University of Newcastle
SENG1110/6110 Programming Assignment 1 – Semester 1, 2022
Due: By electronic submission (Canvas) by 11:59pm on **Friday May 27**.

Income Tax and Investment Calculator – Part II

Introduction

The objective of this assignment is to extend the implementation of Assignment 1 using <u>**arrays**</u> (as seen in weeks 9 and 10) and <u>**external files**</u>.

SENG1110 students - This assignment can be completed **in pairs**.
SENG6110 students - This assignment must be completed **individually**.

Before you start

Carefully read the specification below. Make sure you have all the information necessary to start the program. If you are uncertain of something, do not make assumptions. Post your questions to the discussion board forum named "assignment 2" and check it regularly.

Problem specification

The program will allow <u>up to 5 clients</u> and <u>each client will be able to have 2 accounts</u>. The program will have the following options:

- <u>Add a client</u>
  If the number of clients is already 5, the program will say that it is not possible to add an extra client.
  If the number of clients is less than 5, then the program will ask the following inputs:
  - **Client name** - the name needs to have at least two names – first name and last name; if not, the program will show an error message and ask the user the name again. In addition, if a client with that name already exists, the program will show an error message, and asks again.
  - **Client annual income/salary** for an entire year (the salary needs to be a positive number and different from zero; if not, the program will show an error message and ask the salary again).
  - **Residence** - if the user is a resident or not.
  - **Expenses -** the amount of money that the user requires (on an average) per week as living expenditure. You are to assume this expenditure value does not change for the individual over time. The amount needs to be a positive number less than the net salary; if not, the program will show an error message and ask the expenditure again.

- <u>Delete Client</u>
  The user needs to enter the client name. Then, the program will delete the client and will display the message “the client was deleted”. If the name is not found, the program will display the message “the client does not exist”. It is not necessary to ask again.

- <u>Display a client</u>
  The program will ask the name of the client. If the name does not exist, show a message. If the name exists, then the program will display the following information of the Client:
  - Name
  - Gross salary per year and per week.
  - Residency status
  - The amount of tax the individual pay per year and per week (as defined by the Tax rates described in assign1).
  - The net salary per year and per week (calculated the same way as in the assign1).
  - If the individual pays the Medicare Levy and if so, how much per year.
  - Expenses per week.

- The remaining amount possible to be invested per week. The remaining amount possible to invest is net salary less expenses less the amount already invested (in case the client already has an account). For example, If the net salary is $500 per week, expenses are $100, and the user has one account already investing $200 per week, then the remaining amount available to invest is $200 (500-100-200) per week.
- Each account information, which will be:
  - Amount invested per week
  - Interested rate
  - Number of weeks
  - Total amount in the end of the period

- <u>Display all clients</u>: If it does not have any client, the program will display the message "no clients". If clients exist, the program will display the following information of all clients.
  - Client 1:
    - Name:
    - Resident:
    - Gross salary (per week):
    - Net salary (per week):
    - Tax (per week):
    - Medicare (per week):
    - Expenses (per week):
    - No accounts (if the client does not have any account)
      OR
      Number of Accounts:

  - Client 2:
    …
    etc

- <u>Add an account</u>
  The program will ask the name of the client. If the name does not exist, show a message. If the name exists, then the program will check if the client already has 2 accounts. If it is 2, the program will show a message. If it is less than 2, then the program will ask
  - **Investment value** - the amount of money that the user would like to invest per week (it is necessary to verify if this amount is feasible. If not, show an error message and ask again). To be considered a feasible amount, it should be less than the remaining amount available to invest. If not, ask again. For example, if the net salary is $500 per week, and the user has no account, and expenses are $100, then the remaining amount available to invest is $400. If the net salary is $500 per week, and the user has one account already investing $200 (and expenses are $100), then the remaining amount available to invest is $200.
  - **Interest rate** - the interest rate of the investment account per annum (the interest rate needs to be between 1% and 20%; otherwise the program will show an error message and ask the user for the interest rate again)
  - **Investment length** - number of weeks that the user will invest the money (it needs to be a positive number and different from zero; otherwise the program will show an error message and ask the number of weeks again)

- <u>Display account.</u>
  The program will ask the name of the client. If the name does not exist, show a message. If the name exists, then the program will ask the account number (1 or 2, which is the first or second position in the array). If the account does not exist, the program will display the message "the account does not exist". If not, the program will display the following data about the client and account:

  - a table with details of the investment at four week intervals for the length of the investment. For the purposes of this assignment you will assume that interest is only applied at the end of a complete four week period, see the example below. Exactly as assignment 1.

