

|  |
| --- |
| **COURSEWORK ASSESSMENT SPECIFICATION** |

|  |  |
| --- | --- |
| **Module Title:** | *Programming 1* |
| **Module Number:** | *CG0047* |
| **Module Tutor Name(s):** | *Alan Maughan* |
| **Academic Year:** | *2014/15* |
| **% Weighting (to overall module):** | *20%* |
| **Coursework Title:** | *Homework 4* |
| **Average Study Time Required by Student:** | *8 hours* |

**Dates and Mechanisms for Assessment Submission and Feedback**

|  |
| --- |
| **Date of Handout to Students:**  Week 9 |
| **Mechanism for Handout to Students:**  *via elp* |
| **Date and Time of Submission by Student:**  During Week 10 Lab Class |
| **Mechanism for Submission of Work by Student:**  Papers collected in lab |
| **Date by which Work, Feedback and Marks will be returned to Students:**  Marks & feedback will be given as the assessment is marked in the lab week 10. |
| **Mechanism for return of assignment work, feedback and marks to students:**  Marks & feedback will be given as the assessment is marked in the lab |

**Further Information**

*(Please ensure the assessment specification includes the following items)*

**Learning Outcomes tested in this assessment (from the Module Descriptor):**

1. Design a program from a specification;
2. Formulate solutions to a number of basic programming problems using an appropriate design notation;
3. Make effective use of basic data types arrays and structured programming control constructs: sequence, selection and iteration.
4. Understand and make basic use of functions/procedures.

**Assessment Criteria/Mark Scheme:**

**NOTE:** Program must compile otherwise only marks for coding style are available.

|  |  |  |
| --- | --- | --- |
| Category | Mark | Description |
| Coding style | 2 | Use of header comments  Use of method comments  Layout and naming convention  Visibility |
| AccountList class  **(Must contain student name in @author tag or 0 marks for class!)** | 1 | Use of appropriate UI methods: menu, displayMenu, etc |
| 6 | addAccount, getNumberOfAccounts, showAccount showAllAccounts, removeAccount, quitCommand |
| AccountDriver class | 1 | Program runs from runs from the driver class’ main method. |

**Nature of the submission required:**

Paper copies of source code. Code execution in lab.

**Instructions to students:**

*This is an individual piece of work.*

**Referencing Style:**

*N/A*

**Expected size of the submission**:

Under 10 pages – printed copies of source code

**Academic Conduct:**

You must adhere to the university regulations on academic conduct. Formal inquiry proceedings will be instigated if there is any suspicion of misconduct or plagiarism in your work. Refer to the University’s regulations on assessment if you are unclear as to the meaning of these terms. The latest copy is available on the university website.

# Homework 4

This work is due for marking at the start of your lab in week 10. It counts for 20% of the overall module mark.

You must bring with you a printed copy of your source code (the .java files) for the classes **AccountList**, **AccountTUI** and **AccountDriver.** These should be produced before you come to the lab. Do not come to the lab and attempt to print copies then. These will be collected by the tutor when they mark your homework – make sure that they have your name / id on them. These will be retained for audit and internal moderation. If these files are not submitted (and printed before the lab) then you will score 0 (zero)!

You must work on the program on your own, outside any formal classes and it must be ready to execute at the start of the scheduled laboratory class. All code must be completed using the BlueJ IDE. Any work utilising other IDEs will score zero.

All code must:

* Have the class header and all methods commented to ‘Javadoc’ standards using @author, @version, @param and @return tags as appropriate.
* Be coded to required layout (e.g. indentation) and naming standards.

Notes on the above were supplied in week 1 and there are numerous examples in the code you will have seen. Failure to meet these standards will result in loss of marks.

You may be asked questions about your program to confirm your understanding and that it is your own work. Failure to answer the questions may result in a deduction or total loss of marks.

***The work must be wholly your own. Collusion counts as academic misconduct and will be punished according to the University’s regulations detailed in “Assessment Regulations for Northumbria Awards” (ARNA) a copy of which is available on the University website.***

## The Task

Create a project which should be called “HW4”. For this homework you are expected to write a program based upon the **Account,** **Address** and **AccountList** classes supplied on Blackboard.

