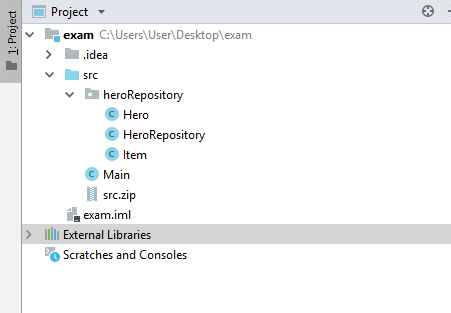
# Problem 3. HeroRepository

## Project Structure

For this problem you should create a new package named **"heroRepository",** which should hold inside the classes **Item** **, Hero and Hero**Repository**.** 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.**

## Item

Create Java class Item that has the following structure:

**public class** Item {  
 *//* ***TODO: implement this class***}

### Fields

* **strength: int**

# agility: int

* **intelligence: int**

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

### Methods:

* Getter **getStrength()**
* Getter **getAgility()**
* Getter **getIntelligence()**
* Method **toString()** which returns the information about a single Item object in the following format:

**"Item:"**

**" \* Strength: {Strength Value}"**

**" \* Agility: {Agility Value}"**

**" \* Intelligence: {Intelligence Value}"**

## Hero

Create Java class Hero that has the following structure:

**public class** Hero {  
 *//* ***TODO: implement this class***}

### Fields

* **name: Strinig**
* **level: int**
* **item: Item**

The class **constructor** should receive all the fields parameters (**name, level, item**).

### Methods:

* Getter **getName()**
* Getter **getLevel()**
* Getter **getItem()**
* Method **toString()** which returns the information about a single Hero object in the following format:

**"Hero: {Name} – {Level}"**

**" \* Strength: {Strength Value}"**

**" \* Agility: {Agility Value}"**

**" \* Intelligence: {Intelligence Value}"**

## Hero**Repository**

Write a Java class **Hero**Repository that has **data** (a collection which stores the entity **Hero**). All entities inside the repository have the **same properties**.

**class** HeroRepository {  
 *//* ***TODO: implement this class***}

### Fields

* Field **data** – **collection** that holds added entities

The class **constructor** should initialize the **data** with a new instance of the collection**.**

1. **Methods:**

* Method add(entity) – adds an entity to the Data
* Method remove(name) – removes an entity by given hero name.
* Method **getHeroWithHighestStrength()** – returns the Hero witch poses the item with the highest strength
* Method **getHeroWithHighestAgility()** – returns the Hero witch poses the item with the highest agility
* Method **getHeroWithHighestIntelligence()** – returns the Hero witch poses the item with the highest intelligence
* Getter getCount – returns the number of stored entities
* Оverride **toString()** – Print all the heroes.

### Examples

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

|  |
| --- |
| Sample code usage |
| //Initialize the repository  HeroRepository repository = new HeroRepository();  //Initialize entity  Item item = new Item(23, 35, 48);  //Print Item  System.out.println(item);  //Item:  // \* Strength: 23  // \* Agility: 35  // \* Intelligence: 48  //Initialize entity  Hero hero = new Hero("Hero Name", 24, item);  //Print Hero  System.out.println(hero);  //Hero: Hero Name – 24  // \* Strength: 23  // \* Agility: 35  // \* Intelligence: 48  //Add Hero  repository.add(hero);  //Remove Hero  repository.remove("Hero Name");  Item secondItem = new Item(100, 20, 13);  Hero secondHero = new Hero("Second Hero Name", 125, secondItem);  //Add Heroes  repository.add(hero);  repository.add(secondHero);  Hero heroStrength = repository.getHeroWithHighestStrength(); //returns secondHero  Hero heroAbility = repository.getHeroWithHighestAgility(); //returns hero  Hero heroIntelligence = repository.getHeroWithHighestIntelligence(); //returns hero  System.out.println(repository);  //Hero: Hero Name – 24  // \* Strength: 23  // \* Agility: 35  // \* Intelligence: 48  //Hero: Second Hero Name – 125  // \* Strength: 100  // \* Agility: 20  // \* Intelligence: 13 |

### Constraints

* The names of the heroes will be always unique.
* The items of the heroes will always be with positive values.
* The items of the heroes will always be different.
* You will always have an item with the highest strength, agility and intelligence.

### Submission

Submit **single .zip file**, containing **heroRepository package, with the three classes inside (Item, Hero and HeroRepository) 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: