### Battleground – workshop

Overview

-In this workshop we shall create a console game application called "BATTLEGROUND".

BattleGround size: 5x5.

UnitTypes: Marine and Carrier.

**Commands**

* spawn {unitType} {name} {coordinateX} {coordinateY}
* add an unit in the battleground
* status {name}
* give information about the unit in this name.
* move {name} {coordinateX} {coordinateY}
* move the unit to a new position on the field.
* fight

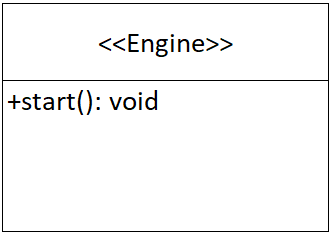
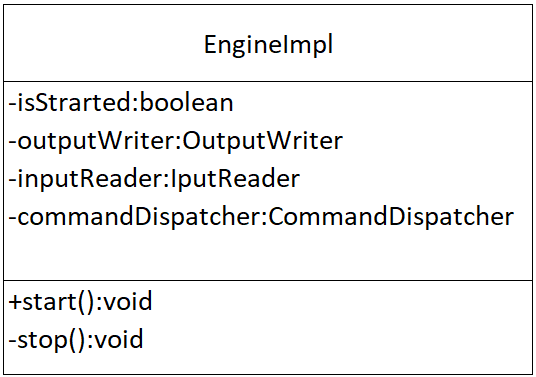
- every unit hits someone (if there are some whithin it's range). Their ranges are static final variables and are described in classes Marine and Carrier.

- unit can hit only one opponent (within its range)

- when an unit chooses the target he picks one with lower health points. If on battleground have more than one unit with same lower points he compare them by name.

* game-over
* the game is over and you should print "Game Over!"

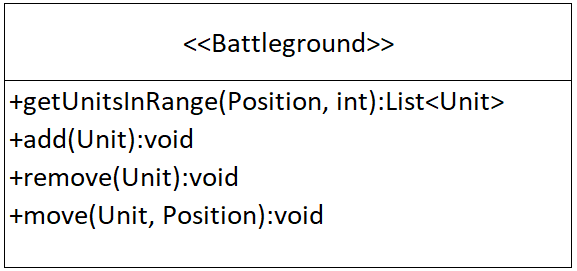
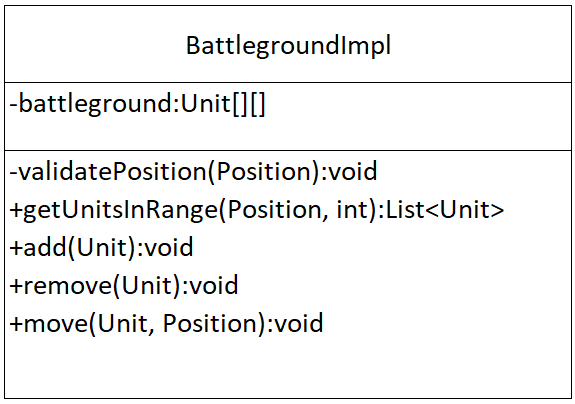
### Engine

### Battleground

Battleground has a fixed size: 5x5.

Throws InvalidPositionException in add(), remove() and move() in interface and in BattlegroundImpl.

### Unit

Each Unit has Type, Name and Coordinates of the Battleground.

Depending on the Type (Marine or Carrier), their qualities are:

