**A Report on the Project Conducted for ‘Economic Analyses With Matrices (ECO2048)**

**Authors**

* Bradley Caton-Garrett (6309998)
* Jean Monique Delos Santos (6312397)
* Anjna Hirani (6253788)
* Jordan Poulter (6308473)
* Hareena Matoo (6308099)

**Project Summary** During our first meeting as a group, we discussed our strengths and weaknesses in terms of MATLAB, and also our interests as a group – this part of the conversation largely centred on television! The general consensus was that we should make the project fun in order to keep interest and passion in our work and so the idea of the television quiz-game became our focus.

**Work Development**

The initial priorities were to set-up a method of file-sharing and communication to enable collaboration over the Christmas period, with the aim of finishing the project by the 25th December (in hindsight, a very optimistic target!). We started by each choosing two programmes of our choice, and creating 15 questions for each, comprising of five easy, medium and hard difficulty questions. Jordan wrote an initial code in MATLAB which was similar to assignment 2 of the coursework. It would randomly generate a sentence with set of answers from a-d. Each question set was split by full stops. The user would need to input a letter a, b, c or d and depending on whether the answer was right or not the user could proceed to the next question with the accumulated winnings or the loop would terminate. However this original code wasn’t compatible with GUI and so we modified the bank of questions and code to support GUI.

At the Christmas mark we realised the work was progressing too slowly and so Jean suggested delegating jobs for each member of the group to complete – this included, but was not limited to: use of the lifelines; creating a ‘moneybar’; and of course, the main body of getting the right questions and answers to randomly appear. After this point the work progressed in a quicker fashion as the game rapidly came together and began to take a recognisable shape which was not too dissimilar to the finished project.

Jean took on the role of getting the main body of the quiz to work, this included getting random questions from a txt file to come up in a static text box and at the same time getting the associated choices in the individual pushbuttons. This process took some time to progress as initially we had put the easy, medium and hard questions in 3 txt files; Jean decided that the most efficient way to get rid of the issues of making sure that questions weren’t being asked twice in one game was to split the questions into 15 separate txt files corresponding a question. Once this occurred, it made it much simpler to get the code to work the way it does. There was also a lot of trial and error with creating the GUI and how we wanted the game to look, Jean made the initial layout of the game, Hareena recreated it for the rest of the questions and Brad decided on the colour scheme of the game that would look best.

From the start we knew tackling how to operate and disable the lifelines would be a tricky task. Therefore, two individuals (Anjna & Hareena) were assigned to completing this task as well as the ‘walk away’ button. We had a clear idea about how we wanted the lifelines to work, but had to do research into how to disable the button once it had been used and how the data would be stored and passed on through the different files throughout the game. The 50/50 option was fairly straightforward in the sense that we had to turn the visibility off for two buttons in each of the question files. We decided to personalise the buttons for each question so it worked in the favour of the player, rather than generalising and randomly choosing two options to be displayed. For ask the audience, we created 4 graphs from excel (each favouring either A,B,C or D) which displays the audiences results in a bar chart. Again, the results are suited for each question so that it helps the player progress in the game. Finally, call a friend is an audio sound clip which has a ring tone to make it sound realistic. A message is played and offers advice to the player suggesting which option should be picked. Once these images and audio files were completed we tried various methods which would disable the lifeline buttons preventing further use. Eventually, we realised Matlab has a built-in function 'setappdata' whereby you can store data by defining a particular function/button and you can turn the function/button on or off. Additionally, 'getappdata' allows you to retrieve the data stored under the defined function and if the button has been previously used then the button was disabled. If not, then it was enabled. Knowing where to place the first half of the code (in the opening function) was not clear at first but then became apparent with the hints that Matlab provides.

**How the Code Works**

The GUI was used to create the game, making use of buttons and axes. Each question has a separate script built for it, but all follow the same pattern; the difference arises in the .txt files that are called up to form the questions and answers. The questions are separated using the ‘strsplit’ command and then selected randomly, before the answers are individually split and put into the buttons representing options A, B, C and D.

Once the user selects their answer, the game will progress in one of two ways. In the case of the correct answer, the game will progress on to the next round; in the case of an incorrect answer, the gameedn.fig is brought up and the game finishes. Also, the player is allowed three lifelines if they need assistance in answering the questions. The 'moneybar' is put in place via an axes with the image changing for each question.

**How to Run the Code**

The files needed to run the code are all included in the submission, and must be added to the file path for the game to work. The player needs to open up playgame.m that can be found in the game folder of the submission file and run the file.

There are instructions on how to play the game that can be accessed from the start screen of the GUI. Examples of commenting can be found in the script for qu1.m and qu2.m(for lifelines).

CHEAT SHEET: 1)C 2)A 3)C 4)A 5)A 6)C 7)B 8)C 9)A 10)B 11)B 12)C 13)C 14)A 15)C

**Reviewing the Work Process**

As a group we largely worked well, getting the game finished to a standard we felt happy with. As no one has claimed anything other than an even 20% to each member, that is the allocation we are requesting.

**Code Citation**

<http://blogs.mathworks.com/pick/2007/10/16/matlab-basics-setting-a-background-image-for-a-gui/> - used to insert the 'moneybar' in each case