

OOP Exam - Winter Is Coming

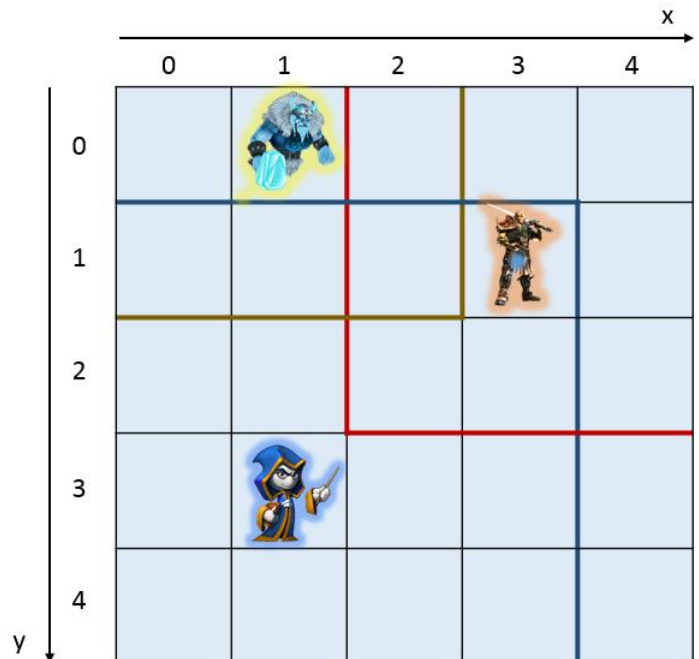
Winter is coming and champions from all over the realm gather for the final battle. Warriors, mages, ice giants... have come to contest the Iron Throne. Unbeknownst to them, the Game has no winners...

There are units in the game - each unit has the following characteristics: **x** and **y** (position on map), **name**, **attack points**, **health points**, **defense points**, **energy points** and **range** of attack.

There are 3 types of units - **warrior**, **mage** and **ice giant**. Units have an **attack range** and can move **1 cell** in **any direction** at a time.

Units can also attack other units in their range by casting spells.

Your task is to implement each of the described units and their behavior.



Project Structure

Here follows a short description of the project namespaces:

- **Contracts**
 - **ICombatHandler** - a combat handler used for determining a unit's targets and spells during a fight. Defines two methods: **PickNextTargets()** and **GenerateAttack()**.
 - **ICommand** - an executable action used by the engine for
 - **ICommandDispatcher** - a command manager (dispatcher) called by the engine whenever a command must be executed
 - **IEngine** - defines an engine that holds a **collection of units**, **input/output** means, etc.
 - **IInputReader** - defines a source of input (e.g. console)
 - **IOutputWriter** - defines an output source (e.g. console)
 - **ISpell** - an attack produced by units during battle; holds **damage** and **energy cost**
 - **IUnit** - a unit with **position** (x, y), **name**, **name**, **range**, **attack points**, **health points**, **defense points**, **energy points** and a **combat handler** (used for determining the unit's actions during combat)
 - **IUnitContainer** - a container for keeping units in some kind of a structure (i.e. 2D matrix or 3D plane)
 - **IUnitEffector** - a global entity that gives effects to all units in the game
- **Core**
 - **CommandDispatcher** - implements **ICommandDispatcher**, used by the engine for executing text commands
 - **ConsoleReader** - implements **IInputReader**, reads and returns input from the console
 - **ConsoleWriter** - implements **IOutputWriter**, writes output to the console
 - **EmptyUnitEffector** - empty class that implements **IUnitEffector**, its **ApplyEffects()** method does nothing
 - **Engine** - the core class responsible for running the game
 - **GlobalMessages** - a static class that holds constant strings

- **MatrixContainer** - the game map (a 2D matrix that holds units) - supports adding/removing and moving units around
- **Unit Factory** - a static factory class that produces units
- **Models** - namespace left for implementing units, spells and combat handlers...

Study the provided code in depth. 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 clean by **performing validations**. Avoid code repetition through **abstraction** and **inheritance**.

Units

There are 3 types of units you must implement - **Warrior**, **Mage** and **Ice Giant**, each with their own characteristics and behavior during combat.

- **Warrior** - picks the first target in range with **least health points** and casts **Cleave**. If there are several targets with **equal health points**, the one with **alphabetically first name** is picked. If the warrior's health is **greater than 50**, it costs him energy - otherwise it doesn't.
 - **Default stats:** 120 attack points, 180 health points, 70 defense, 60 energy, range 1
 - **Cleave damage:** Equals the warrior's **attack points**. If the warrior's health is equal or below 80, his **health * 2** is added to the damage.
- **Mage** - picks **3 targets** with **most health points**. If there are several targets with **equal health points**, picks those with **alphabetically first name**. Casts **Fire Breath** and **Blizzard**, alternating each time he **successfully casts** a spell (i.e. first **Fire Breath**, next time **Blizzard**, then **Fire Breath** again, etc.).
 - **Default stats:** 80 attack points, 80 health points, 40 defense, 120 energy, range 2
 - **Fire Breath damage:** Equals the mage's **attack points**.
 - **Blizzard damage:** Equals the mage's **attack points * 2**.
- **Ice Giant** - casts **Stomp** on all units in range. If his health is less or equal to **150**, Stomp is cast only on the **first unit** in range. Each time he casts Stomp, the Ice Giant's **attack points** are increased by 5.
 - **Default stats:** 150 attack points, 300 health points, 60 defense, 50 energy, range 1
 - **Stomp damage:** Equal to the Ice Giant's **attack points**.

If a unit does **not have enough energy** to cast a spell, an error should be returned with message "{unit} does not have enough energy to cast {spell}". In that case, his **turn ends**.