**Marine:**

-RANGE = 1

-HEALTH\_POINTS = 50

-ENERGY\_POINTS = 80

-ATTACK\_POINTS = 15

-DEFENCE\_POINTS = 5

Marine spell: RageShoot => ENERGY\_COST = 10

**Carrier:**

-RANGE = 2

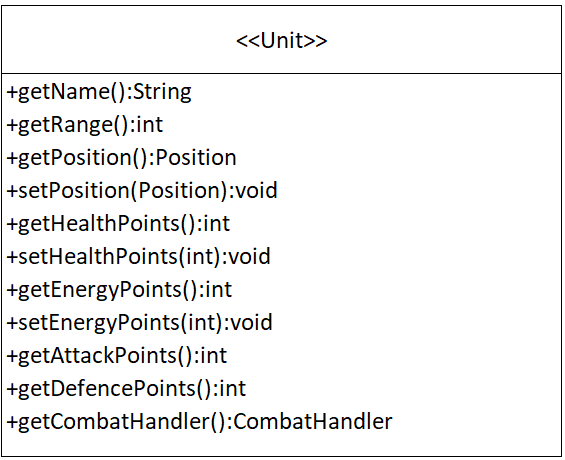
-HEALTH\_POINTS = 50

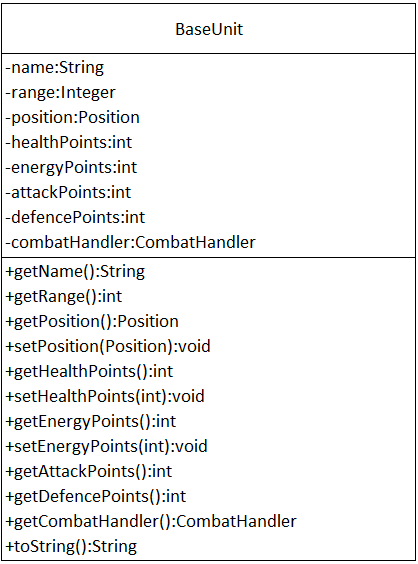
-ENERGY\_POINTS = 80

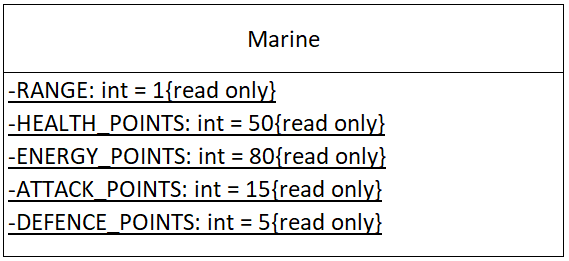
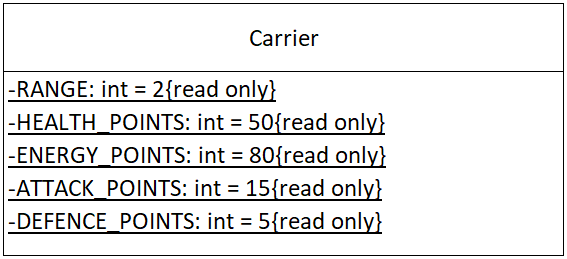
-ATTACK\_POINTS = 15

-DEFENCE\_POINTS = 5

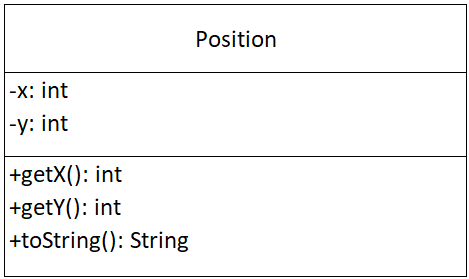
Carrier spell: BulletRain => ENERGY\_COST = 15





### Position



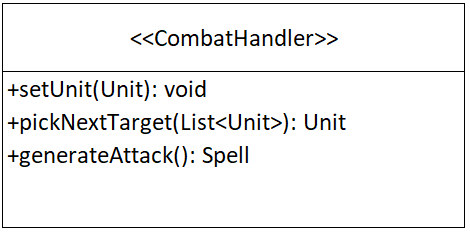
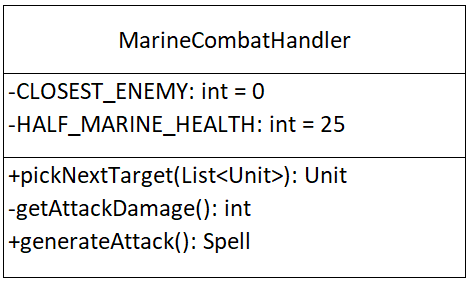
### Combat Handler

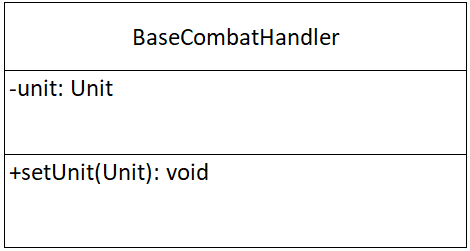
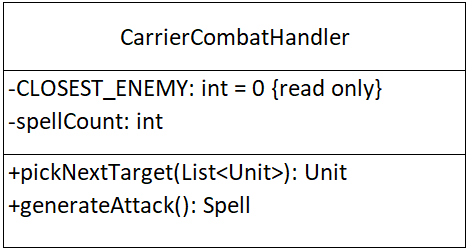
Throws NotEnoughEnergyException in interface CombatHandler.

