COMP101 Lab 9: Sixth Assessed Coursework
Inheritance

worth 16% of the final mark
assignment six of seven

Failure to submit this assignment or submission of work that is deemed not to be a reasonable attempt will result in the failure of the module. The completion of the implementation and report as described below will obtain 90% of the marks. To obtain the final 10%, students should also complete the Extended Requirements. This may involve doing some additional reading beyond what has been presented in lectures or more complex programming. Only attempt the extended requirements if you are confident with programming.

Learning Outcomes. This addresses the following learning outcomes.

- Be able to implement, compile, test and run Java programmes, comprising more than one class, to address a particular software problem.
- Understand how to include arithmetic operators and constants in a Java program;
- Demonstrate the ability to employ various types of selection constructs in a Java program.
- Demonstrate the ability to employ repetition constructs in a Java program.
- Be able to employ a hierarchy of Java classes to provide a solution to a given set of requirements.

Requirements. The costs and features for three broadband, TV, and phone packages for a cable company are summarised below.

	Bronze	Silver	Gold
Package Costs (monthly)	£36.00	£46.00	£61.00
Daytime phone costs (per minute)	12p	12p	0p
Evening/Weekend phone costs (per minute)	5p	0p	0p
Number of TV Channels	60	130	230
Broadband (included)	500Mb	1000Mb	1520Mb
Broadband (per Mb above included amount)	2p	1p	1p

Additionally, the Silver and Gold accounts provide a free Spotify account, and Gold accounts include music on demand. This information can be used by potential customers to decide which package is the most suitable for them given their recent phone call and broadband usage. Use Java's inheritance mechanism to implement classes to store this information. The

Bronze Account should be at the root of the inheritance hierarchy. Your implementation should include methods that return the type of the account (e.g. Bronze, Silver, or Gold). You should also include methods that print the account details.

A user should be able to input the number of daytime phone minutes that have been used, the number of evening/weekend minutes that have been used, and the amount of broadband usage, expressed in Mb (megabytes), for that month. These should be integer values, greater than or equal to zero (i.e. negative values should be disallowed and the user asked to re-input these values). For each account (Bronze, Silver, and Gold), the account information should be printed, the total cost of each type of call should be calculated and printed, the total cost for (extra) broadband usage, and the total cost of that account (made up from the call costs, broadband costs, plus the package costs) should be calculated and printed. Additionally, the program should output which account(s) (Bronze; Silver; Gold; Bronze and Silver; Bronze and Gold; Silver and Gold; or all three) has the total cost that is the cheapest.

Example.

```
$ java AccountUser
Input daytime telphone minutes used:
100
Input evening and weekend telephone minutes used:
100
Input the total broadband usage (in Mb):
600

Account Summary for Bronze Account
Package Cost: 36.00
Cost of daytime calls: 0.12/min
Cost of evening and weekend calls: 0.05/min
Number of Channels: 60
Broadband Included: 500Mb
Total daytime calls cost: 12.00
Total evening calls cost: 5.00
Total (extra) broadband cost: 2.00
Total cost: 55.00

Account Summary for Silver Account
Package Cost: 46.00
Cost of daytime calls: 0.12/min
Cost of evening and weekend calls: 0.00/min
Number of Channels: 130
Broadband Included: 1000Mb
Total daytime calls cost: 12.00
Total evening calls cost: 0.00
Total (extra) broadband cost: 0.00
Total cost: 58.00
Spotify Account provided
```

```
Account Summary for Gold Account
Package Cost: 61.00
Cost of daytime calls: 0.00/min
Cost of evening and weekend calls: 0.00/min
Number of Channels: 230
Broadband Included: 1520Mb
Total daytime calls cost: 0.00
Total evening calls cost: 0.00
Total (extra) broadband cost: 0.00
Total cost: 61.00
Spotify Account provided
Music on Demand provided

The Bronze Account is cheapest.
```

Hints.

