MSc in High Performance Computing

MSc in High Performance Computing with Data Science

Software Development Coursework

Session 2016/17

Introduction

The assessed coursework for Software Development is intended to apply and evaluate the processes and techniques covered in the course. The coursework should reinforce the points made in the course and allow you to realise issues that come in applying theory in practice. This does not mean that you have to stick only to the techniques in the course; prior experience and techniques are valid (where proven to work, are applicable and reliable). The aim of the assessment is to look beyond just the coding aspects of software development, to consider the software development process as a whole.

Dates

Key dates in the assessment are as follows:

- Assessment available: Monday 16/01/2017
- Assessment Part 1 (Planning and Risks) deadline: 12pm Friday 10/02/2017
- Assessment Part 2 (UI and Evaluation Plan) deadline: 12pm Monday 06/03/2017
- Assessment Part 3 (Refactoring and Reflection) deadline: 12pm Friday 31/03/2017

Feedback and provisional marks will be returned within fifteen working days of the deadlines above.

Late Submissions Policy

The late submissions policy follows the process outline in the student handbooks. This information can be found at:
https://www.wiki.ed.ac.uk/display/SPTH/Late+submission+2016-17

Tasks

The assessment is broken into three submissions across the semester. This will allow the opportunity for you to receive feedback between submissions.

The three components to the coursework are:

- Planning and Risks
- UI and Evaluation Plan
- Refactoring and Reflection

Note - do not submit all parts at the same time, this will not get marked any sooner and you will miss the opportunity to improve on past performance.

1. Planning And Risks

Your first task is to analyse the code provided as part of this assessment and its related specification, identify problems, and for each problem you identify, propose a solution. Once you have a set of problems and proposed solutions, create a plan for implementing

some of these solutions, including estimates of the time and effort required for each solution. Your plan should contain a risk analysis for the work you propose.

Your first submission is a document summarising the following:

- Identified Problems
- Solutions
- Time/Effort Estimation Plan
- Risk Analysis and Management Strategy

Note: you are not required to propose a plan to fix every problem in the code - but the solutions you plan to implement must be justified and must make a significant improvement to the code.

Note: you can ask the course organiser if what you are planning to do would constitute a significant improvement.

Your document should include a Gantt chart or similar plan to illustrate how you have planned your workload.

This component of the submission is worth 30 marks out of 100 marks total for the course.

2. UI and Evaluation Plan

The supplied code uses a web interface (implemented in HTML and JavaScript). Your second task is to design and prototype a new and improved user interface (UI). Your submission should explain the reasons for your UI design choices.

Note: your prototype does not have to be a functional UI prototype. The UI can be in any form you want to design but it must have an accessible design such that it can be evaluated.

Create a plan for how you will evaluate the usability of your UI design. The plan can be done in any fashion, but it must be clearly laid out to show what is being evaluated, how it is being evaluated, who is evaluating it and how the results will be interpreted.

Your second submission should contain:

- UI prototype and design
- Summary of design choices
- UI evaluation plan

This component of the submission is worth 30 marks out of 100 marks total for the course.

3. Refactoring and Reflection

Your third task is to implement the solutions you proposed in your first submission, Planning and Risks. The main part of this submission should be a new version of the code. Your submission should include any required files such as build and/or test files.

The submitted code must be runnable and be able to be used to fulfil the purpose in the specification.

Note: The code submission should be a link/reference to a source code revision control system such as GIT or SVN.
The repository should contain everything needed to run the code - including build files to acquire any required dependencies. Non-working or non-compiling code will be marked down accordingly.

Please note that your code (and any supplied tests) must run in a standard Windows, MacOS or Linux environment – installing any dependencies if required. Any queries on this should be directed to the course organiser.

You should provide a short report to accompany the code containing a summary of the changes to the code that you have made and the benefits they give to the code. The report should also contain your personal reflection on the whole process, including how your actual work compared with the plan you created at the start and whether you encountered any new risks you had not considered.

The short report should include a future enhancement you think would improve or add value to the code - you should summarise the purpose of your proposal and highlight any risks or impact that it may have on the existing code.

Your third, and final, submission should contain:

- Improved Code
- Summary of Changes
- Reflection on Process
- Future Enhancement

This component of the submission is worth 40 marks out of 100 marks total for the course.

Specifications

The application under development is a squad roster builder for a tabletop miniatures skirmish game, a type of war game. The game is based around the movement of individual squad member and actions by these squad members (e.g. shoot/fight). The success of actions is based upon the roll of a D20 (20 sided dice).
In the game, each player has one of up to 10 individuals. The majority of these individuals are standard soldiers with specialisms in some skill (e.g. engineering, psychology or defence, see below). Each squad is led by a Captain, who can gain experience after each battle that earns them upgrades to skills and stats. An optional squad member is the Ensign who is like the Captain, earlier on in their career. Over time the Ensign can become a powerful member of the squad.

Each player starts with a squad with only a Captain (who is free) and 500 credits. These 500 credits can be spent to buy more squad members and equipment for the Captain and, if present, Ensign. After each battle, the squad will earn credits and gain skills so the amount of credits can go up as well as down.

Each squad can have a maximum of ten members (this includes the Captain and Ensign).