The class **AccountList on** Blackboardhas to be amended to include a further method **getAccount(String accountNumber)**.

You are expected to write:

* a class **AccountTUI** which represents a text-based user interface to the application and should manage use of **AccountList**; and
* a driver class **AccountDriver**.

**Note:**

1. Only the creation of account pobjects via the following constructor is expected:

**public Account(String firstName, String lastName, String accountNumber,**

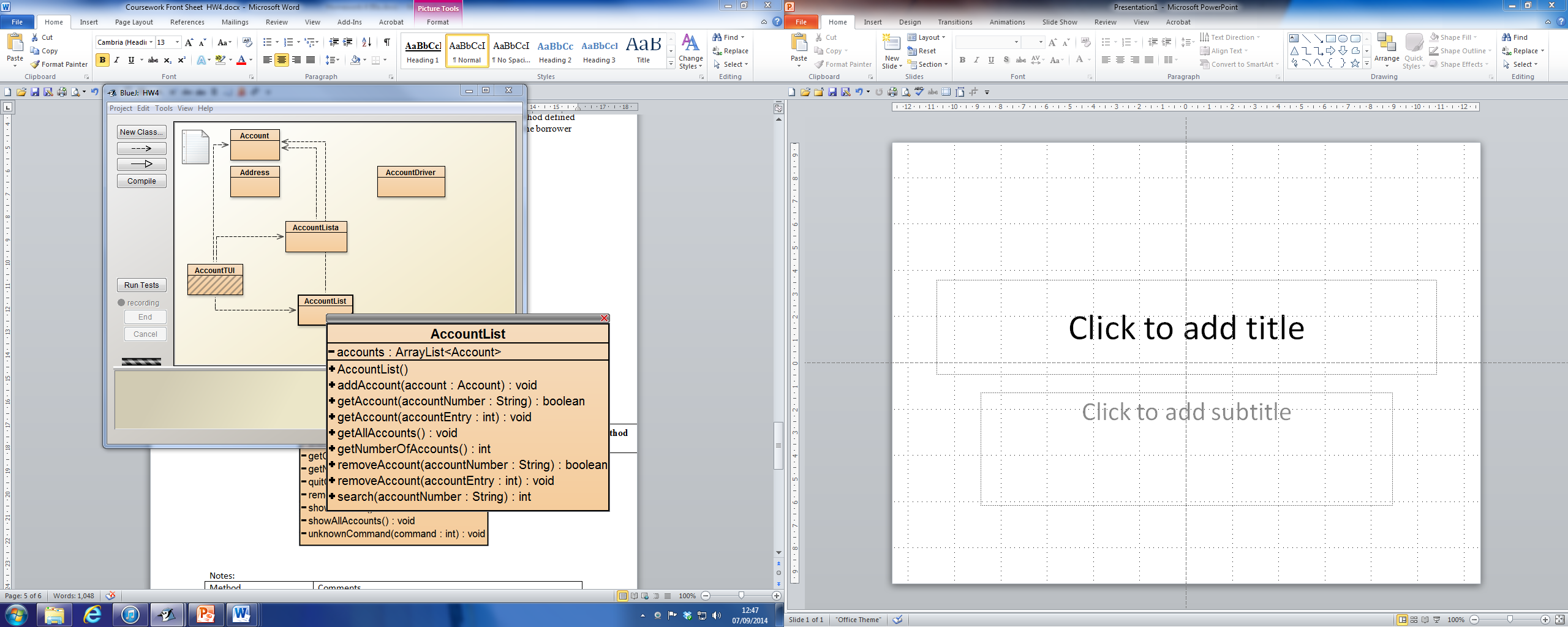
**String street, String town, String postcode)**.

