Programming Fundamental Knowledge

Java related knowledge

- What is strongly typed?

> A strongly typed language has stricter typing rules at compile time, which implies that errors and exceptions are more likely to happen during compilation.

- What is the advantage of strongly type? Disadvantage?

> 1. Advantages:
> 2. Earlier detection of errors in compiling, speeds development
> 3. Type helps team members understand code easier
> 4. The posed type restriction can avoid code confusions (in JS, we can add string to number "2"+3=5, which could cause bugs)
> 5. Disadvantage: Not much, if really want to list some disadvantage: since strongly typed need more code for specification, it might affect early stage of development speed, (but in big project, strongly type will speed up development)

- What is difference between run-time error and compiling error. Example for each.

> A run time error will only occur when the code is actually running. These are the most difficult, can be hard to track down.
>
> 1. Example of compiling error (checked)
> 2. Syntax Error (mismatch braces/brackets; missing semi-colon)
> 3. Type mismatch
> 4. Missing Return Statement
> 5. Example of run time error (unchecked)
> 6. Incorrect comparison operators (e.g., not using double equal signs to indicate assignment)
> 7. Referencing objects that don't exist, or don't exist using the capitalization supplied in the code
> 8. Referencing an object that has no properties

- What is Data Abstract in Java, how can it be achieved?

> Data abstraction is the process of hiding certain details and showing only essential information to the user. In Java, Abstraction can be achieved with either abstract classes or interfaces

- What is difference between interface and abstract class?

> 1. Type of methods: Interface can have only abstract methods. Abstract class can have abstract and non-abstract methods. From Java 8, it can have default and static methods also.
> 2. Final Variables: Variables declared in a Java interface are by default final. An abstract class may contain non-final variables.

> 3. Type of variables: Abstract class can have final, non-final, static and non-static variables. Interface has only static and final variables.
> 4. Implementation: Abstract class can provide the implementation of interface. Interface can't provide the implementation of abstract class.
> 5. Inheritance vs Abstraction: A Java interface can be implemented using keyword "implements" and abstract class can be extended using keyword "extends".
> 6. Multiple implementation: An interface can extend another Java interface only, an abstract class can extend another Java class and implement multiple Java interfaces.
> 7. Accessibility of Data Members: Members of a Java interface are public by default. A Java abstract class can have class members like private, protected, etc.

- Which is better? Interface or subclassing?

> (Arguably) Interface is generally considered to be better
>
> 1. More flexible, since a class can extends only 1 class, but can implement multiple interfaces
> 2. Separate the implementation and abstraction

- Different between Array and ArrayList

> 1. Implementation of array is simple fixed sized array but Implementation of ArrayList is dynamic sized array.
> 2. Array can contain both primitives and objects but ArrayList can contain only object elements
> 3. You can't use generics along with array but ArrayList allows us to use generics to ensure type safety.
> 4. You can use length variable to calculate length of an array but size() method to calculate size of ArrayList.
> 5. Array use assignment operator to store elements but ArrayList use add() to insert elements.

- Priority Queue in Java (Very important for Heap data structure)

> 1. A PriorityQueue is used when the objects are supposed to be processed based on the priority. It is known that a queue follows First-In-First-Out algorithm, but sometimes the elements of the queue are needed to be processed according to the priority, that's when the PriorityQueue comes into play.
> 2. Make sure to know the **Exact Syntax** of priority queue to implement Heap Data Structure

- try/catch/finally

> 1. The finally block always executes when the try block exits. This ensures that the finally block is executed even if an unexpected exception occurs
> 2. The finally block is a key tool for preventing resource leaks. When closing a file or otherwise recovering resources, place the code in a finally block to ensure that resource is always recovered.

- final keyword

> In Java the final keyword is used in several contexts to **define an entity that can only be assigned once**, examples:
>
> 1. A final class cannot be subclassed.

> 2. A final method cannot be overridden or hidden by subclasses.
> 3. A final variable can only be initialized once, either via an initializer or an assignment statement.

- Mutable/Immutable data structure

> 1. Mutable object – You can change the states and fields after the object is created. For examples: StringBuilder, java.util.Date and etc.
> 2. Immutable object – You cannot change anything after the object is created. For examples: String, boxed primitive objects like Integer, Long and etc.