Spells

Spells deal **damage** according to their caster and cost him some amount of **energy**:

- **Cleave** - costs 15 energy
- **Fire Breath** - costs 30 energy
- **Blizzard** - costs 40 energy
- **Stomp** - costs 10 energy

Commands

Currently the game engine supports the following commands.

- **spawn {type} {name} {x} {y}** - adds a new unit to the game at the specified position
- **fight** - each unit (**in order of insertion** into the game) is allowed to perform an **attack** on other units

- **move {name} {x} {y}** - moves the specified unit to a new position
- **status {name}** - prints information about a unit in the following format:

If Health > 0	If dead
>{name} - {type} at ({x},{y}) -Health points = {...} -Attack points = {...} -Defense points = {...} -Energy points = {...} -Range = {...}	>{name} - {type} at ({x},{y}) (Dead)

- **toggle-effector** - executes the current engine effector by calling its **ApplyEffects()** method (**Note:** not supported by the engine at the moment)
- **winter-came** - stops the engine

You are not allowed to edit these commands.

Add Toggle-Effector Command

Currently there is a **ToggleEffectorCommand** that is not supported by the engine. Insert it into the game without directly editing any of **Core namespace** classes.

The command uses the current engine **unit effector** by calling its **ApplyEffects()** method. Make so that whenever the **toggle-effector** command is executed, all living units have their **health points** raised by **50**.

Constraints

You are **NOT** allowed to edit any of the **Core namespace** classes in any way (only the **UnitFactory**). You are also not allowed to change the provided interfaces. You may edit everything else, including the **WinterIsComingMain** class.

Additional Notes

- The world size will always be **5x5** (5 rows and 5 columns).
- If a unit does not have enough energy to cast a spell, his turn ends.
- The mage's attacks should alternate only if he successfully casts a spell.
- If a unit's health falls below 0 (dies), the engine automatically raises it back to 0.

Examples

Zero Test #1

Input	Output
spawn Mage Nakov 1 3 spawn IceGiant Emi 1 0 spawn Warrior Alex 3 1 fight status Alex status Emi status Nakov winter-came	Nakov has spawned Emi has spawned Alex has spawned Nakov cast FireBreath on Alex for 10 damage >Alex - Warrior at (3,1) -Health points = 170 -Attack points = 120 -Defense points = 70 -Energy points = 60 -Range = 1

	>Emi - IceGiant at (1,0) -Health points = 300 -Attack points = 150 -Defense points = 60 -Energy points = 50 -Range = 1 >Nakov - Mage at (1,3) -Health points = 80 -Attack points = 80 -Defense points = 40 -Energy points = 90 -Range = 2
--	--

Zero Test #2

Input	Output
spawn Mage Nakov 0 1 spawn Mage Emi 0 3 spawn Mage Alex 2 3 fight status Alex status Emi status Nakov winter-came	Nakov has spawned Emi has spawned Alex has spawned Nakov cast FireBreath on Alex for 40 damage Nakov cast Blizzard on Emi for 120 damage Nakov has killed Emi Alex cast FireBreath on Nakov for 40 damage >Alex - Mage at (2,3) -Health points = 40 -Attack points = 80 -Defense points = 40 -Energy points = 90 -Range = 2 >Emi - Mage at (0,3) (Dead) >Nakov - Mage at (0,1) -Health points = 40 -Attack points = 80 -Defense points = 40 -Energy points = 50 -Range = 2

Zero Test #3

Input	Output
spawn Mage Fil 3 3 spawn Warrior Petya 1 2 spawn IceGiant Nasko 0 3 fight move Nasko 1 3 move Nasko 2 3 fight status Petya status Nasko status Fil winter-came	Fil has spawned Petya has spawned Nasko has spawned Fil cast FireBreath on Petya for 10 damage Petya cast Cleave on Nasko for 60 damage Nasko cast Stomp on Petya for 80 damage Nasko has moved to (1,3) Nasko has moved to (2,3) Fil cast Blizzard on Nasko for 100 damage Fil cast FireBreath on Petya for 10 damage Petya cast Cleave on Nasko for 220 damage Petya has killed Nasko >Petya - Warrior at (1,2) -Health points = 80 -Attack points = 120 -Defense points = 70

	-Energy points = 30 -Range = 1 >Nasko - IceGiant at (2,3) (Dead) >Fil - Mage at (3,3) -Health points = 80 -Attack points = 80 -Defense points = 40 -Energy points = 20 -Range = 2
--	--

Zero Test #4

Input	Output
spawn Mage Fil 3 3 spawn Warrior Petya 1 2 spawn IceGiant Nasko 0 3 toggle-effector status Fil status Petya status Nasko winter-came	Fil has spawned Petya has spawned Nasko has spawned >Fil - Mage at (3,3) -Health points = 130 -Attack points = 80 -Defense points = 40 -Energy points = 120 -Range = 2 >Petya - Warrior at (1,2) -Health points = 230 -Attack points = 120 -Defense points = 70 -Energy points = 60 -Range = 1 >Nasko - IceGiant at (0,3) -Health points = 350 -Attack points = 150 -Defense points = 60 -Energy points = 50 -Range = 1

Zero Test #5

Input	Output
spawn Mage Nasko 2 2 spawn Warrior Dancho 0 0 spawn IceGiant Fil 4 4 fight fight fight winter-came	Nasko has spawned Dancho has spawned Fil has spawned Nasko cast FireBreath on Fil for 20 damage Nasko cast Blizzard on Dancho for 90 damage Nasko cast FireBreath on Fil for 20 damage Nasko does not have enough energy to cast Blizzard Nasko does not have enough energy to cast Blizzard