Throws NotEnoughEnergyException in class CarrierCombatHandler.

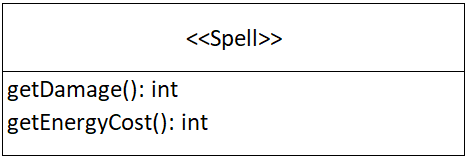
Throws NotEnoughEnergyException in class MarineCombatHandler.

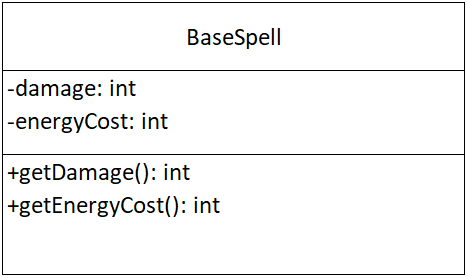
When an unit chooses the target he picks one with **lower health points**. If on battleground have more than one unit with same lower points he **compare them by name**.

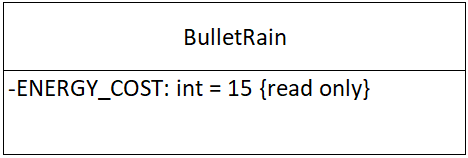
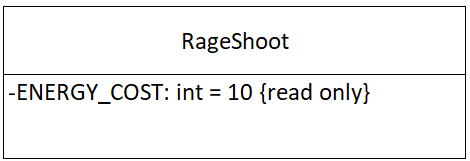
 

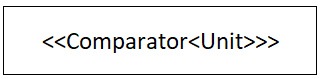
### Spells

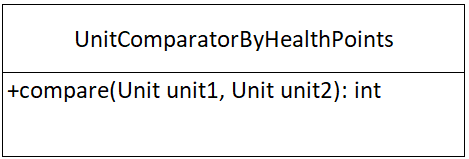
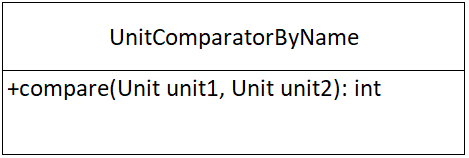




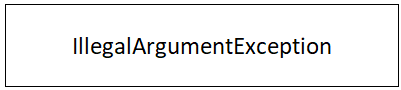
 

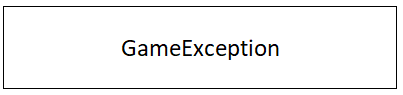
### Comparators

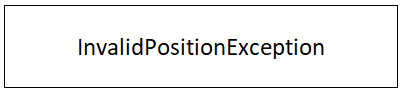
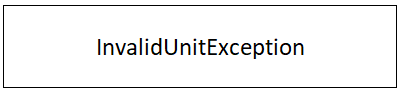
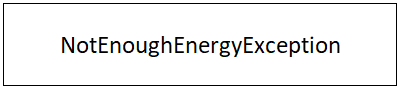


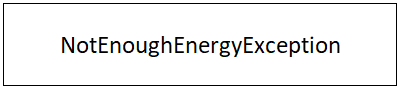
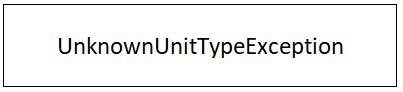
 

### Exceptions



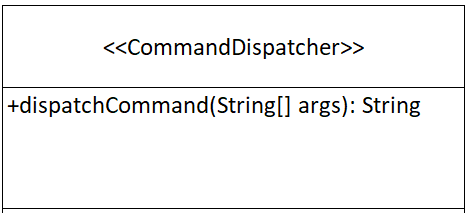


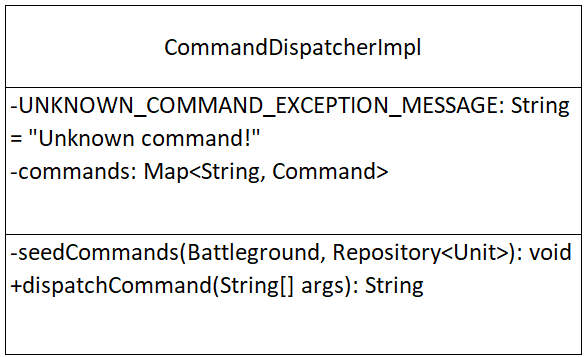
  

### Command Dispatcher

Throw GameException (UNKNOWN\_COMMAND\_EXCEPTION\_MESSAGE).





