INDIVIDUAL PROGRAMMING PROJECT

Version 1.0
Computing
COMP102

Matt Watkins

Assignment Brief

Experimentation, ingenuity, and creativity are at the heart of everything that creative developers do. To this end, building your own custom interface is the perfect place to exercise these characteristics. However, you will also gain invaluable exposure to working with computer hardware and embedded systems. In recent years, there has been considerable growth in the development of new fabrication technologies, such as 3D printers and laser cutters. In addition, electronics, from primitive transistors to complex computer chips, have all become much cheaper. Accessibility to these tools has, therefore, unveiled an unprecedented opportunity to invent and innovate in this space. Increasingly, app developers are augmenting mobile software with wearable devices. With the advent of VR, AR, and XR and the demand for augmenting reality through new sensing technology, developers are increasingly expected to consider new interfaces in the design of entertainment products.

In this assignment, you will leverage software engineering techniques to create a playful experience which interfaces with a custom control system. You will also illustrate design and technical characteristics of the experience you create through a poster demonstration.

This assignment is formed of several parts:

(A) **Propose** a playful experience that will:
  i. **fulfill** requirements implied by a contract;
  ii. **incorporate** physical computing;
  iii. **interface** with a custom control or sensing system;
  iv. and **leverage** object-orientated software architecture.

(B) **Implement**, as an **individual**, a prototype of the experience.

(C) **Create** an A3 portrait poster which serves to:
  i. **highlight** the contract, key requirements, and prominent features
  ii. **illustrate** the technical characteristics of your system, using UML notations and code excerpts
  iii. and **direct** assessors to the repository containing your source code.

(D) **Review** the draft experiences presented by peers, ensuring that you:
  i. **create** a video that outlines your work-in-progress demo and poster;
  ii. **attend** the timetabled peer-review session and share your video;
  iii. and **suggest** improvements to your peers.

(E) **Finish and Present**, as an **individual**, the final version of your playful experience alongside your poster to:
  i. **show** how you have translated requirements and technical notations into code;
  ii. **highlight** legal, social, ethical, professional, and sustainability considerations;
  iii. **illustrate** your individual technical competencies.
  iv. as well as **demonstrate** your academic integrity.

"As soon as we started programming, we found out to our surprise that it wasn't as easy to get programs right as we had thought. Debugging had to be discovered. I can remember the exact instant when I realized that a large part of my life from then on was going to be spent in finding mistakes in my own programs."

— Maurice Wilkes

"C++ is history repeated as tragedy. Java is history repeated as farce."

— Scott McKay

'Spin' - A game where you spin a chair to fly a helicoptor by Falmouth student Jae Knight

Contracts

To give you a wide brief and accommodate a broad range of applications you will **choose one** of two contracts. Please refer to the **separate contracts document** here:

https://learningspace.falmouth.ac.uk/mod/resource/view.php?id=228876

Assignment Setup

Fork the GitHub repository at:

https://github.falmouth.ac.uk/Games-Academy/COMP102-Project

Use the existing directory structure, the Unity Project should be placed inside the **Unity Project** folder and the Arduino project files should be placed inside **Arduino Project** folder. Ensure that you maintain the readme.md file.

Part A

Part A consists of a **single formative submission**. You should demonstrate your progress to a tutor in the timetabled session by **Week 10**

You will receive **immediate informal** feedback from your **tutor**.

Part B

Part B consists of a **single summative submissions**. You should commit your files into a **'documentation'** sub folder and push them to your **repo** it should contain the following:

(A) The Unity Project including all source code and assets
(B) The Arduino Project including .ino files
(C) Two images of the controller: one of the wiring and another with case/housing
(D) Video footage of the controller and game/experience being played. This should be uploaded to **Microsoft Streams** or other video streaming service. A **link to the video** should be provided (see next item).
(E) A readme.md with references to all sources and assets used in the project as well as image files and a link to the video.

This work is individual and will be assessed on a criterion-referenced basis. Please refer to the marking rubric at the end of this document for further detail.

Part C

To complete Part C, implement the final changes to your project. Prepare a **practical demonstration** of the project. Ensure that the source code and related assets are pushed to GitHub and a pull request is made prior to the scheduled viva session. Then, attend the scheduled viva session via Microsoft Teams.