**Task 1**

You are expected to amend the **AccountList** class given on Blackboard to include a new method:

**getAccount(accountNumber : String) : boolean**

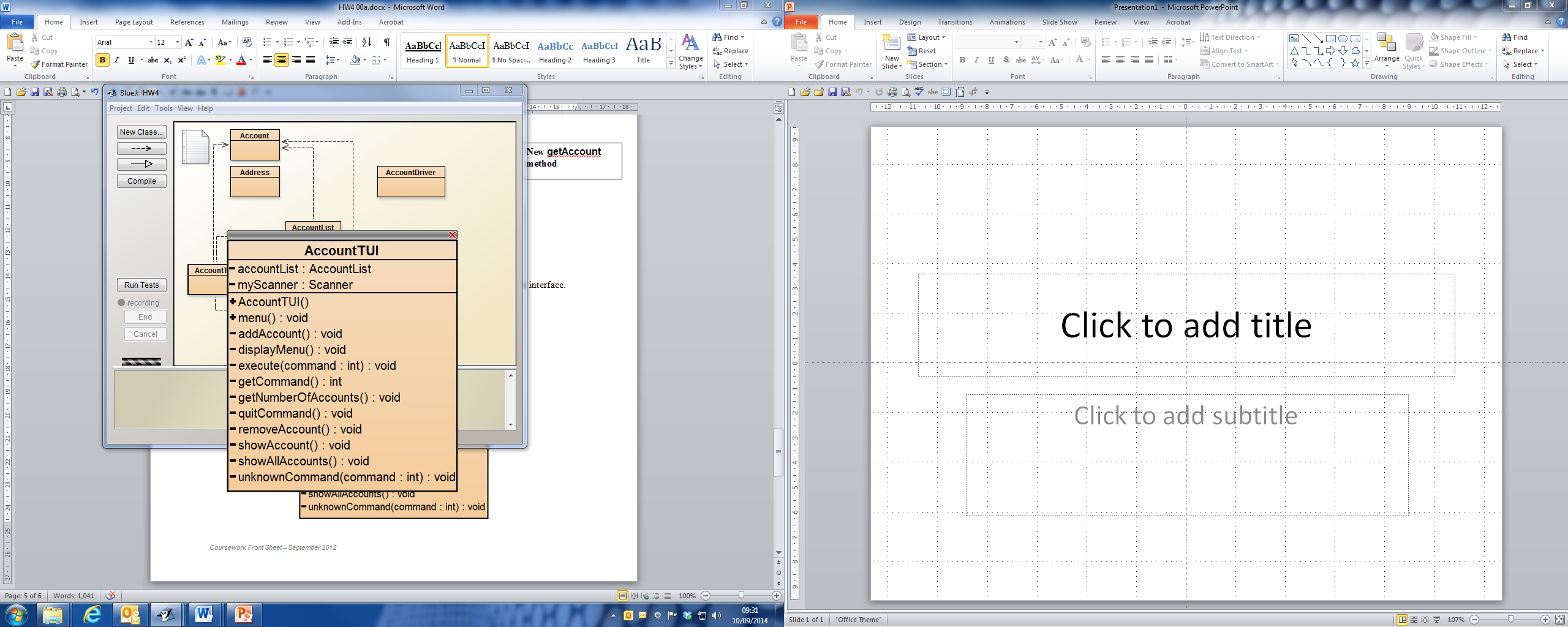
This method is similar to the existing **getAccount** method but should receive a string representing the account’s account number and should print details of a account with the given account number to the conslole window via the **printAccountDetails** method defined in **Account**. If the number is found then it should then return true. However if the account is not found it should return false. The ***revised*** class diagram will be:

****

**New getAccount method**

**Task 2**

You are now expected to write a class **AccountTUI** to provide a text-based user interface. The expected class should be:



Notes:

|  |  |
| --- | --- |
| Method | Comments |
| menu  displaymenu  execute  getCommand | Expected TUI methods. |
| addAccount | Creates a new account. |
| getNumberOfAccounts | Return the number of accounts in a formatted String  e.g. “We have 7 accounts.” |
| quitCommand | Produce a message that application is closing then call System.exit(0) |
| removeAccount | Remove a account with the given account number. If successful output “Account with account [account No] removed.” If not then: “Account with account [account No] NOT removed.” |
| showAllAccounts | Print out all accounts. Space should be left between each to make it easier to read. |
| showAccount | Print out account with chosen account number. If method is unsuccessful then a message “Could not find account.” should be output. |
| unknownCommand | Prints out “Unknown command.” if option is not between 0 – 5. |
|  |  |

**Task 3**

You should also provide a class “AccountDriver” containing the **main** method.