### Commands

When the command is invalid, print: “Unknown command!”.

**All Commands**:

* **Unknown command!**
* **spawn** {typeUnit} {name} {position} {position}

- Valid command: {name} of type {typeUnit} has spawn @({position}, {position})

- Invalid command:

- Invalid unit type!

- Coordinates are not correct!

- There is an unit at this position, already!

* **status** <name>

- Valid command: -

>Gosho

- Type: {typeUnit}

- Position: ({position}, {position})

- Attack Points: {attack points}

- Defence Points: {defence points}

- Energy Points: {energy points}

- Health Points: {health points}

- Invalid command:

- Not exist unit with name {name}!

* **fight**

- Valid command:

- {name} cast RageShoot spell and did 15 damage

{name} cast BulletRain spell and did 15 damage

- {name} killed {name}

- Invalid command:

- No enough spawns for fight!

- Units are outside range of their enemies, move closer and try again!

* **move**

- Valid command:

- {name} moved to ({position}, {position})

- Invalid command:

- There is an unit at this position, already!

- Not exist unit with name {name}!

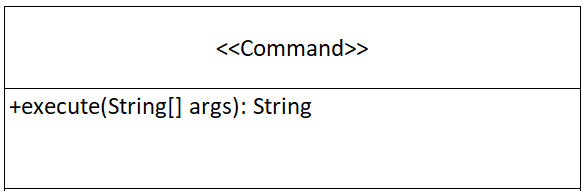
- Coordinates are not correct!

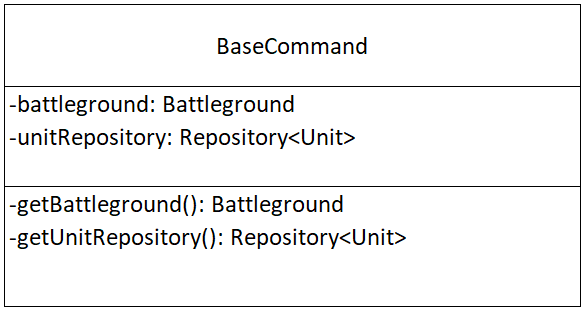
* **game-over**

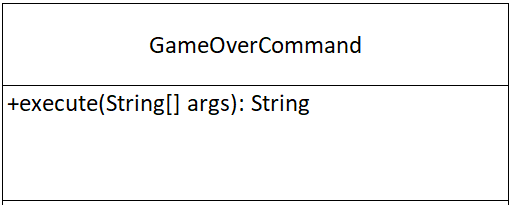
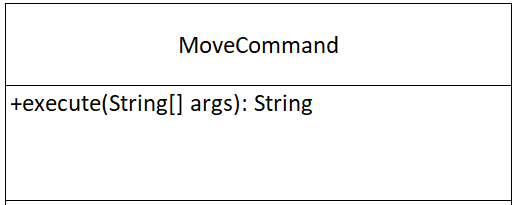
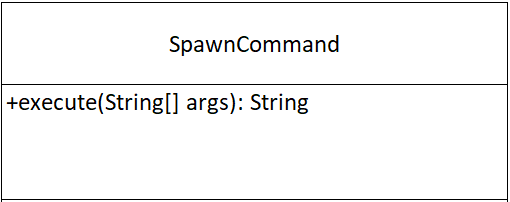
- Valid command:

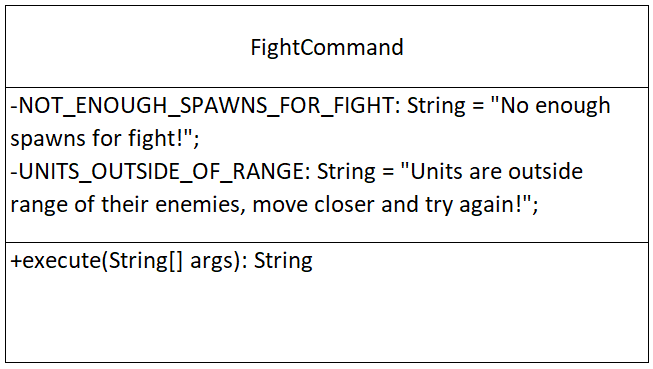
- Game Over!

