SCC110: Software Development Term 2. Java Programming.

Problem Set 3: Principles of OO Programming

Aims

This week's lab aims to:

- Introduce you to Object Oriented thinking
- Show you how to write your own class
- Teach you how to create and use instances of classes written by you and others

Exercise 1: The Person Object

This exercise is all about creating your own classes containing instance variables and methods. In this exercise you'll be creating a class to represent a person. The lecture notes from this week contains all the information you need to complete the tasks... but you may need to read on beyond the slides we covered in the lecture, and remember to follow the Java camel case naming conventions!

The Task:

- Create a new class called **Person**.
- Add **instance variables** to your class to hold information on a person's **name** and **age**… make the default name for a Person be John Doe and the age zero. Think carefully about the type these variables should be.
- Write a method called **display** in your Person class. This should print out the person's name and age to the screen using System.out.println(). This method should take no parameters, and return nothing (void).
- Create another class called **Driver**. In this class write a main method that **makes an instance of your Person class**. Add another line of code that calls the display method on your instance. Test the behaviour of your program.
- Write a method in your Person class called **birthday** that increases the age of a Person by one year when called. This method should take no parameters, and return nothing (void). Update your main method to test your new method.
- Write a method in your Person class called **changeName**. This should take a single parameter, and change the name of the Person to the one given by the parameter. **HINT: Remember you discovered how to assign values to Java Strings last week…**
- In your main method, create two more instances of your Person object, one to represent me (Joe Finney) and one to represent Adrian Friday. (You can guess our ages and call birthday repeatedly to set this!). Once created, use your display method to print the details of your two instances to the screen.

Exercise 2: Top Trumps Legendary Computers

This exercise provides more practice with creating your own classes and methods. Your task is to model the classic children's card game "Top Trumps". If you're not familiar with the game, you can try it out here: http://supercarworld.com/cgi-bin/setuptrumps.cgi.

We're not expecting you to create a complete implementation of the game in this lab (that would take too long). But we do want you to create the necessary classes, instance variables, methods and instances to model the cards in a simple **Legendary Computers** Top Trumps game. More specifically, using OO principles, create appropriate Java classes and instances to represent and **display** top trumps cards with the following information:

Name	Age (Years)	Speed (MIPS)	RAM (Kbytes)
Manchester Baby	72	0.001	0.125
TRADIC	66	1	0.062
Sinclair Spectrum	38	3.5	16
Commodore Amiga	33	7.09	512
iMac G3	22	233	524288
iPhone 1	13	620	131072
ThinkPad X1 Carbon	1	27405	16777216

HINT: We're leaving it up to you to decide how to structure your program. This may seem a little difficult at first, as Object Oriented design is a different skill to procedural programming. Just remember what we discussed in this week's lecture and apply it to problem here.

Remember:

- Classes let you define new types of object – like a blueprint. What class might you create here to help?
- Instance variables hold the information that defines an object (what data it contains). What instance variables will your class have, and what type will those variables be?
- Methods let you add functionality to an object (what it can do). What methods might you write to make your code is simple as possible?
- Instances of classes are a realization that blueprint. What instances might you create here?

Advanced Task: Build a working Top Trumps game.

This is an optional addition task, if you've completed the first two tasks and want to practice more OO concepts.

Develop the top trumps classes you have already stated into a completed game! For a full set of rules, see here: https://toptrumps.com/classics/

I'd suggest taking the following approach:

- Write a method to compare two cards in your program, and given the appropriate category, determine which card wins.

- Try using arrays to hold your cards, so you can model each of the player's piles of cards.

- Just use a text-based command line user interface. We'll be looking at simple graphical user interfaces later in the course… You might find the Scanner class useful. Here's a simple example: https://www.w3schools.com/java/java_user_input.asp

References

If you'd like to find out more about some of these historic computers, see these references:

- https://en.wikipedia.org/wiki/Manchester_Baby
- https://en.wikipedia.org/wiki/TRADIC
- https://www.computer.org/csdl/proceedings/afips/1954/5045/00/50450046.pdf
- https://en.wikipedia.org/wiki/ZX_Spectrum
- https://en.wikipedia.org/wiki/Amiga_500
- https://en.wikipedia.org/wiki/IMac_G3
- https://en.wikipedia.org/wiki/IPhone_(1st_generation)