

# OOP Exam – Clash of Kings

In a land full of magic, dragons and White Walkers, the thing you should fear the most is other people. The battle for the Iron Throne is raging and in this game you win or you die. Welcome to Westeros.

The game takes place on the continent of Westeros which is divided among the rival noble **Houses**. Each **House** controls a number of **Cities** and **Military Units**. It's not just about who has the largest army though, money is important as well. A House without money cannot muster an army or upgrade its cities and is doomed to perish.

Houses move units from one city to an adjacent city, attack cities controlled by their rivals, collect taxes and food and basically try to destroy each other.

## Project Structure

The project is relatively large, comprising of many interfaces and classes. You should get familiar with the basic structure, but make sure not to get lost exploring details you won't need!

Here is a short description of the project's namespaces:

- **Contracts** – contains numerous interfaces; the main ones are:
  - **IHouse** – defines the behavior of Houses; a **House** has a list of controlled cities, collects taxes, has a treasury, etc.;
  - **ICity** – defines Cities; a **City** can hold structures and units, can be upgraded to accommodate more advanced structures, produces food, pays taxes and more;
  - **IArmyStructure** – an **Army Structure** costs money to build and can hold a certain number of specific unit types (infantry, cavalry, etc.). A city should be advanced enough to accommodate certain structures.
  - **IMilitaryUnit** – a **Military Unit** has parameters like armor and damage. It costs money to train and on each turn (after each command) it costs some food to support it. It takes a certain amount of housing spaces; a city cannot hold more units than its available unit capacity (determined by the structures built in that city);
  - **IGameEngine**
  - **IContinent**
  - **ICommand** – interface for **Command** classes
- **Engine**
  - **WarEngine** – implements **IGameEngine**; the core class responsible for running the game
  - **Westeros** – implements **IContinent**; a container holding all game entities; also contains some logic for managing cities and troops
  - **Factories namespace** – contains several factory classes
- **UI** – classes responsible of user input and output
  - **ConsoleInputController** - reads and returns input from the console
  - **ConsoleRenderer** - writes output to the console
- **Exceptions** – holds custom exceptions for the application
- **Attributes** – holds attribute classes which are used to designate certain entities – commands, units, structures
- **Models** - namespace containing all army units, structures and commands with respective sub-namespaces

Implement the necessary functionality by applying the best practices of Object-Oriented Programming and Object-Oriented Design. **Encapsulate** everything correctly and keep the object state valid by **performing validations**. Avoid code repetition through **abstraction** and **inheritance**.

## Military Units

Each unit has a type (**Infantry**, **Cavalry**, or **Air Force**). The respective unit types have namespaces of their own. There is a **UnitFactory** which the engine uses to create units. Military Units have **Armor**, **Damage**, **Training Cost** (how much money it costs to produce), **Upkeep Cost** (the amount of food per turn needed to supply the unit), and **Housing Spaces Required** (how much space the unit takes).

Some units are already implemented, you need to add several more:

- **Dragon** – the Dragon is an **Air Force** unit, it has the following characteristics: **700 Armor, 1200 Damage, 1500 Training Cost, 100 Upkeep Cost, 1 Housing Space Required**
- **Dothraki** – the Dothraki is a **Cavalry** unit, it has the following characteristics: **5 Armor, 25 Damage, 25 Training Cost, 1.8 Upkeep Cost, 2 Housing Spaces Required**
- **Unsullied** – the Unsullied is an **Infantry** unit, it has the following characteristics: **5 Armor, 25 Damage, 42.5 Training Cost, 0.75 Upkeep Cost, 1 Housing Space Required**

*Score: 30 points*

## Army Structures

Army structures provide housing spaces for specific unit types. Each structure has **build cost** (decimal, cannot be negative), **capacity** (integer, also non-negative, representing the amount of housing spaces it provides) and **unit type**. E.g. a House pays the build cost of a Barracks to increase the infantry capacity of one of its cities by 5000. Additionally, each structure requires a city of certain type **or higher** – e.g. a Stable can be built in a Fortified City or Capital, but not in a Keep.

