COSC2511 Introduction to Programming

Assignment, Semester 1, 2017 STAGE 1 (Version 1)

A component based marking scheme. The biggest confusion is delivering an end application whereas the program component workings are the most valuable part of this stage. The project can be in pieces and individual parts tested, and get top marks. Prototype code should be an involved consideration of the problem in more depth than your class exercises.

You can do anything in Java that you like, or use your current skill set and implement features. There is no limit on marks, as additional features can be added. You are free to `incorporate' code from the internet – but as your own. e.g. adding file read and write to save your application state can be a major additional feature (the interviews at each of the two stages help validate this)

However, in practise a considerable amount of work is needed to implement additional features. Your code is general the primary source of the projects marks at any stage.

Stage 1 due Week 10- Week 12
Requires:

- Problem formation and description
- communication of solo or group work and members (in Week 9)
- competed in-class interview (needs to be arranged: Week 10-12)
- 1 page report *(demonstrated in interview)*
- Original data (the problem should have its own associated data)
- working Java prototype code *(demonstrate in interview)*
- + optionally other documents
- Code draft required before interview

All documents submitted through gradecentre before the interview. If non-attendance of interview, no marks for the report stage as no verification of work.

Dates: now + Week 9/10; Week 12 starts 1-May

Resources: the class examples provide many coding resources

If group work, all members share the same mark.

Default project/theme : not sure if we will have one. An inventory type application is always possible, and can employ the array data structure and other programming. Text console menus allow for submenus and a more detailed application. Games of any sort – can use random number generation to include chance. If you run out of ideas, think bigger, then you will always have more features to implement.

Marking Criteria	Marks allocated
Problem description (can be verbal in demo)	10
Menus	5
Storing test data in the application	10
Program logic	20
Non-trivial data structures (e.g. arrays)	10
Functions and methods (not monolithic or badly coded)	20
Minor feature	+2.5
Moderate feature	+5
Major feature	+10
Testing	Variable 2.5-5.0
Documents (some projects can be assessed with documentation if the project is agreed to by the assessor)	Variable on project
Other marks are possible in project	

N.B. Almost never is a project rejected. Sometimes projects are modified – this is always for the benefit of the student. Advice base on the marking scheme may be given. e.g. if there is not enough work with the current project - this is a choice. Or conversely, adding testing to improve the mark. Occasional there may be recommendations to consolidate work, if some projects from getting out of hand. If you have a difficult first phase project, you can change this in the second phase. After the interview, a second draft code may be submitted.