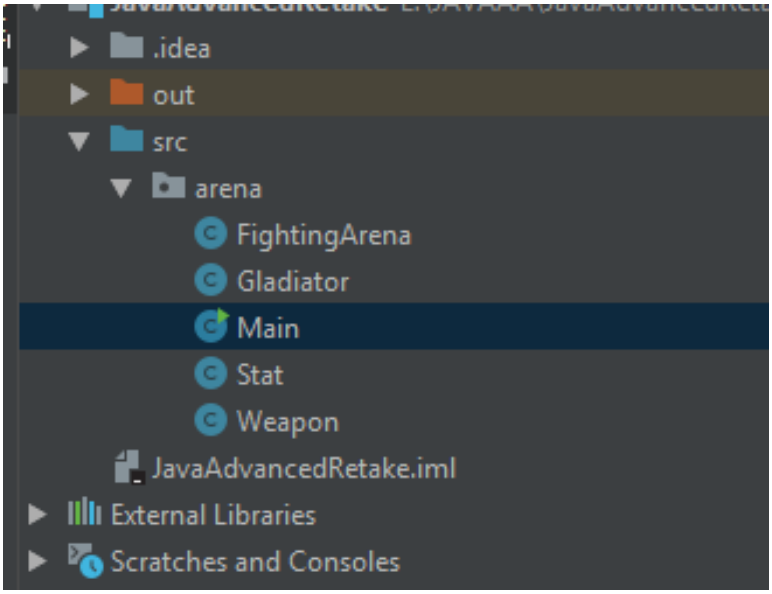
Problem 3. Fighting Arena

I. Project Structure

For this problem you should create a new package named **"arena",** which should hold inside the classes **Gladiator , Stat, Weapon and FightingArena.** The Main class can also be inside this package however it is not a must it may also be outside the package. Your project structure should look like that:



Pay attention to name the package, all the classes, their fields and methods exactly the same way they are presented in the following document. It is also important to keep the project structure as described above.

II. Weapon

Create Java class **`Weapon that has the following structure:`**

```
public class Weapon {
    // TODO: implement this class
}
```

1. Fields

- **`size: int`**
- **`solidity: int`**
- **`sharpness: int`**

The class **constructor** should receive all the fields parameters (**size, solidity, sharpness**).

2. Methods:

- Getter **getSize()**
- Getter **getSolidity()**
- Getter **getSharpness()**

III. Stat

Create Java class **`Stat that has the following structure:`**

```
public class Stat {
    // TODO: implement this class
}
```

3. Fields

- **`strength: int`**
- **`flexibility: int`**
- **`agility: int`**
- **`skills: int`**
- **`intelligence: int`**

The class **constructor** should receive all the fields parameters (**strength, flexibility, agility, skills, intelligence**).

4. Methods:

- Getter **getStrength()**
- Getter **getFlexibility()**
- Getter **getAgility()**
- Getter **getSkills()**
- Getter **getIntelligence()**

IV. Gladiator

Create Java class **`Gladiator that has the following structure:`**

```
public class Gladiator {
    // TODO: implement this class
}
```

1. Fields

- **name: String**
- **stat: Stat**
- **weapon: Weapon**

The class **constructor** should receive all the fields parameters (**name, stat, weapon**).

2. Methods:

- Getter **getName()**
- Getter **getStatPower() : int –** return the sum of the stat properties
- Getter **getWeaponPower(): int -** return the sum of the weapon properties.
- Getter **getTotalPower(): int -** return the sum of the stat properties plus the sum of the weapon properties
- Method **toString()** which returns the information about a single Gladiator object in the following format:

```
"{gladiatorName} - {gladiatorTotalPower}"
"  Weapon Power: {gladiatorWeaponPower}"
"  Stat Power: {gladiatorStatPower}"
```

V. FightingArena

Write a Java class **FightingArena** that has **gladiators** (a collection which stores the entity **Gladiator**). All entities inside the arena have the **same properties**.

```
class FightingArena {
    // TODO: implement this class
}
```

1. Fields

- Field **gladiators** – **collection** that holds added entities
- Field **name - String**

The class **constructor** should initialize the **gladiators** with a new instance of the collection and should receive the name field **(name)**.

2. Methods:

- Method **add(entity)** – adds an entity to the Data
- Method **remove(name)** – removes an entity by given Gladiator name.
- Method **getGladiatorWithHighestStatPower()** – returns the Gladiator which poses the Stat with the highest stat power.
- Method **getGladiatorWithHighestWeaponPower()** – returns the Gladiator which poses the Weapon with the highest weapon power.
- Method **getGladiatorWithHighestTotalPower()** – returns the Gladiator who has the highest total power.
- Getter **getCount** – returns the number of stored entities
- Override **toString()** in following format:

```
"{arenaName} - {countOfGladiators} gladiators are participating."
```

Examples

This is an example how the **FightingArena** class is **intended to be used**.

Sample code usage

```
//Creates fightingArena
FightingArena fightingArena = new FightingArena("Armeec");

//Creates stats
Stat firstGlariatorStat = new Stat(20, 25, 35, 14, 48);
Stat secondGlariatorStat = new Stat(40, 40, 40, 40, 40);
Stat thirdGlariatorStat = new Stat(20, 25, 35, 14, 48);

//Creates weapons
Weapon firstGlariatorWeapon = new Weapon(5, 28, 100);
Weapon secondGlariatorWeapon = new Weapon(5, 28, 100);
Weapon thirdGlariatorWeapon = new Weapon(50, 50, 50);

//Creates gladiators
Gladiator firstGladiator = new Gladiator("Stoyan", firstGlariatorStat, firstGlariatorWeapon);
Gladiator secondGladiator = new Gladiator("Pesho", secondGlariatorStat,
secondGlariatorWeapon);
Gladiator thirdGladiator = new Gladiator("Author", thirdGlariatorStat, thirdGlariatorWeapon);

//Adds gladiators to fightingArena
fightingArena.add(firstGladiator);
fightingArena.add(secondGladiator);
fightingArena.add(thirdGladiator);

//Prints gladiators count at the fightingArena
System.out.println(fightingArena.getCount());

//Gets strongest gladiator and print him
Gladiator strongestGladiator = fightingArena.getGladiatorWithHighestTotalPower();
System.out.println(strongestGladiator);

//Gets gladiator with the strongest weapon and print him
Gladiator bestWeaponGladiator = fightingArena.getGladiatorWithHighestWeaponPower();
System.out.println(bestWeaponGladiator);

//Gets gladiator with the strongest stat and print him
Gladiator bestStatGladiator = fightingArena.getGladiatorWithHighestStat();
System.out.println(bestStatGladiator);

//Removes gladiator
fightingArena.remove("Author");

//Prints gladiators count at the fightingArena
System.out.println(fightingArena.getCount());

//Prints the fightingArena
System.out.println(fightingArena);
```

Constraints

- The names of the Gladiators will be always unique.
- The items of the Gladiators will always be with positive values.
- The items of the Gladiators will always be different.
- You will always have a Gladiator with the highest weapon power, stat power, and total power.

Submission

Submit **single .zip file**, containing "**arena" package, with the four classes inside (Weapon, Stat, Gladiator and FightingArena) and the Main class**, there is no specific content required inside the Main class e. g. you can do any kind of local testing of you program there. However there should be **main(String[] args)** method inside: