UNIVERSITY OF
LINCOLN

Lincoln School of Computer Science

Assessment Item Briefing Document

Title: CMP1127M Programming and Data Structures: Assessment 2	**Indicative Weighting: 50%**

Learning Outcomes:

On successful completion of this assessment item a student will have demonstrated competence in the following areas:

- [LO2] apply appropriate data structures in common programming solutions;
- [LO3] implement programs consisting of multiple procedures;
- [LO4] apply simple testing techniques.

Requirements

This assignment asks you to design and implement the 'Three or More' dice game as described on http://icebreakerideas.com/dice-games/

Directions for play:

Players take turns rolling five dice and score for three-of-a-kind or better. If a player only has two-of-a-kind, they may re-throw the remaining dice in an attempt to improve the remaining dice values. If no matching numbers are rolled after two rolls, the player scores 0.

Scoring:

- 3-of-a-kind: 3 points
- 4-of-a-kind: 6 points
- 5-of-a-kind: 12 points

The player who reaches a fixed value of points (say 50) is the winner.

In your solution, you should implement:

- A 'Game' class
- A 'Player' class
- A 'Die' class
- Any other classes that you feel may be appropriate.
- At least one data structure which is appropriate for the task it is designed to complete.

The solution can be implemented as a console solution or a GUI solution, however, the GUI (or 'frontend') will not be assessed in this assignment.

Stretch Exercises:

Implementing the game described above well, will enable you to achieve good marks in this assignment (up to 2:1). However, in order to challenge yourself and potentially achieve higher marks, attempting further tasks will be needed:

- Keep and display a 'history' of the dice throws in the game, eg:
    - `Turn 1 Player 1: 3 3 5 6 1`
    - `Turn 2 Player 1: 2 2 2 4 6`
    - Show statistics on these throws (eg: average of each die, total for each throw, average total for all throws, etc)
- Implement a 'throw all dice once' option where the points totals are doubled
- Implement a primitive AI which allows you to play against the computer.

The design and testing report should contain the following:

- A design for the application which should include:
    - A contents page
    - A written description of the application (1 page)
    - A class diagram for the application
    - A sequence diagram showing message passing involving at least one of the classes implemented. (1 page for the UML diagrams)
    - **Diagrams should either be hand drawn and scanned into your assignment (e.g. by pasting in a photo) or drawn from scratch using a UML or other graphics package. NO credit will be given for diagrams that are reverse engineered from the programme code**
- Any stretch exercises implemented
- A simple testing strategy for the application which should include: (1 page)
    - A 'black box' testing strategy and results
    - A 'white box' testing strategy and results
- The video URL
- Any references used in your report (In Harvard format)

A short (up to 1 minute) video of your application running should also be produced and uploaded to YouTube.

Your report will then comprise of the following pages:

- Contents page
- Application description page
- UML diagrams page
- Testing page
- References page

Useful Information

This assessment is an individual piece of work. Your work must be presented according to the Lincoln School of Computer Science guidelines for the presentation of assessed written work. Please make sure you have a clear understanding of the grading principles for this component as detailed in the accompanying Criterion Reference Grid.

If you are unsure about any aspect of this assessment component, please seek the advice of a member of the delivery team.

Submission Instructions

The deadline for submission of this work is included in the School Submission dates on Blackboard.

You should submit your work as a single ZIP to the *CMPXXXM Assessment Item 2 – Supporting Documentation Upload* section, and a report to the *CMPXXXM Assessment Item 2 – Upload* section. **Use of other compression formats such as RAR files will be penalised**.

a) The ZIP file which is uploaded to *Assessment 2 – Source Code* should contain the project files, accompanying grade input files, any output files, executable and source code files for your application. The project should be able to be opened in Visual Studio (or any other IDE that you have used – *Mono* for example).
b) The pdf report should be uploaded to *Assessment 2 – Report Upload*

Note: The URL of the video

i. The short video should be captured with free software such as Screencast-O-Matic (http://www.screencast-o-matic.com). Download and install the software. Using the 'free version', follow the instructions to capture your video. When the video is complete (no greater than 1 minute in length), select 'Upload to YouTube'. Upload your video as an 'unlisted' video (this allows us to see the video, but only when you tell us of its URL). Full, illustrated instructions for this process will also be available. The video should show your application running while you describe what you have done, how you have implemented it and how it works. You **will not** be assessed on the quality of the recording.

DO NOT include this briefing document with your submission.