- Java Enum:

> An enum is a special "class" that represents a group of constants (unchangeable variables, like final variables).
>
> ```
> enum Level {
> LOW,
> MEDIUM,
> HIGH
> }
> ```
>
> You can also have an enum inside a class:
>
> ```
> public class MyClass {
> enum Level {
>   LOW,
>   MEDIUM,
>   HIGH
> }
> public static void main(String[] args) {
>     Level myVar = Level.MEDIUM;
>     System.out.println(myVar);
>     }
> }
> ```

- List all the collection types you know

> **At least remember first five, and Priority Queue**
>
> 1. Array
> 2. ArrayList
> 3. LinkedList
> 4. HashMap
> 5. Hashtable
> 6. LinkedHashMap
> 7. TreeMap
> 8. HashSet

> 9. LinkedHashSet
> 10. TreeSet
> 11. Priority Queue

- Overriding vs Overloading

> 1. Overloading occurs when two or more methods in one class have the same method name but different parameters.
> 2. Overriding means having two methods with the same method name and parameters (i.e., method signature). One of the methods is in the parent class and the other is in the child class. Overriding allows a child class to provide a specific implementation of a method that is already provided its parent class.

- public, protected, private (& package)

> 1. If the class member declared as public then it can be accessed everywhere.
> 2. If the class members declared as protected then it can be accessed only within the class itself and by inheriting and parent classes.
> 3. If the class members declared as private then it may only be accessed by the class that defines the member.

- What will happen if we do not provide constructor for Java class?

> Java does not actually require an explicit constructor in the class description. If you do not include a constructor, the Java compiler will create a default constructor in the byte code with an empty argument.

- How to manage memory in Java? What could cause memory leak?

> Java itself manages the memory and needs no explicit intervention of the programmer. **Garbage Collector** itself ensures that the unused space gets cleaned and memory can be freed when not needed. (Not all the language have garbage collector, like C)

- this() and super()

> 1. super() as well as this() both are used to make constructor calls.
> 2. super() is used to call Base class's constructor(i.e, Parent's class) while this() is used to call current class's constructor.

- How to clone an object and Java? Clone array, ArrayList, set etc.

> You can manually create a new object and copy each field, but you can also use clone() method

- What is the default value of the local variables?

> 1. The local variables are not initialized to any default values.
> 2. We should not use local variables with out initialization.

- Is everything in Java an Object?

> Every object is a java.lang.Object (NOTE:java.lang.Object has no super class.) However, there are many things which are not Objects.

> 1. primitives and references. **You can ignore all the other things, and only remember primitive**
> 2. fields (the fields themselves not the contents)
> 3. local variables and parameters.

- How many Primitive Types does Java have

> 1. byte 1 byte
> 2. short 2 bytes
> 3. int 4 bytes
> 4. long 8 bytes
> 5. float 4 bytes
> 6. double 8 bytes
> 7. boolean 1 bit
> 8. char 2 bytes

- Classic OOP concepts

> I'll skip the examples here, but want to emphasis that: need to make sure fully understand all of the following concept, can write code example within 5 minutes:
>
> 1. Inheritance
> 2. Encapsulation
> 3. Polymorphism
> 4. Abstraction

- What is Singleton

> In object-oriented programming, a singleton class is a class that can have only one object (an instance of the class) at a time.

Web Development knowledge

- What is server, what is client

> 1. Client–server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. Server is **the providers of a resource or service** Client is **service requesters**, including Web page in web browser, iPhone/Android App, or Mac/Windows clients Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system.

- RESTful

> 1. Representational state transfer (REST) is a software architectural style that defines a set of constraints to be used for creating Web services.
> 2. It is used for separate server/client, frontend/backend by providing specs how they communicate

- What is JSON?

> 1. JSON stands for JavaScript Object Notation
> 2. JSON is a lightweight format for storing and transporting data
> 3. JSON is often used when data is sent from a server to a web page
> 4. JSON is "self-describing" and easy to understand

- What is serialization/de-serialization?

> 1. Serialization is a mechanism of converting the state of an object into a byte stream.
> 2. Deserialization is the reverse process where the byte stream is used to recreate the actual Java object in memory.
> 3. This mechanism is used to persist the object.

- What is cookie and session?

> 1. A cookie is a bit of data stored by the browser and sent to the server with every request.
> 2. A session is a collection of data stored on the server and associated with a given user (usually via a cookie containing an id code)

- What is API

> An application programming interface (API) is an interface or communication protocol between different parts of a computer program intended to simplify the implementation and maintenance of software.