1. You will need (at least) four classes: the three mentioned above (BronzeAccount, SilverAccount, and GoldAccount) and an application class.
2. The classes BronzeAccount, SilverAccount, and GoldAccount should be arranged in a class hierarchy. Your software should demonstrate both inheritance and overriding.
3. You should check the input number of minutes (for daytime and evening/weekend calls) and the amount of broadband usage, and disallow the input of negative values and ask the user to re-input these. You can assume that each of these values is an integer.
4. Before proceeding with the Regular Requirements, please read the Extended Requirements below. If you submit the Extended Requirements, you should not submit files for the Regular Requirements. But make it clear in your report that your files submitted are to satisfy the Regular Requirements or the Extended Requirements.

Extended Requirements. Re-implement (or implement from the beginning) your code in two ways as follows:

- Firstly, add another class *StandardAccount* which is an abstract class at the root of the hierarchy. Some methods should be declared as abstract, for example, the one that returns the type of each account.
- Additionally, add methods that calculate (approximate) profit for each account. This method should also be declared abstract in the *StandardAccount*. The approximate profit for each account is calculated as follows.
  - Bronze account: £15.00 plus 5p for every daytime minute, 2p for every evening minute, and 1p for each Mb above the alloted account amount of 500Mb;
  - Silver account: £20.00 plus 5p for every daytime minute minus 1p for every evening minute;

– Gold account: £25.00, minus 3p for every daytime minute, minus 1p for every evening minute;

The (approximate) profit for each account should also be printed. Print out this information after the information for the customer (that is, print out the customer information for all three accounts first, then print out the profit for each of the three accounts).

As stated, if you submit files for the Extended Requirements, do not submit files for the Regular Requirements, and make it clear in your report which set of requirements your files are to supposed to satisfy.

Submission Instructions. You must submit all of your files (PDF report and Java source files) as a single compressed zip file. Do not use any compression method other than zip. (Note that this is the compression method that the Windows and Mac file managers use to compress files.)

Your submission, should consist of a report (a PDF file) and implementation files.

- The report (a PDF file) should consist of

  Requirements: Summary of the above requirements statement.

  Analysis and Design: A short (one paragraph) description of your analysis of the problem including a Class Diagram outlining the class structure for your proposed solution, and pseudocode for key methods. Note that your solution should comprise (at least) four classes: an "application class" and three (or more) "target classes".

  Important: Make certain that your main application class (the class with the "main" method) is called "AccountUser".

  If you do the extended requirements, make certain that your main application class is called "AccountUserExt".

  Recall that you are submitting either files for the Regular Requirements or the Extended Requirements, not both. So you should have one application class file called either "AccountUser.java" or "AccountUserExt.java", but not both.

  Testing: A set of proposed test cases presented in tabular format including expected output, and evidence of the results of testing (the simplest way of doing this is to cut and paste the result of running your test cases into your report).

  Make certain that your test cases reflect the set of requirements (Required or Extended) that your code is supposed to satisfy.

- The implementation should consist of

  Your Java source files, i.e. the relevant .java files, not the class (.class) files.

Upload your files (as a single compressed zip file) to

https://sam.csc.liv.ac.uk/COMP/Submissions.pl

(you will need to log in using your Computer Science username and password).

Submission Deadline. Monday 5th December, 5pm.

Mark Scheme. Marks will be awarded for

- Analysis and Design 15%
- Implementation 60%
- Testing 15%
- Extended requirements 10%

Please see the module web page for the feedback form.

Note.

- Because submission is handled electronically, ANY FILE submitted past the deadline will be considered at least ONE DAY late. Late submissions are subject to the University Policy (see www.csc.liv.ac.uk/department/regulations/practical.html).
- Please make sure your Java classes successfully compile and run on ANY departmental computer system. For this reason, the use of powerful IDEs like NetBeans is discouraged.
- Note this is an individual piece of work. Please note the University Guidelines on Academic Integrity (see https://www.liverpool.ac.uk/media/livacuk/tqsd/code-of-practice-on-assessment/appendix_L_cop_assess.pdf).