This work is individual and will be assessed on a criterion-referenced basis. Please refer to the marking rubric at the end of this document for further detail.

You should also share your repo with your peers in the **Report Peer Review** session in **Week 13**.

You will receive **immediate informal** feedback from your **tutor**.

Part D

To complete part D please create an **A3 digital poster (PDF)** and present it at the **viva** session alongside your completed artefact.

The poster must contain the following

i. A brief description of the concept
ii. Detail the hardware of the control system
iii. Detail the design of the control system
iv. Detail the elements of the game/experience
v. UML Diagram of the software architecture

You will **receive feedback** at the session from **two assessors** and your peers.

Additional Guidance

Nobody learns in a vacuum: you are allowed, and indeed encouraged, to discuss your work with your peers. However you must be very careful to avoid falling into **academic misconduct**, in particular, **plagiarism**. If any part of your solution is **not your own individual work**, you must make this as clear as possible in your submission, for example in source code comments.

FAQ

- **What is the deadline for this assignment?**
  Each worksheet has its own formative deadline, specified on that worksheet and also communicated in class. Falmouth University policy states that summative deadlines must only be specified on the MyFalmouth system.
- **What should I do to seek help?**
  You can email your tutor for informal clarifications. For informal feedback, make a pull request on GitHub.
- **How will I receive feedback on my work?**
  You will be given verbal feedback on your work during the session in which it is marked. If you require more in-depth feedback or discussion, please book an appointment with your tutor.
- **Is this a mistake?**
  If you have discovered an issue with the brief itself, please provide feedback to your module leader.
  Please comment accordingly.
- **What coding standards are we using on this assignment?**
  We are using the Microsoft's coding styles for C#

  https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions

  and we are using the Arduino's coding styles for C++

  https://www.arduino.cc/en/Reference/StyleGuide

Additional Resources

- Wilkinson, K. and Petrich, M. (2014) The Art of Tinkering: Meet 150 Markers Working at the Intersection of Art, Science & Technology. Weldon Owen: London.
- Alicia Gibb. Building Open Source Hardware: DIY Manufacturing for Hackers and Makers. Addison Wesley, 2014.
- Jeremy Blum. Exploring Arduino: Tools and Techniques for Engineering Wizardry. John Wiley, 2013.

- Kelly, K. (2014) Cool Tools: A Catalogue of Possibilities. Cool Tools.
- https://www.sitepoint.com/heuristic-evaluation-guide/
  https://www.overleaf.com/project/61a0fc76d8ae11352a023a52
- https://www.usability.gov/how-to-and-tools/methods/heuristic-evaluation.html
- https://github.com/arduino/Arduino/blob/master/.gitignore
- https://gitignore.io/

Marking Rubric

All submissions and assessment criteria for this assignment are individual.

Criterion	Weight	Near Pass	Adequate	Competent	Very Good	Excellent	Outstanding
Basic Competency Threshold	30%	At least one part is missing or is inadequate.	Adequate ability to generate ideas, problem solving, concepts, technical competency and proposals in response to set briefs and/or self-initiated activity. The work demonstrates an adequate, ethically informed, real-world experience of industry/business environments and markets. Enough work is available to hold a meaningful discussion. Clear evidence of programming knowledge. Constraints followed No breaches of academic integrity.				
PROCESS: UML Diagrams	10%	No UML Diagrams.	UML Diagrams are incomplete. Non-standard UML notation has been used. Incorrect use of diagram type.	UML Diagrams are mostly complete. The UML notation are mostly correct.	UML Diagrams are appropriate. The choice of diagrams are appropriate. There are only minor issues with the notation.	UML Diagrams is directly linked to the software architecture. There are only minor errors in the diagrams.	UML Diagrams is of a professional standard. UML notation is correct. There are no errors in the diagrams.
PROCESS: Software Architecture & Sophistication of Code	20%	No attempt to describe the design of the software. No insight into the appropriate use of programming constructs is evident from the source code. No attempt to structure the program (e.g. one monolithic function).	Little insight into the appropriate use of programming constructs is evident from the source code. There is a description of the key classes but no insight into the data structures or design patterns used The program structure is poor.	Some insight into the appropriate use of programming constructs is evident from the source code. There is a description of the key classes. The data structures and design patterns have been described with no context. The program structure is adequate.	Much insight into the appropriate use of programming constructs is evident from the source code. There is a description of the key classes with reference to their functionality. The data structures and design patterns have been described with context to their application. The program structure is appropriate.	Considerable insight into the appropriate use of programming constructs is evident from the source code. There is a description of the key classes with reference to their functionality. The program structure is effective. There is high cohesion and low coupling. The data structures and design patterns have been described with context to their application. There is some justification for the selection of data structures and design patterns.	Significant insight into the appropriate use of programming constructs is evident from the source code. The program structure is very effective. There is high cohesion and low coupling. There is significant evidence of software design There is synergy between data structure and design pattern selection.