Throws **UnknownUnitTypeException, InvalidPositionException, NotEnoughEnergyException** in interface.







In spawn command we should **throws UnknownUnitTypeException, InvalidPositionException** with the appropriate messages.

In move command we should **throws InvalidPositionException, NotExistingUnitException** with the appropriate messages.

In game-over command we should throws **UnknownUnitTypeException, InvalidPositionException, NotEnoughEnergyException** with the appropriate messages.

In status command we should throws **NotExistingUnitException** with the appropriate messages.

In fight command we should throws **NotEnoughEnergyException** with the appropriate messages.

**Exceptions Messages:**

NOT\_EXISTING\_UNIT\_EXCEPTION\_MESSAGE = "Not exist unit with name %s!"

INVALID\_UNIT\_TYPE\_EXCEPTION\_MESSAGE = "Invalid unit type!"

NOT\_ENOUGH\_ENERGY\_EXCEPTION\_MESSAGE = "Not enough energy to produce the spell!"

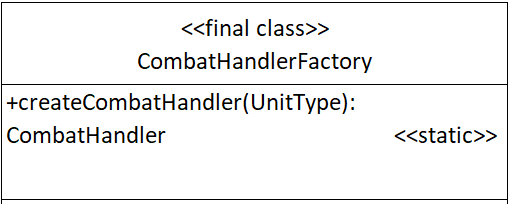
INVALID\_COORDINATES\_EXCEPTION\_MESSAGE = "Coordinates are not correct!"

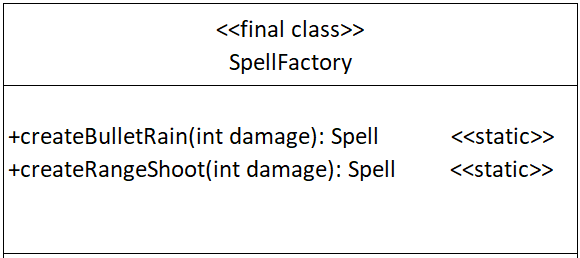
POSITION\_NOT\_EMPTY\_EXCEPTION\_MESSAGE = "There is an unit at this position, already!"

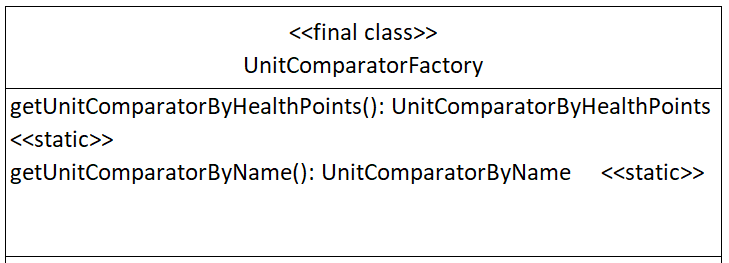
NO\_UNIT\_TO\_REMOVE\_EXCEPTION\_MESSAGE = "There is not unit to remove!"

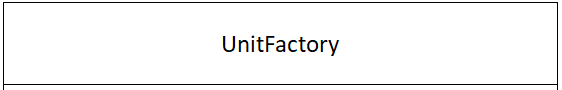
### Factories

Throw new UnknownUnitTypeException("Provided unit type is unknown!") in CombatHandlerFactory, SpellFactory and UnitFactory.

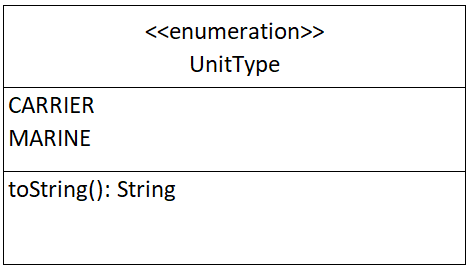
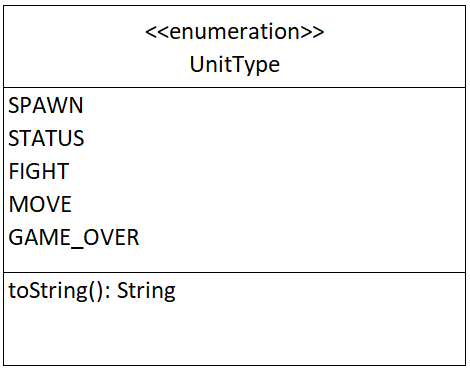




