Module name: Software Development 3
CMP020N201
Lecturer: Dr Lisa Haskel

Coursework specification

PREPARATION
Before you begin this coursework, it is important to plan your activities and to manage the delivery of your milestones in the context of the submission deadline. Therefore, you are strongly advised to **READ THIS COURSEWORK DOCUMENT CAREFULLY!**

ACADEMIC MISCONDUCT
Your submission for this coursework will be scrutinised for plagiarism, collusion, and other forms of academic misconduct. Please ensure that the work that you submit is your own, and that you have cited and referenced appropriately, to avoid having to attend an academic misconduct hearing. You can find more information about academic integrity and how to mitigate plagiarism here: https://roehampton.libwizard.com/f/academic_integrity

Overview

You will build a small application using c#. You will then contrast this by implementing some of the same features in F#.

The coursework should be delivered in 4 'sprints'. In each sprint, you will add new functionality and improve your code using the knowledge you will gain as the module progresses.

Planning and documentation is as important as working code. We are looking for a professional approach in which you consider readability, maintainability and conformance to standards as well as correct operation.

Please note that this is an individual piece of work. You will be asked to present your work in person to a lecturer and answer questions about your solution. There will be two formative assessments (not marked – to help you stay on track) and two summative assessments (marked in code reviews in class).

Task

Please choose one of these projects to develop in your coursework. You should choose one project and stick to it throughout the module, fulfilling each of the sprint requirements in the context of your chosen project. Think carefully about which topic interests you most as you will work on it throughout the term.

Your overall aim is to produce well architected OOP code that shows an understanding of the problem domain, created using standard OOP technique, and properly documented both within the code and in accompanying design documents. For each sprint there will be baseline requirements that must be met to achieve a pass and advanced features that you can implement if you are aiming for higher marks.

Projects

SpareB&B

Users can book different kinds of accommodation. Houses, flats, hotel rooms and sofas are available, plus other accommodation types you can think of. Your accommodations may have different facilities such as kitchens or washing machines.

ZipAbout
Users can locate shared lightweight electric transport vehicles – bikes, scooters, skateboards, segway's and other vehicles, perhaps yet to be invented. For some vehicles, users will need equipment such as helmets, kneepads or gloves to use them.

LibraryOfStuff
Users can borrow all kinds of items - tools, sewing machines, phone chargers, bicycle pumps – little things we often need but can't lay our hands on. Items can be categorised by the part of the home in which they are usually used: kitchen, garden, workshop, or 'out and about'. They may also be categorised by an interest or hobby, or some other method of categorisation that you can think of.

Sprint 1 – Basic Classes -
FORMATIVE assessment in lab sessions:
w/b 16 Oct and w/b 23 Oct (weeks 4 and 5)

- Consider the fundamental objects that can be booked in your application, and their methods and properties.
- Use the OOP principles of Abstraction, Encapsulation, Inheritance and Polymorphism to design your classes
- Create a class diagram showing the relationships between your classes.
- Write well documented c# code to implement your classes within a console application. Minimum requirements should be:
    - a constructor
    - some properties
    - a print() method which gives some user-friendly output.
- Provide a Program.cs file you can run in the console in which:
    - At least 3 objects of different types are created in your code
    - A user-friendly listing of the objects can be printed to the console

Sprint 2 - Booking functionality
Deadline: 21 November 2023
Code reviews: w/b 27 November 2023
50% of marks

Implement a booking system for your chosen project. **You do not need to implement booking for specific dates or durations**, but it should be possible for a user to book and release items. Once an item is booked, it should not be available for booking by any other user.

Basic requirements

- Provide a class diagram showing the relevant relationships.
- Your design and code should incorporate the sprint 1 requirements above plus booking functionality
- Provide a Program.cs file in which **at minimum**:
    - At least 3 objects of different types are created directly in your code.
    - At least 3 users are created from user input or directly in your code.
    - In code, show that a booking can be made of an item by a user.
    - Show in your code that if another user tries to book the same item, they are politely informed that the item is unavailable
    - User-friendly output can be printed showing correct information about items and bookings at any particular time.

Advanced requirements

- Include at least one design pattern, accompanied by a short explanation of how it has been implemented, and/or
- Devise a feature of your own choice that demonstrates your deeper understanding of design patterns. Perhaps by rewarding your application users in some way with loyalty points or a random prize draw or perhaps by adding features to your application such as managing cleaning or maintenance or providing output in different formats (these are just examples).

Sprint 3 - GUI development
Deadline 4 January 2024
Code reviews (with sprint 4): w/b 8 January 2024
30% of marks

Create a GUI for your application using WPF.

Basic requirements

- Implement basic button and list functionalities taught in class with some original design and formatting.
- Use the classes you have created in sprint 2 in your WPF 'code behind' to create some basic functionality where you list bookable items.

Additional requirements

- Use the classes you have created in sprint 2 in your WPF 'code behind' to implement basic booking functionality and additional features of your choice.

Formative assessment available in weeks 11 and 12

Sprint 4
F#: Deadline 4 January 2024
Code reviews (with sprint 3): w/b 8 January 2024

20% of marks

- Implement the bookable items from your project that you designed in sprint 1 as types in F#. I will expect one base class and maximum 3 derived classes. You do not need to implement anything else.
- In code, create at least one of each of your derived class
- Print to the console, a listing of the items in a user-friendly format.
- Provide a solution to the additional f# exercises that will be provided

Marking scheme/ criteria

Sprint 2

Basic requirements (3rd – 2.2)

- Good design
- Correct code
- Class diagram
- Coding standards
- Inheritance
- Encapsulation

Advanced features (2.1 and above)

- Association/aggregation
- Design patterns
- Additional features

Sprint 3

Basic requirements (3rd – 2.2)

- Button
- List items
- Coding standards
- GUI design
- Use of classes from sprint 1 in 'code behind'
- List bookable items
- Coding standards
- GUI design

Advanced requirements (2.1 and above)

- Use of classes from sprint 1 in 'code behind'
- List bookable items
- Choose a bookable item
- Coding standards
- GUI design
- Advanced features

Sprint 4

Basic requirements

- Correct code
- Coding standards

.