Several structures should be implemented:

- **Barracks** – requires a city of type **Keep**; costs **15000** and adds **5000** housing spaces for **Infantry** units
- **Dragon Lair** – requires a city of type **Capital**; costs **200000** and adds **3** housing spaces for **Air Force** units
- **Stable** – requires a city of type **Fortified City**; costs **75000** and adds **2500** housing spaces for **Cavalry** units

*Score: 40 points*

## Commands

Currently the game engine supports the following commands:

- **Attack {attackingCity} {defendingCity}** – an attack is performed where all military units from the attacking city are sent to attack the defending city. All validations are already performed and result messages are printed on the console.
- **City-Status {cityName}** – prints information about a city
- **Continent-Status** – prints information about the continent
- **Create-City {name} {house} {defense} {upgradeCost} {initialFoodStorage} {foodProduction} {taxBase} {cityType}** – creates a new city with the provided parameters
- **Create-House {name} {initialTreasuryAmount}** – creates a new house with the provided parameters

- **Move {startCity} {destinationCity}** – moves all units from startCity to the destination city. All validations are already performed.
- **Exit** – stops the game
- **House-Status** – prints information about a house in the following format:

```
House {name}:
-Treasury Amount: {amount}
-Controlled Cities ({numberOfCities}): {city1}, {city2}, ...
```

## Commands to Add

You need to **add the following commands** to complete the game's functionality:

- **Add-Neighbors-To-City {city} {neighbor\_1} {distanceToNeighbor\_1} {neighbor\_2} {distanceToNeighbor\_2} ...**

For the provided {city}, add the respective neighbors and distances. The distance between cities affects the amount of food needed in order to move troops from one to the other. Example:

Add-Neighbors-To-City Winterfell KingsLanding 600

The relationship is **bidirectional** – Winterfell and KingsLanding are neighbors and the direction of movement is irrelevant; the cost for moving units from one to the other is 600 and is subtracted from the storage of the starting city when performing the movement (this is already implemented in the Move command).

In each record, both cities should already **exist** and the distance between them should be **non-negative**. If the {city} doesn't exist, the command returns an error. If one of the provided neighbor records contains **invalid** info (non-existent neighbor or negative distance), print the appropriate error message and move on to the next pair. Use the table below for reference:

Outcome	Message
Success	"All valid neighbor records added for city {cityName}"
Non-existent city	None (just throw a proper exception)
A neighbor does not exist	"Specified neighbor does not exist"
A distance is negative	"The distance between cities cannot be negative"

- **Build-Structure {structureName} {city}**

Builds the specified structure in the specified city. If no errors occur the controlling house pays for the structure and it's added to the city.

Outcome	Message
Success	"Successfully built {structureName} in {cityName}"
Non-existent city	None (just throw a proper exception)
City is not advanced enough for the specified structure	"Structure requires a more advanced city"
Controlling house cannot afford to build the specified structure	"House {controllingHouseName} doesn't have sufficient funds to build {structureName}"

- **Create-Unit {numberOfUnits} {unitType} {city}**

Creates the specified number of units of the type and adds them to the specified city. **Either all units are created or none at all.**

Outcome	Message
Success	"Successfully added {numberOfUnits} units of {unitType} to city {cityName}"
Negative number of units	"Number of units should be non-negative"
Non-existent city	None (just throw a proper exception)
Not enough housing spaces in city	"City {cityName} does not have enough housing spaces to accommodate {numberOfUnits} units of {unitName}"
House cannot afford to train the specified number of units of the type	"House {controllingHouseName} does not have enough funds to train {numberOfUnits} units of {unitName}"

- **Upgrade-City {city}**

Upgrades the specified city.