Criterion	Weight	Near Pass	Adequate	Competent	Very Good	Excellent	Outstanding
PROCESS: Maintainability of Code	10%	There are no comments in the source code, or comments are misleading. Most variable names are unclear or inappropriate. Code formatting hinders readability.	The source code is only sporadically commented, or comments are unclear. Some identifier names are unclear or inappropriate. Code formatting is inconsistent or does not aid readability.	The source code is somewhat well commented. Some identifier names are descriptive and appropriate. An attempt has been made to adhere to Microsoft's formatting style. There is little obvious duplication of code or of literal values.	The source code is reasonably well commented. Most identifier names are descriptive and appropriate. Most code adheres to the Microsoft's formatting style. There is almost no obvious duplication of code or of literal values.	The source code is reasonably well commented in the Microsoft's style Almost all identifier names are descriptive and appropriate. Almost all code adheres to the Microsoft's formatting style. There is no obvious duplication of code or of literal values. Some literal values can be easily changed in the Unity Editor.	The source code is very well commented, with C# doc-strings. All identifier names are descriptive and appropriate. All source code adheres to the Microsoft's formatting style. There is no obvious duplication of code or of literal values. Most literal values are, where appropriate, easily changed in the Unity Editor.
PROCESS: Hardware Design & Functionality of Physical Prototype	20%	No physical prototype, or the prototype is not functional. No description of the hardware design.	Little insight into the hardware design and the physical prototype has a little functionality. There is little evidence of design, no regard to the selection of components and form factor. There are critical technical and/or constructional flaws.	The physical prototype has some functionality. There are major technical and/or constructional flaws. Some insight into the hardware design. There is some evidence of deliberate practice from the student. The selection of components and form factor is justified.	The physical prototype has much functionality. If any, only minor technical and/or constructional flaws. Much insight into the hardware design. There is some evidence of deliberate practice from the student. The selection of components and hardware complement each other	The physical prototype has considerable functionality. If any, only minor technical and/or constructional flaws. Considerable insight into the hardware design. There is considerable evidence of deliberate practice from the student. The selection of components and hardware complement is justified.	The physical prototype has significant functionality. If any, only superficial technical and/or constructional flaws. Significant insight into the hardware design. There is significant evidence of deliberate practice from the student. The selection of hardware is complemented and justified in terms of the interactive experience.
PROCESS: Electronics Sophistication	5%	Solution lacks even basic use of electronic components.	Solution has some sophistication in terms of electronics. Little insight insight into electronic circuits.	Solution has some sophistication in terms of electronics. Some insight into electronic circuits.	Solution leverages electronic components with much effectiveness. Much insight into electronic circuits.	Solution leverages electronic components with much effectiveness. Considerable insight into electronic circuits.	Solution leverages electronic components with much effectiveness. Significant insight into electronic circuits.
PROCESS: Physical Form Factor Sophistication	5%	No physical prototype, or it is limited to a breadboard without housing.	Physical form factor has a little sophistication. Little insight into human-computer interaction.	Physical form factor has some sophistication. Some insight into human-computer interaction.	Physical form factor has much sophistication and sturdiness. Much insight into human-computer interaction.	Physical form factor has much sophistication and sturdiness. Considerable insight into human-computer interaction.	Physical form factor has considerable sophistication and sturdiness. Significant insight into human-computer interaction. The control system has both practical and aesthetic value.