School of Science
Computer Science

Loughborough University

19COA202 Embedded Systems Programming Coursework *Version 2.1*

Dominik Freydenberger and Iain Phillips Semester 2, 2019-2020.

Introduction

This coursework forms 100% of the assessment for this module.

The Memory Game

In the memory game the Arduino shows a sequence of characters on the screen, blanks the screen and the user then repeats that sequence using the buttons. If the player is correct, then the game repeats becoming increasingly complicated. The system maintains a high score table of progress.

In slightly more details:

1. The machine displays a sequence of symbols
2. The human enters this sequence into the machine
3. If correct, then repeat

The number of potential symbols can be 2, 3 or 4. They will be entered using the keypad buttons. Each round of the game is a sequence of up to 10 symbols.

In the simplest form of the game implement a menu system to allow selection of how many symbols to choose and how many to choose from. Then run the game, print "success" or "failure" and return to the menu.

Specification

In this coursework, you are given an Arduino Uno and LCD keypad shield. Your assignment is to develop a software that plays a memory game.

The basic game

In the basic game the display shows a sequence S of N characters chosen from a set of M possible characters, where M is a subset of {LEFT, RIGHT, UP, DOWN}.

Each character is displayed in the same position on the display and is shown for an amount of time D (the delay).

The user remembers S and then enters it into the Arduino using the direction keys. In its simplest form the game displays 4 characters from a set of 2, say, LEFT and RIGHT.

As the game becomes more complicated the number of characters in N and M increases.

If the user enters the sequence correctly, then flash the backlight and move to the next stage.

If the sequence is entered incorrectly, then a suitable message is printed and the game is over.

Minimal feature set

The minimal feature set required to pass is a practice mode:

1. A game that implements sequences of up to 10 characters from up to 4 possibles.
2. A menu system to change the length of S (that is, N) and the size of M.
3. The delay D is set to 1 second.

To get a higher mark, you will need to implement the minimal game as described above and extensions (also see the marking scheme below).

Extension features

Possible extension features include:

- Handling sequences N that are longer than 10. Document the limitations of your solution for this.
- A time limit T for inputting the next character (e.g. 2 seconds). If a typing in takes longer than T, the user fails. The value of T should be set in your menu system.
- Displaying T as a countdown (for example, counting down in steps of 0.1 seconds).
- A *story* mode to build up though increasing N and M and reducing D. The order you do these is up to you, but you might for instance start with N=*4* and M=*2* and move up to N=*7* before moving to N=*4* and M=*3* etc.
- A menu to switch from *practice* mode to *story* mode.
- Selectable difficulty levels that set the the start values for M, N and D (and T, if implemented), in story mode.
- When displaying the characters to remember define your own graphical representation. This does not have to directly represent the directions (e.g. as arrows), but it should be clear to the user how characters and arrow keys correspond.
- Define additional characters so you can use a matrix of 2x2 display characters to display a bigger symbol.
- Suitable displays for winning and losing.
- A high score table written to the EEPROM and an addition initial menu option to display the table of high scores.
- A way for the winner of a high score to enter 3 initials to represent their alias that is displayed in the high score list.
- Anything reasonable that you can come up with. But keep in mind that they need to be concise enough for a 5-minute demo and that you will not be present at the demo. Feel free to use the serial monitor in a meaningful way. Remember to highlight your extra features in the report.

Assessment

Coursework type: individual coursework.

Submission date: Week 10, details will be announced later.

Demo: ~~5 minutes per student, in week 11; further details will be announced later.~~ The demo has been replaced with a test plan that is part of the written report.

Deliverables

Your submission needs to include the following:

- Source code of your implementation a single `.ino` file submitted to LEARN.
- A written report—submitted into the text field of the submission link.

Source code

Submit this as the actual `.ino` file, not copy-pasted into the report.

Written report

The report should contain the following:

- Simple instructions (a very short user manual).
- A list which parts of the minimal feature set have been implemented fully, partially (then include to which degree), and not at all.
- A list of optional or additional features (if present), with a short description.
- A test plan (see below).

A template for the report is available on the Learn page. It is probably wise if you write the report in some text editor and then copy-pate it into the Learn form.

Test plan

The test plan is a detailed instruction that shows us how to test all the features in your software. It must alternate between user actions (via Arduino buttons or the serial monitor) and intended behavior of the Arduino (on the display or over the serial monitor).

In the assessment, we will sit down and follow your test plan to see whether you have implemented all features (of course, we will also try out other things). If we need to see the output across the serial port, then meniton this clearly.

But make sure that the test plan covers every feature that you implemented, otherwise we might miss it.

Further information

- You may only use the following libraries (as identified by their header files):
  - `Wire.h`
  - `Adafruit_RGBLCDShield.h`
  - `utility/Adafruit_MCP23017.h`
  - `EEPROM.h`
  - `avr/eeprom.h`
  - `TimerOne.h`
- Your code has to consist of a single `.ino` file, which needs to be compilable using the Arduino IDE.
- Your code has to work with the Arduino Uno and LCD shield that we issued to you. In particular, if you develop your code on a different type of Arduino or with a different type of shield, you risk failing this module.
- We will use various types of software for code similarity detection to compare each submission to all others, to submissions from previous years, and to code from other sources. Please be reminded that this is a individual assessment, which means that group work is not permitted.

Grading scheme

Most marks are expected to fall into a scale between 40 (“scraped pass”) and 80 (“impressive”). These are explained in the grading descriptors below.

- A (80) - The project significantly exceeds the minimal specification in the amount of features and in the technical challenge behind these features. All features have been realized at an excellent level. The project is the culmination of a full design process with the implementation carried out at a high-quality level. Additional features are also present. The interface is highly intuitive.

- B (70) - The project significantly exceeds the minimal specification in the amount of features or in the technical challenge behind these features. The project implementation and testing have been well executed with some aspects at an excellent level. The interface is easy to understand and requires almost no explanation.
- C (60) - The minimal feature set and some non-trivial extra features have been realized at an adequate level. The project implementation and testing have been achieved at an adequate level. The interface is easy to understand after some explanation.
- D (50) - The minimal feature set and some minor extra features have been realized a mostly adequate level. Enough has been done, but nothing is very exciting and the overall achievement is mediocre. The interface can be understood with some effort.
- E (40) - The minimal feature set has been realized at a somewhat adequate level. The overall implementation and testing of the project is rudimentary. The interface is not intuitive and requires significant explanation.

To obtain higher marks, include extra features or implement features in a way that is particularly impressive.