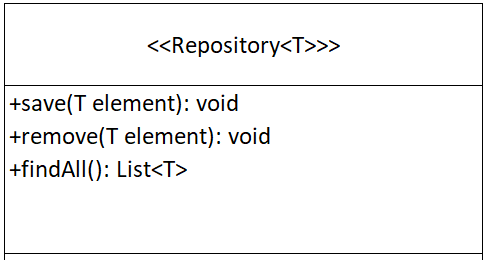


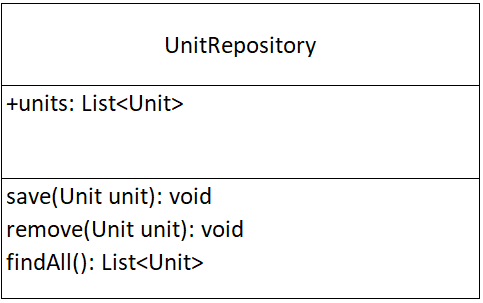


### Enum

### Repository





### Exception Messаges Constants

NOT\_EXISTING\_UNIT\_EXCEPTION\_MESSAGE: String = "Not exist unit with name %s!"

INVALID\_UNIT\_TYPE\_EXCEPTION\_MESSAGE: String = "Invalid unit type!"

NOT\_ENOUGH\_ENERGY\_EXCEPTION\_MESSAGE: String = "Not enough energy to produce the spell!"

INVALID\_COORDINATES\_EXCEPTION\_MESSAGE: String = "Coordinates are not correct!"

POSITION\_NOT\_EMPTY\_EXCEPTION\_MESSAGE: String = "There is an unit at this position, already!"

NO\_UNIT\_TO\_REMOVE\_EXCEPTION\_MESSAGE: String = "There is not unit to remove!"

### Game Constants

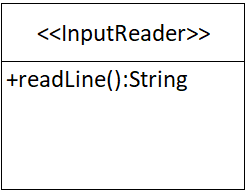
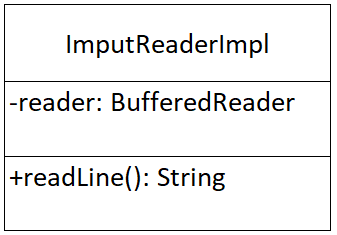
GAME\_OVER\_MESSAGE: String = "Game over!"

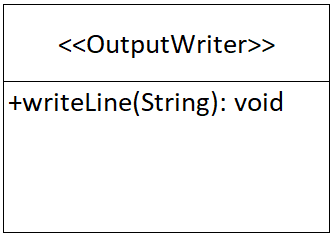
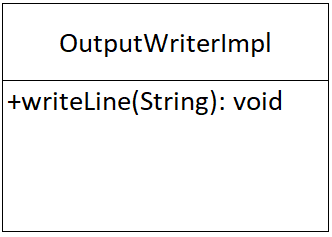
### I / O (Input / Output)

#### Input

* The input will come in the form of commands, in the format specified above.
* The input sequence ends when you receive the command “**game-over**”.

#### Output

#### Constrains

* **All integers** in the input will be in **range [0, 4]**.
* **All strings** in the input may consist of **any ASCII character**, except **SPACE**
* Note that throughout the program, you are working **ONLY** with **INTEGERS**.

#### Examples

|  |
| --- |
| **Output** |
| *spawn Marine Pesho 0 0*  Pesho of type Marine has spawn @(0, 0)  *spawn Carrier Gosho 4 4*  Gosho of type Carrier has spawn @(4, 4)  *fight*  Units are outside range of their enemies, move closer and try again!  *move Gosho 1 0*  Gosho moved to (1, 0)  *status Gosho*  ->Gosho  - Type: Carrier  - Position: (1, 0)  - Attack Points: 15  - Defence Points: 5  - Energy Points: 80  - Health Points: 50  *fightt*  Unknown command!  *fight*  Pesho cast RageShoot spell and did 15 damage  Gosho cast BulletRain spell and did 15 damage  *status Gosho*  ->Gosho  - Type: Carrier  - Position: (1, 0)  - Attack Points: 15  - Defence Points: 5  - Energy Points: 65  - Health Points: 35  *fight*  Pesho cast RageShoot spell and did 15 damage  Gosho cast BulletRain spell and did 15 damage  *fight*  Pesho cast RageShoot spell and did 30 damage  Pesho killed Gosho  *game-over*  Game over! |