SOFT 6017 PROJECT

- Due: **Sunday 22nd April @ 23:59**
- Submit your code & evidence of using **Git** through **Blackboard**
- Oral **assessment** & **demonstration** in week 12
- Worth: **25%** SOFT6017's overall mark
- Code will be tested for **plagiarism** (refer http://www.cit.ie/aboutcit/reports_plansandpolicies/academic for CIT policies)

Specification

Data for a bank is kept in a file called `bank.txt`.

The file stores bank records and each bank record consists of

- a bank account number
- a balance in the account
- a name

A snippet of the file might be:

```
98768
8077.00
Jim McIntyre
58697
23233.99
Michael Murphy
35318
4545.12
Abigail Buckley
20454
23233.45
Marie Delaney
```

Or if you prefer the data on one line might be

```
98768 8077.00 Jim McIntyre
58697 23233.99 Michael Murphy
35318 4545.12 Abigail Buckley
20454 23233.45 Marie Delaney
```

Develop an application that allows the user to choose from one of the following options, until they choose to quit.

1. Open an account
2. Close an account
3. Withdraw money
4. Deposit money
5. Generate a report for management
6. Quit

Details

1. When opening an account a new account number must be created. It is a random 6 digit number which is not already used as a bank account number.
2. Search for a bank account by its number and delete all data associated with that bank account.

3. Search for a bank account by its number and withdraw a user-specified amount of money if there are sufficient funds.
4. Search for a bank account by its number and deposit a user-specified amount of money.
5. Print the following details:
    a. Details of all accounts
    b. Total amount on deposit in the bank
    c. Largest amount on deposit specifying the account holder(s)
6. Quit and write the data from the list(s) back to bank.txt

Advice

After completing the files/lists lab, you should be able to begin coding by reading the data from the text file into three lists.

The lists will be edited while the program runs e.g. when **creating a new account**, extra data is appended to each list; when **deleting an account** data is removed from each list; when **depositing** or **withdrawing money**, the money list is edited. We have completed many useful examples of processing lists using in-built functions (`.sort(), .index(), max(), sum(), len()` etc) and functions you write yourself, some of which you should be able to **re-use**.

Once the user chooses to **quit the program** (*but not before*), the data in the lists are written back to the file so that any changes are permanent.

Version Control System (VCS)

As part of programming a large application it is important to use a version control system to keep track of the changes you make to a program. VCS allow you to revert to an old version if you need to do that. We have chosen **Git**, which is an industry standard. Each time you add functionality, commit that change to Git. At the end of the project, take a **screenshot** of the results of `git log` and `git log --oneline` commands – this will list all the changes in full and brief format, as evidence of how you applied version control.[1]

[1] Git will be covered in a lab after Easter. If you are working on the project during the Easter break, you can begin to use Git after Easter on whatever functionality remains – 5/6 commits are sufficient to demonstrate your knowledge of Git. If you complete the project during the Easter holidays, you will be assigned a small exercise to complete to demonstrate your knowledge of Git.

Marking Scheme

Quality of code	• naming conventions • comments • use of whitespace • use and re-use of functions/modules	10%
Version Control	• implementation of git for project	10%
Functionality	• menu • reading from file into lists • write from lists into file • open an account • close an account • withdraw money • deposit money • statistics	80%

Git Reference

After completing the Version Control lab in Week 9, you will be ready to use Git. Open your project folder using the GitBash console, and initialise that folder as a git folder. When you add functionality / documentation to your project, commit these changes to Git through GitBash console.

`git init` – start Git

`git add .` – add all files in the folder to the staging area

`git commit -m "describe the reason for the edit"` - commit a file

`git log` – list the details of changes made to the project

`git log -oneline` – list a summary of changes made to the project