Agenda

- Array Revision
- Variable Arity/Argument Method
- Object/Field Initializer
- static Keyword
- Singleton Design Pattern
- ~~Association~~

Variable Arity/Argument Method

- It is a method which can take variable no of arguments.
- We can also pass array to this method.
- If we want to pass different types of variables to this arity method then we can use the object as the type.

```
public static int sum(int... arr) {
int total = 0;
for(int num: arr)
        total = total + num;
return total;
}

public static void main(String[] args) {
int result1 = sum(10, 20);
System.out.println("Result: " + result1);
int result2 = sum(11, 22, 33);
System.out.println("Result: " + result2);
}
```


Object/Field Initializer

- In C++/Java Fields of the class are initialized using constructor
- In java, field can also be initialized using
    - 1. field initializer
    - 2. object initializer
    - 3. Constructor

static Keyword

- In OOP, static means "shared" i.e. static members belong to the class (not object) and shared by all objects of the class.
- Static members are called as "class members"; whereas non-static members are called as "instance memebers".
- In Java, static keyword is used for
    - 1. static fields
    - 2. static methods

    - 3. static block
    - 4. static import
- Note that, static local variables cannot be created in Java.

1. static Fields

- Copies of non-static/instance fields are created one for each object.
- Single copy of the static/class field is created (in method area) and is shared by all objects of the class.
- Can be initialized by static field initializer or static block.
- Accessible in static as well as non-static methods of the class.
- Can be accessed by class name or object name outside the class (if not private). However, accessing via object name is misleading (avoid it).
- eg :
    - Integer.SIZE
- Similar to field initializer, static fields can be initialized at declaration.

2. Static methods

- These Methods can be called from outside the class (if not private) using class name or object name. However, accessing via object name is misleading (avoid it).
- When we need to call a method without creating object, then make such methods as static.
- Since static methods are designed to be called on class name, they do not have "this" reference. Hence, they cannot access non-static members in the static method (directly), However, we can access them on an object reference if created inside them.
- eg:
    - Integer.valueOf(10);
    - Factory Methods -> to cretae object of the class

static Field Initializer

- Similar to field initializer, static fields can be initialized at declaration.

```
static double roi = 5000.0;
// static final field -- constant
static final double PI = 3.142;
```

static Initializer Block

- Like Object/Instance initializer block, a class can have any number of static initialization blocks, and they can appear anywhere in the class body.
- Static initialization blocks are executed in the order their declaration in the class.
- A static block is executed only once when a class is loaded in JVM.

static import

- To access static members of a class in the same class, the "ClassName." is optional.
- To access static members of another class, the "ClassName." is mandetory.

- If need to access static members of other class frequently, use "import static" so that we can access static members of other class direcly (without ClassName.).

Singleton Design Pattern

- Singleton is a design pattern.
- Singleton class is a class whose single object is created throughout the application.
- To make a singleton class in Java
- step 1: Write a class with desired fields and methods.
- step 2: Make constructor(s) private.
- step 3: Add a private static field to hold instance of the class.
- step 4: Initialize the field to single object using static field initializer or static block.
- step 5: Add a public static method to return the object.