University of HUDDERSFIELD

CFS2160: Programming Stream

Tutorial/Practical 16

More Inheritance

Introduction

There is no Log Book this term.

The exercises in the lab will gradually build up a library of code that you will be able to draw on when you start work on the second assignment.

Activities

Make sure your work so far this term is up to date. If you need to ask about anything, do so.

By now you have developed a collection of classes that could have been used in some banking application. The spec was as follows:

1. A `CurrentAccount` class that has a balance, allows deposits and withdrawals, may allow an overdraft, but does not pay any interest. (You probably have most of this class already.)
2. A `DepositAccount` that is the same as a `CurrentAccount`, but which pays interest and which cannot go overdrawn.
3. A `StudentAccount` that has a fixed £500 overdraft.
4. A `YoungSaversAccount` that is the same as a `DepositAccount` but cannot have a balance of more than £100 (unless this happens because of interest being added).

Your solution to this should now use inheritance.

Now implement a class for the Bank itself (or extend your work from last week if you had time) - call it `Bank` - that contains a collection of bank accounts. Include methods that will:

- Display the sum of all the balances of these accounts (you should have this from last week).
- Display the details of every account in a reasonably neat table. Note that the details for each account are different and all relevant details should be shown, so there will need to be some overriding in order to achieve this[1].

[1] But **think**. You probably do not need to add some display method to *every* class. To do this neatly, there is a programming trick, which your friendly tutor will be happy to reveal.

- Display the table as above, but only for those accounts that are overdrawn (that is, they have a negative balance).
- Display the details as above, but *only* for student accounts.