School of Science
and Engineering
University of Dundee

AC22005 Assignment 1 – C# Grid Game

Deadline for Submission: Friday of Week 4 at 5pm (10th February 2023)

Hand in Method: MyDundee - You should upload a **single ZIP file** containing **all** the components of your source code (including compiled EXE) and your report. Failure to upload as a single file will reduce your overall mark. The **name** of your ZIP file should include your **team number**.

Date Feedback will be Received by: This will be received within the University's 3 week policy which for this assignment is Friday 3rd March 2023.

Penalty for Late Submission: One grade point per day late (meaning if a submission is one day late and marked as a C2 it will receive a C3 grade). A day is defined as each 24 hour period following the submission deadline including weekends and holidays. Assignments submitted more than 5 days after the agreed deadline will receive a zero mark (AB).

Percentage of Module: This assignment is worth 17% of AC22005.

Overview of Assignment

This assignment is designed to give you practice in using C# by developing a simple game, considering both the UI (for visual appeal) and the program structure (for the gameplay). This assignment is to be conducted in **teams of 3 students** including at least one Applied Computing student and one Computing Science student.

Specification

Building on the practical exercises, your task is to develop a "tile based" game i.e. a game **based around a grid of buttons** (or other UI objects). This will demonstrate a combination of a **"design view"-constructed** UI with components for entering your name and displaying a score etc. In addition to this, you **must** utilise a grid of "**code-generated" UI** components which share a **common event handler**. You should develop and design the game concept on paper prior to coding. The assignment must be implemented using **C# in MS Visual Studio 2022** and must run on the QM Labs computers **without** the need for any additional libraries or add-ons.

The choice of game is yours - it may emulate an existing program (e.g. *Minesweeper*, *Tetris*, *Candy Crush*), emulate a real physical game (e.g. *Connect 4*, *Boggle*, *chess*), or be entirely new, but it should:

- be based around a **grid** of tiles/buttons - the **user must be able to interact directly with the grid** as part of the gameplay (e.g. clicking to reveal a tile in *Minesweeper*); indirect interaction (e.g. via cursor keys or keyboard key presses) should ***not*** be the main method of interaction)
- have some "programmed intelligence" behind the buttons based on the way the **items in the grid interact** with each other (e.g. detecting checkmate in *chess* or a winning line in *Connect4*)
- should be neat, attractive and functional
- include a **minimum** of three different GUI object **types** and also include a **menu**

Your game may be for two players, one player against the computer, or even the computer against itself (or any combination).

Your game should feature the skills covered in class and where possible go beyond this additional features to enhance your "product". "Whack-a-mole" was demonstrated in the videos (sample code is available on MyDundee for this) and there are some examples of how to do a Noughts & Crosses (aka Tic-Tac-Toe) game online, which you might like to view for reference - ***for this reason, submissions of games similar to Noughts & Crosses or Whack-a-Mole will not be accepted.***

Simple Extensions

- demonstration of editing the properties of GUI objects (variety of colours, fonts etc.)
- "top" and "tail" the game by giving it a defined entry point and defined end point (e.g. "Game Over - your score is in the top 10")
- include a suitable scoring system
- include simple rules/help using a standard message box
- offer a "play again?" feature

Further Extensions

- offer different levels of play (easy / medium / hard)
- include a high score table (preferably which persists between sessions)
- include more sophisticated rules/help using a custom form
- include a timer (if appropriate to your game)
- include sound effects and/or music

- include appropriate graphic images
- include a new icon for your game

You are reminded that single lines of code taken from sources (such as manuals) may be used *without* reference, sections of 2 to 5 lines of code may be used *if the original source is cited*, but sections of code larger than this ***may not be taken*** from sources *even if cited* - it is expected that you write most of the submitted code yourself with no copying from other sources or collusion with other students (other than your team partners).

Submission & Assessment

This task must be carried out *in teams of 3* and you must submit a peer assessment detailing the contribution of all team members via **`peerreview.computing.dundee.ac.uk`**

This coursework is due for submission via MyDundee (under AC22005 **Assessment Area** on MyDundee) at 5pm on Friday 10th February (Friday of Week 4) and is worth 17% of your total grade for this module. Keep a copy of what you submit in case there are problems with your submission. As well as the source code for your C# game, you should include a **report** of 550-600 words (with word count) describing your approach to the problem, any difficulties that you encountered, and possible future enhancements to your game; you should also include **rules** for your game if these are not obvious or included in the game itself (*not part of your word count*) - see the **report writing guidelines** on MyDundee for more details on what is expected in the report. You should also include your **design draft** sketches (one page). Do ***not*** include a title page or a contents list in your report.

Your submission should be a single ZIP (not RAR) file containing:

- your **whole C# project folder** (including **all** source files and an **executable**)
- a one-page **paper design** (scan or photograph hand-drawn sketches if necessary)
- your team's **550-600 word report** (as a Word document, with word count given)

Your software should run on the QM Lab PCs, as these will be used when marking, and should **not** require the use of any special code libraries which are not normally available on these machines.

Constraints:

Your submission should be a **single ZIP file** containing your report, design and source code files. Failure to upload as a single file will reduce your overall mark. **All program elements and your report should include your team number and members' names in a header or comment.** One submission *per team* is sufficient but you must include the **identity of the team in your ZIP filename**.

Marking Scheme

A fully working but very simple game with small grid and which includes few (if any) of the extensions will achieve 34 marks (out of 83). A fully working game with some sophistication and which includes some of the further extensions will achieve 60 marks out of 83. Additional marks will be given for a particularly engaging user interface, sophisticated gameplay or inclusion of other (appropriate) extensions. Marks will be lost for poor or incomplete gameplay or coding, or for a poorly constructed or unappealing user interface.

A well-written report which meets all of the given guidelines, with an adequate design document, will achieve 14 marks (out of 17).

A simple solution with good report will thus achieve around 46% (D2) while a sophisticated solution with good report will thus achieve around 72% (A5).

A full sample marking scheme is available on MyDundee; you might like to try self-assessment of your program against this marking scheme.