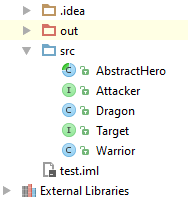
# Lab: Object Communication and Events

Problems for exercises and homework for the ["Java OOP" course @ SoftUni](https://softuni.bg/trainings/2245/java-oop-february-2019).

# Part I: Command Design Pattern

## Resources

You are given a file with some classes. Place them in a new project and get familiar with them.



## Command

Create a **Command Pattern** Executor and provide:

* interface Command
  + void execute()
* interface Executor
  + void executeCommand(Command command)
* Concrete Executor named CommandExecutor implements Executor
* Concrete Commands
  + TargetCommand with constructor (Attacker, Target)
  + AttackCommand with constructor (Attacker)

### Hints

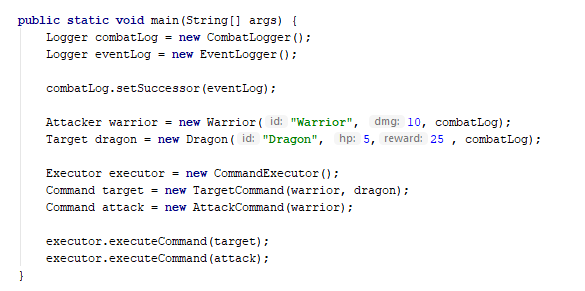
Create the interfaces

Each new command should implement Command, so it can be executed by the Executor



Create as many commands as you like

Test your commands



# Part II: Observer Design Pattern

## Observer

Implement the **Observer Design Pattern** by providing the following:

* interface Subject
  + void register(Observer)
  + void unregister(Observer)
  + void notifyObservers()
* interface Observer
  + update(int)

If a **Target** dies, it should **send reward** to all of its **Observers**

### Hints

Create the interfaces

Attacker should be the Observer

\* Dragon should be the Subject - (the easiest way is to make Target extends Subject, but this is violation of the **Interface Segregation Principle**). The better solution is to create a new interface ObservableTarget and implement both Target and Observer.