Outcome	Message
Success	"City {cityName} successfully upgraded to {cityType}"
Non-existent city	None (just throw a proper exception)
House doesn't have funds to upgrade city	"House {houseName} has insufficient funds to upgrade {cityName}"
City cannot be upgraded further	"City {cityName} is at the maximum level and cannot be upgraded further"

**Score:** 80 points

## Add Great Houses

A house can become a **Great House** if it controls **at least 10 cities**. A great house does not depend on its treasury when **upgrading cities**. This means that it can keep running with a **negative balance**, unlike regular Houses (all actions that require money still affect the treasury though). Additionally, a great house **does not declare bankruptcy** which occurs when the treasury is empty. It is, in essence, Too Big to Fail. Note that soldiers and builders are not stupid, they will not fight or build structures for a House that doesn't have money, so a Great House with negative treasury amount **cannot train new troops or build Army Structures (just like a regular House)**.

If a Great House is left with **fewer than 5 cities**, it is **downgraded** back to a regular house.

Houses turn into Great Houses and back into regular Houses **automatically** on every iteration of the game loop (i.e. after every command), make sure the application does that.

A Great House's **Print()** method is the same with one difference – the first line is "**Great House {name}:**"

**Score:** 50 points

## Constraints

You are **NOT** allowed to edit **any** of the **provided** classes in any way except the **ClashOfKingsMain** class.

## Additional Notes

- Use defensive programming – public methods should check their arguments for validity and throw errors when needed
- Reuse code – study the provided code to avoid code repetition and use already implemented features; aim to achieve good abstraction

- Perform actions and validations in the proper places. E.g., is a military unit competent to check whether a house can afford to train it?
- **Perform the validations in the order in which they are described in the tables above**
- Name all classes **exactly** like the names of the units/structures/commands without the spaces and hyphens, e.g. CityStatusCommand
- Throw the appropriate types of exceptions. You are free to add new custom exceptions if you feel none of the provided ones are appropriate. Performing the same validation in more than one place is acceptable in some cases

## Examples

### Zero Test #1 Input

Input
<pre>Create-House Lannister 3000000 Create-House Stark 1200000 Create-City Winterfell Stark 150 450 5000 10000 97.75 Fortress Create-City CasterlyRock Lannister 1 1 1 1 1 default Create-Unit 5000 FootSoldier Winterfell Build-Structure Barracks Winterfell Create-Unit 5000 FootSoldier Winterfell Add-Neighbors-To-City Winterfell CasterlyRock 250 Attack Winterfell CasterlyRock House-Status Stark House-Status Lannister City-Status Winterfell Continent-Status Exit</pre>

### Zero Test #1 Output

Output
<pre>Successfully created House Lannister Successfully created House Stark Successfully created city Winterfell Successfully created city CasterlyRock City Winterfell does not have enough housing spaces to accommodate 5000 units of FootSoldier Successfully built Barracks in Winterfell Successfully added 5000 units of FootSoldier to city Winterfell All valid neighbor records added for city Winterfell House Stark conquered CasterlyRock from House Lannister House Stark: -Treasury Amount: 1124337.5 -Controlled Cities (2): Winterfell, CasterlyRock House Lannister: -Treasury Amount: 3000004.0 -No Cities Controlled Winterfell (Fortress): -Food storage: 59750.0 -Food production: 10000.0 -Tax base: 97.8 -Army: 5000 Infantry -Ruling House: House Stark Westeros: -Houses (2): Lannister, Stark -Cities (2): Winterfell, CasterlyRock For the Watch!</pre>

## Zero Test #2 Input

### Input

```
Create-House Stark 100000
Create-House Stark 0
Create-City Winterfell Stark 0 0 0 0 0 Fortress
Create-City Winterfell Stark 0 0 0 0 0 Fortress
Build-Structure Barracks Winterfell
House-Status Stark
City-Status Winterfell
Create-Unit 10000 FootSoldier Winterfell
Create-Unit 5000 FootSoldier Winterfell
House-Status Stark
City-Status Winterfell
Build-Structure DragonLair Winterfell
Upgrade-City Winterfell
Upgrade-City Winterfell
Upgrade-City Winterfell
Build-Structure DragonLair Winterfell
House-Status Stark
Create-City Dreadfort Stark 0 0 0 0 0 Fortress
Add-Neighbors-To-City Winterfell Dreadfort 100
Move Winterfell Dreadfort
Exit
```