- Delete Account
  The program will ask the name of the client. If the name does not exist, show a message. If the name exists, then the program will ask the account number (1 or 2, which is the first or second position in the array). If the account exists, then the program will delete the account and the program will display the message “the account was deleted”. If the account does not exist, the program will display the message “the account does not exist”.

- Save in a file
  The program will save in a file (txt format) the information of each client (the same information described in “show a client” above) and each account (the same information described in “display account” above). If there are no clients, the files should have “no client”. For each client, if the client does not have an account, the file should have “no account”.

- Exit. The program will end.

Program Requirements

There must be **at least three** classes: Client, Account and CalculatorInterface.

- **The Account Class** (the file needs to be **Account.java**)

  You must have the following instance variables (you can include others if you feel they are necessary) and all instance variables must be private.
    - rate – double
    - numberOfWeeks – int
    - amount – double

  You need to implement at least two constructors. One will be the default one, it will initialize the variables with 0 or “”. The second constructor will have the parameters rate, numberOfWeeks and amount. The class needs to have methods to change and access all instance variables. You will decide which will be the other methods to be implemented in this class.

- **The Client Class** (the file needs to be **Client.java**)

  You must have the following instance variables (you can include others if you feel they are necessary) and all instance variables must be private.
    - name – a String
    - grossSalary – double
    - netSalary – double
    - resident – boolean
    - tax – double
    - medicare – double
    - weeklyExpenses - double
    - account – one-dimensional array of Account objects (size 2)

  It will hold the required instance data for a client and it will have suitable methods to access and modify the data for a client. You will decide which will be the other methods to be implemented in this class.

- **The CalculatorInterface Class** (the file needs to be **CalculatorInterface.java**)

  You must have the following variables (you can include others if you feel they are necessary)
    - clients – one-dimensional array of Client objects (size 5)
    - noClients – the number of clients (it will start with zero). *This variable can be declared in the Client class as static variable.*

  You will decide which methods will be implemented in this class.
  This class will create the interface (Terminal IO or GUI, you choose). Only this class will
    - Receive the inputs from user and show the outputs.

- Validate input (ie. Negative numbers etc).
- Show error messages.

Notice also that only this class needs to be modified if you decide to implement more than one interface (TIO and GUI).

Marks will be awarded for layout (including visual aspects (variable names, indentation) and structural aspects (variable scope, method usage)), documentation (comments), and the submission's ability to perform as specified. A more detailed marking schema is available on Canvas.

What to submit

You should submit only the three .java files (Account.java, Client.java, CalculatorInterface.java), in a compressed .zip file, via the "Assignment 2" link on Canvas. Do not include .class files in your submission.

Add your student name(s) on the top of each Java file submitted. If you are completing the assignment as a group (only SENG1110 students), add both names in each Java file. Follow the following template.

```
/*Author: name
 *Student No: XXXXXX
 *Date: 02-03-2022
 *Description:
 */
```

Extra Work for SENG6110 students

When the user chooses the option "display all clients", the program will display the information sorted by name. You need to implement a sort algorithm.

SENG1110 students that implement the SENG6110 task will receive bonus marks (note though that you cannot receive more than 100% for the assignment when the bonus is applied).

Late Penalty and adverse circumstances

Note that your mark will be reduced by 10% for each day (or part day) that the assignment is late. This applies equally to week and weekend days. You are entitled to apply for special consideration if adverse circumstances have had an impact on your performance in an assessment item. This includes applying for an extension of time to complete an assessment item. See https://www.newcastle.edu.au/current-students/learning/assessments-and-exams/adverse-circumstances for more details.

In Canvas you will find a new forum in the discussion board: "assignment2". Any question about the assignment 2 you can post there. Check this forum regularly.

Prof Regina Berretta
May- 2022