Class 3 – Packages, Inheritance

~~Exercise 1:~~

Lets go back to our restaurant problem. Create a new package that is called class3.collectionExamples. Inside it copy the Menu class code from your previous package that has the class Menu.

Part One
What compilation problems can you find?
Rename Menu to ArrayList menu using Eclipse's Refractor functionality.

Part Two
Replace your original structure for storing menus (stored as an Array) with an ArrayList and adapt your code (you may need to create a new class for the restaurant called ArrayListRestaurant). Compile and run your program.
Did you see how the principle of encapsulation can help update the original code of the restaurant menu program?

Part Three
Consider the previous exercise. How would you make an order that is not just 3 dishes, but as many as the customer wants?

Exercise 2:

Let's create a simple RPG game. We have people in our RPG world, that all have a name, a sex, a health, and a mana indicator (that is 0). People can sometimes be wizards, thieves or fighters. Wizards have mana of 100, in addition to health. All people can say their name, but because both wizards and fighters are proud of their profession, they always say it together with their name (thieves do not, they are clearly reluctant to advertise their profession!).

Part one
Take pen and paper and create a class diagram (i.e., a schematic) that demonstrates the different classes that can help you solve this problem. Each class is a box, that includes the class name, its fields and its methods. Make sure to add the *visibility* of the class members, and for methods their return type and argument types. Draw any inheritance relations with an arrow. Note that you'll need a class that represents the RPG world to start your game.

Addition: We did not insist in class yet on this, but if you want an inherited class (e.g., Wizard) to see members of Person (e.g., change the field _health that comes from person), then the members need to be public (BAD) or protected (instead of private). We'll come back to this again.

Part two
Create a package class3.inheritanceRPG and a main class that represents your RPG world. Next add the classes you chose in part one, adding all fields and the methods for saying their name. In your RPG class create three characters, one of each profession and tell them to introduce themselves!

Part three

First improve the construction of your people (use constructors ☺) assigning values to all the fields of your characters. Make sure Wizards have mana that is more than 0.

~~Exercise 3:~~

A queue is an abstract data type for adding and removing elements. The first element added to a queue is the first element that is removed (first-in-first-out, FIFO). Queues can be used, for instance, to manage processes of an operating system: the first process added to the waiting queue is reactivated prior to all other processes (with the same priority).

Design an interface Queue, with methods to add and remove elements (integers). Furthermore, a method to check whether the queue is empty or not should exist.

1. Implement the queue with an Array (call it ArrayQueue). If the array becomes to small to hold all added elements, create a new larger (double the size of instance) array and copy all elements of the small array to the new one.

2. Implement the queue with an ArrayList (call it ArrayListQueue).

Thoroughly test your implementation with small and large queues.

Exercise 4:

Let's continue working on our RPG game. We are now ready to do fun stuff! Do this question in steps, and test each step.

1. People in our RPG can attack other people (otherwise where is the fun). Each person has a default attack of 5, that means that every time a person attacks another, it reduces their opponent's current health by 5.
If they are attacked on the other hand, their health is reduced by the damage/attack of their opponent.
If their health becomes 0 they replenish it to their original health (let's say they have healing potions!). So we need to keep track of the original health.

Think that a person both attacks, and is attacked.

Let's code! Add the new methods to your classes. Try to use encapsulation.
In your RPG world have some of your characters attack each other. Make sure to print some information about the attack.

2. We'll now adapt attacks to our characters. When wizards attack, they do magic so they use their mana. At each wizard attack his or her mana is reduced by10. But they deal more damage to their opponents, their attack reduces the opponent's health by 20 points. When they run out of mana, the take a potion to replenish it to their original mana (so we need to keep track of this).
Fighters also do higher damage than a regular person. Their attack is such that it reduces their opponent's health by 10.

What are the methods you need to override? What fields to add?

Let's code! Add the new methods to your classes. Try to use encapsulation.
In your RPG world have some of your characters attack each other. Make sure to print some information about the attack.

Part five
Just for fun, place all your characters in an ~~ArrayList~~ Array and have them each attack the next person on the list (the last character should attack the first). Consider using the modulo % operator.
Make this attacking spree continuous, until all but one of the characters looses their health. To ensure they do not go on attacking forever, assume they have no healing potions, i.e., change your code to make sure they do not replenish their health to their original health every time it turns to 0.

~~Exercise 5:~~

Imagine that someone asks you to extend your RPG world. Assume your people have all the attributes we discussed already. They can also be of different race: elves, orcs and humans. After they get created, elves have a bonus of 50 extra mana, but 30 less health than expected. Orcs have 20 more health points after their creation, but also 20 less points of mana. They have other distinctions (skin color, height, weight, etc.) and can do different things (e.g., elves can see in the dark and run fast), that would all go into their respective class, but we'll ignore them for now.

Each character in your RPG may be of one race, but have more than one professions (wizard, thief, fighter). Each profession has a special type of attack:
Wizards can do magic attack that depletes their mana (10 per attack) but deals 20 damage to the opponent.
Thieves can do sneak attack that deals the normal 5 damage but gives these 5 health points to the attacker thief helping him heal (cool!).
Fighters can do power attacks that deal 15 damage.

You want to design types of characters that are of one race, but can have more than one profession. To save time, pick 3: e.g., elves that are wizards and thieves, orcs that are fighters, and humans that are wizards (or pick any 3 combinations that you like). Think of having a method at each class that chains the possible attacks of your character.

1. We need to rethink our architecture, so use pen and paper again. What makes more sense in terms of inheritance?

2. Create a new package class3.interfaceRPG and create your new world. Create five characters that fall under the 3 types you have defined. And of course have them attack each other!

Where would you use abstract classes in your code?