in C# .NET?
What is the difference between returning IQueryable<T> vs. IEnumerable<T>?

What is LINQ? Explain the idea of extension methods.
What are the advantages and disadvantages of lazy loading?
How to use of “yield” keyword? Mention at least one practical scenario where it can be used?
What are attributes in C#? Give some examples of usage of them.
By what mechanism does NUnit know what methods to test?
What is the GAC? What problem does it solve?
What is the largest number you can work with in C#?