Both Captain and Ensign can each have a specialism allowing them access to some skills that can be used during the game. These are selected when a squad is created and augmented as their experience grows.

Each squad needs to have a name and the name of the player that plays with it.

A squad can never have negative money in the bank – this would be illegal under the game rules.

Each squad member has a cost, stats (representing their abilities, see below) and notes. The Captain and Ensign each have a specialism, weapons, skills and experience as well.

The application should allow players to create and edit squads in the game. The application should ensure validity of the squads according to the game rules above. The application should allow players to add money to their squads, change the roster and increase the abilities of their experience-gaining characters (the Captain and Ensign).

Players should be able to add new squad members and remove old squad members. It is important to factor in the cost of new squad members. After the initial roster creation, if a player removes a squad member, they do not receive any compensation for this – no refunds – so between battles you cannot boost funds by dropping squad members.

The application should allow for a player to delete their own squads but not those of another player. Additionally players should not be allowed to view other players' squads unless the squad has been marked public.

The application should allow for multiple users at one time to use the software.

All calculations should be checked for consistency.

Squads can have a stash of items and weapons stored away.

The stats for a squad member, representing the member's abilities, are:

- Move – Distance the member can move, in inches.
- Fight – Modifier for any D20 rolled in a melee or defence action (positive or negative), representing the fighting strength of this member
- Shoot – Modifier for any D20 rolled in a shooting action (positive or negative), representing the shooting strength of this member
- Shield – Value a weapon must exceed to wound the member .
- Morale – How likely the member will keep fighting.
- Health – How much damage can the member take before dying.
- Cost – How many credits the member costs in the game.

The notes for a squad member provide additional information e.g. a listing of special characteristics and equipment.

Captain and Ensign stats also include:

- Experience – Number of experience points that have been gained.
- Specialism – What is the specialist skill set the member has access to.
- Skillset – Skills currently held by the member.
- Items – Weapons and equipment.

When a captain or ensign gains experience they can increase one of the following stats: Fight, Shoot, Health, Morale or gain a new skill from their specialism.

Captains can carry up to 6 items, 2 of which can be weapons.

Ensigns can carry up to 4 items, 1 of which can be a weapon.

For each 10 experience points, the Captain of Ensign can increase a stat value by one or gain a skill. No stat can be increased by more that one after each battle.

The maximum values for the stats that can be changed for a Captain are:

- Fight: +8
- Shoot: +8
- Health: 20
- Morale: +8

The maximum values for the stats that can be changed for an Ensign are:

- Fight: +6
- Shoot: +6
- Health: 16
- Morale: +4

Each time a squad is saved or created it must be validated against the stipulations listed in this specification.

Equipment and their costs are as follows:

- Blaster: 5
- Needle Gun: 12
- Blade: 3
- Cannon: 15
- Whip: 5

The specialisms and associated skills are:

- Engineering: [Repair, Sabotage, Augment]
- Psychology: [Bolster, Terror, Counter]
- Marksman: [Aim, Pierce, Reload]
- Tactics: [Squad, Ambush, Surround]
- Melee: [Block, Riposte, Dual]
- Defence: [Shield, Sacrifice, Resolute]

Baseline stats for different types of squad member are as follows:

Captain Stats:

- Move: 5
- Fight: 2
- Shoot: 2
- Shield: 12
- Morale: 4
- Health: 12
- Cost: 0
- Skillset: []
- Specialism: None
- Items: []
- Experience:0

Ensign Stats:

- Move: 7
- Fight: 0
- Shoot: -1
- Shield: 10
- Morale: 2

- Health: 8
- Skillset: []
- Specialism: None
- Cost: 250
- Items: []
- Experience:0

Squad Members:

- Augment Gorilla:
  - Move: 8
  - Fight: 3
  - Shoot: 0
  - Shield: 10
  - Morale: 2
  - Health: 8
  - Cost: 20
  - Notes: Animal, Cannot carry treasure or items
- Lackey:
  - Move: 6
  - Fight: 2
  - Shoot: 0
  - Shield: 10
  - Morale: -1
  - Health: 10
  - Cost:20
  - Notes: Melee Weapon
- Security:
  - Move: 6
  - Fight: 2
  - Shoot: 1
  - Shield: 12
  - Morale: 2
  - Health: 12
  - Cost: 80
  - Notes: Blaster, Blade
- Engineer:
  - Move: 4
  - Fight: 0
  - Shoot: 3
  - Shield: 12
  - Morale: 2
  - Health: 10
  - Cost: 60
  - Notes: Blaster, Repair Kit
- Medic:
  - Move: 5

    - Fight: 0
    - Shoot: 0
    - Shield: 12
    - Morale: 3
    - Health: 10
    - Cost: 50
    - Notes: Blade, Medkit
- Commando:
    - Move: 8
    - Fight: 4
    - Shoot: 0
    - Shield: 10
    - Morale: 4
    - Health: 12
    - Cost: 100
    - Notes: Stealth Suit, Blade, Needle Gun
- Combat Droid:
    - Move: 3
    - Fight: 2
    - Shoot: 4
    - Shield: 14
    - Morale: 0
    - Health: 14
    - Cost: 150
    - Notes: Mechanoid, Dual Blaster, Claws