# Object-Oriented Programming – Practical Exam

## Problem 2 – Trade and TravelAPI

You are given an API, which supports interactions between different actors (people) and items, occurring in different locations. You are also given a C# file, which has a Main method and uses the API for processing commands from the input.

There are some simple rules the API supports:

* Everything is an object
  + Every object has a name
* Every object is at some location (items are sometimes “inside” a person’s inventory and are then considered as not having a location)
* Locations are specified by names and can be several types (e.g. town)
* A Person can have items and money (every person has “100 money” initially)
  + The items a person has are referred to as his “inventory”
  + A person can drop all of his items at a location (at that moment, any other person can take them)
  + A person can pick up all items at a location
* A Person can be a Shopkeeper, enabling him to sell things for money
  + A Person can also sell things to a Shopkeeper
  + Any Person can fall in debt – that is, have less than 0 money
* A Person can be a Traveller, enabling him to move from one location to the other
* There can be several types of items, the API currently has “armor” implemented
* Items have “value”. Value is what determines the amount of money is spent when buying/selling an item
  + Shopkeepers have the right to determine at what price they sell or buy items
* There can be several types of locations, the API currently has “town” implemented

### Commands

There are two types of commands the Engine supports:

* Creation commands – create items, people or locations
  + Creating locations requires a location type and location name
  + Syntax: create location town *sofia*
  + Creating items requires an item type, item name and location name
  + Syntax: create item armor *coolarmorsofia*– creates an armor type item, named “coolarmor” at location “sofia”
  + Creating people requires a person type, person name and location name
  + Syntax: create traveller *Nelson sofia*– creates a traveller type of Person, with the name of Nelson
* Person commands – order a person to move, buy, sell, drop, pick up items, etc.
  + Person commands start with the person’s name and continue with the type of command
  + A Person can list his inventory
  + Syntax: Joro inventory – outputs all the names of the items in Joro’s inventory
  + A Person can show his money
  + Syntax: Joro money
  + A Person can drop all his items, leaving his inventory empty
  + Syntax: Joro drop
  + A Person can pick up all items at his location, placing them in his inventory
  + Syntax: Joro pickup
  + A Person can travel from one location to another, if he is created as a traveller
  + Syntax: Joro travel Gabrovo
  + A Shopkeeper can be bought from or sold to
  + Syntax: Joro buy coolarmor NikiTheShopman – Joro buys the “coolarmor” item from NikiTheShopman, who is a shopkeeper
    - Joro and NikiTheShopman must be in the same location for this to happen
    - NikiTheShopman must have an item named “coolarmor” for this to happen
  + Syntax: Joro sell jorosarmor NikiTheShopman – Joro sells his “jorosarmor” item to NikiTheShopman
    - Analogous rules to the “buy” command

### Tasks

You are tasked with extending the API by implementing several commands and object types. You are **not allowed to edit any existing class from the original code of the API**. You **are allowed to edit the Main method**.

* Implement a command to create a Weapon item
  + The Weapon item has a moneyvalue of 10
  + Syntax: **create item weapon *weaponname location*** - creates a weapon with the given name, at the given location
* Implement a command to create a Wood item
  + The Wood item has a money value of 2
  + The Wood item decreases its value each time it is dropped by 1, until it reaches 0
  + Syntax: **create item wood *woodname location***
* Implement a command to create an Iron item
  + The Iron item has a money value of 3
  + Syntax: **create item iron *ironname location***
* Implement a command to create a Mine location
  + Syntax: **create location mine *BobovDol***– creates a location, which is a mine with the name of *BobovDol*
* Implement a command to create a Forest location
  + Syntax: **create location forest *Cherokee*** – creates a location, which is a forest, with the name *Cherokee*
* Implement a “gather” command
  + Gathering means a Person takes an item from a special location
  + A Person should be able to gather from mines and from forests
  + A Person can gather from a forest only if he has a Weapon in his inventory
    - Gathering from a forests results in adding a Wood item in the Person’s inventory
  + A Person can gather from a mine only if he has an Armor in his inventory
    - Gathering from a mine results in adding anIron item in the Person’s inventory
  + Syntax: **Joro gather*newItemName***– gathers an item, naming it *newItemName* if the Person *Joro* is at a mine or forest, and respectively has an Armor or Weapon
* Implement a “craft” command
  + A Person can craft items, provided he has some items in his inventory
  + A Person should be able to craft Weapons and Armor
  + Crafting an Armor requires that the Person has Iron in his inventory
    - Results in adding an Armor item in the Person’s inventory
  + Crafting a Weapon requires that the Person has Iron and Wood in his inventory
  + Syntax: **Joro craft *newItemName***- gathers an item, naming it *newItemName* if the Person *Joro* has the necessary
* Implement a command to create a Merchant
  + A merchant is a Shopkeeper, supporting all of the shopkeeper’s abilities, but can also travel from one location to another
  + Syntax: **create merchant *Joro sofia***–creates a merchant with the name *Joro* at the location *sofia*

### Input and Output Data

You should not concern yourself with handling input and output data – the engine does it for you. You should only consider how to implement the required commands. See the existing API code for hints. Also:

* The names in the commands will always consist of upper and lowercase English letters only.
* In the input, all locations will be created before all other objects
* If for some reason a command is illegal (i.e. trying to sell to someone in a different location), just skip it

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| **create location town whiterun**  **create location town riften**  **create location mine cidna**  **create location forest blackmarsh**  **create item armor theArmor whiterun**  **create item weapon Axe blackmarsh**  **create item armor MineClothes blackmarsh**  **create traveller pesho whiterun**  **create merchant kiro whiterun**  **pesho inventory**  **pesho money**  **pesho pickup**  **pesho inventory**  **pesho travel riften**  **pesho drop**  **create shopkeeper joro riften**  **joro pickup**  **joro inventory**  **pesho buy theArmor joro**  **pesho money**  **pesho sell theArmor joro**  **pesho inventory**  **kiro travel riften**  **kiro buy theArmor joro**  **pesho buy theArmor kiro**  **kiro money**  **kiro travel blackmarsh**  **kiro gather x**  **kiro inventory**  **kiro pickup**  **kiro gathergatheredAtBlackmarsh**  **kiro travel cidna**  **kiro gathergatheredAtCidna**  **kiro inventory**  **kiro craft weapon craftedWeapon**  **kiro craft armor craftedArmor**  **kiro inventory**  **end** | **empty**  **100**  **theArmor**  **theArmor**  **95**  **empty**  **100**  **empty**  **Axe**  **MineClothes**  **gatheredAtBlackmarsh**  **gatheredAtCidna**  **Axe**  **MineClothes**  **gatheredAtBlackmarsh**  **gatheredAtCidna**  **craftedWeapon**  **craftedArmor** |