## Zero Test #2 Output

### Output

```
Successfully created House Stark
House Stark already exists
Successfully created city Winterfell
City Winterfell already exists
Successfully built Barracks in Winterfell
House Stark:
-Treasury Amount: 85000.0
-Controlled Cities (1): Winterfell
Winterfell (Fortress):
-Food storage: 0.0
-Food production: 0.0
-Tax base: 0.0
-Army:
-Ruling House: House Stark
City Winterfell does not have enough housing spaces to accommodate 10000 units of
FootSoldier
Successfully added 5000 units of FootSoldier to city Winterfell
City Winterfell suffered starvation
House Stark:
-Treasury Amount: 23750.0
-Controlled Cities (1): Winterfell
Winterfell (Fortress):
-Food storage: 0.0
-Food production: 0.0
-Tax base: 0.0
-Army:
-Ruling House: House Stark
Structure requires a more advanced city
City Winterfell successfully upgraded to Megapolis
City Winterfell successfully upgraded to Capital
City Winterfell is at the maximum level and cannot be upgraded further
House Stark doesn't have sufficient funds to build DragonLair
House Stark:
-Treasury Amount: 23750.0
```

```
-Controlled Cities (1): Winterfell
Successfully created city Dreadfort
All valid neighbor records added for city Winterfell
No troops to move
For the Watch!
```

## Zero Test #3 Input

### Input

```
Create-House Greyjoy 1000000
Create-City Pyke Greyjoy 100 100 100 100 100 Capital
Build-Structure Barracks Pyke
Create-Unit 100 SellSword Pyke
Create-House Martell 1000000
Create-City Sunspear Martell 500 500 500 500 500 Capital
Build-Structure Barracks Sunspear
Create-Unit 500 SellSword Sunspear
Add-Neighbors-To-City Pyke Sunspear 10
Attack Sunspear Pyke
House-Status Martell
Create-City Tarth Martell 500 500 500 500 500 Fortress
Create-City CastleBlack Martell 500 500 500 500 500 default
Create-City KingsLanding Martell 500 500 500 500 500 default
Create-City StormsEnd Martell 500 500 500 500 500 default
Create-City Riverrun Martell 500 500 500 500 500 default
Create-City Eyrie Martell 500 500 500 500 500 default
Create-City Winterfell Martell 500 500 500 500 500 default
House-Status Martell
Create-City Twins Martell 500 500 500 500 500 default
House-Status Martell
Exit
```

## Zero Test #3 Output

### Output

```
Successfully created House Greyjoy
Successfully created city Pyke
Successfully built Barracks in Pyke
Successfully added 100 units of SellSword to city Pyke
Successfully created House Martell
Successfully created city Sunspear
Successfully built Barracks in Sunspear
Successfully added 500 units of SellSword to city Sunspear
All valid neighbor records added for city Pyke
House Martell conquered Pyke from House Greyjoy
House Martell:
-Treasury Amount: 977600.0
-Controlled Cities (2): Sunspear, Pyke
Successfully created city Tarth
Successfully created city CastleBlack
Successfully created city KingsLanding
Successfully created city StormsEnd
Successfully created city Riverrun
Successfully created city Eyrie
Successfully created city Winterfell
House Martell:
-Treasury Amount: 996400.0
-Controlled Cities (9): Sunspear, Pyke, Tarth, CastleBlack, KingsLanding, StormsEnd,
Riverrun, Eyrie, Winterfell
Successfully created city Twins
Great House Martell:
-Treasury Amount: 1005100.0
```

**-Controlled Cities (10): Sunspear, Pyke, Tarth, CastleBlack, KingsLanding, StormsEnd, Riverrun, Eyrie, Winterfell, Twins  
For the Watch!**