ECM1400 Programming
Continuous Assessment 3

Date set: 26th October, 2020
Hand-in date: **11:59am 4th December, 2020**

This continuous assessment (CA) comprises 50% of the module assessment.

Note that electronic submission is required and instructions are provided at the end of the specification.

This CA is designed to test the programming concepts that we will cover in the entire module. This assignment will require the application of fundamental programming constructs, control flow, data structures and design patterns to develop and deliver a software product to solve a real-world problem. Note that some of the content is only taught within 2 weeks of the deadline, however, this should not stop you making a start and implementing a lot of the functionality well in advance of the deadline.

Specification

Since the outbreak of COVID-19 the day-to-day routine for many people has been disrupted and the instability of our environment means we have to be adaptable to current events. Keeping up-to-date with the rapidly changing local and national infection rates and regularly updated government guidelines has become a daily challenge for many as well as keeping track of the weather and plans that are often changed at short notice.

Smart systems are automated systems that adapt to input data streams. Alarm clocks are an everyday item that we use to schedule our lives. The scope for a smart alarm clock that can access information about the COVID infection rate and provide scheduled updates about the the weather and the news and engage with us through a lightweight interface has the potential to be very useful. In this assignment you will implement a smart covid-aware alarm clock.

The implementation of an alarm clock is not like a typical python script you may have developed before. An alarm clock will need to run continuously and respond to events rather than responding inline with user inputs. These events can be scheduled by the user or triggered by extenal events and require an event driven software architecture that uses scheduled tasks. Your smart alarm will have a series of features defined below and use the modules specified to achieve this smart event driven behaviour.

The specification has been broken down into three sections; the first two are each worth 10 marks and the final section is worth 20 marks.

- **Alarms and notifications**

  The program should enable the user to schedule and cancel 'alarms'. The alarm should trigger a scheduled function when the relevant time is reached. A key part of an alarm is notifying the user to an event. The smart alarm will use two types of notifications, announcements and notifications. For announcement alarms you will use text-to-speech voice announcements. For less immidiate notifications, silent notifications will be tracked through a text based 'notification' list.

  Alarm functionality should be defined using the `sched` and `time` modules. Announcement functionality should be defined using the `pyttsx3` module. Alarms and notifications should be stored using appropriate data structures and modified using functions.

  Note: announcement and notification functions should be implemented as functions independent of the alarm functionality which tracks time. You may want to implement the announcement and notification functionality before the alarm. Event-driven architecture required for alarms will be introduced and demonstrated in a workshop later in the module.

  [10 marks]

- **Smart-ness**

  To make your alarm clock adaptable it will need some smarts. The intelligence in your system will be achieved by pulling information from web services through restful API calls and merging this information with information on your local machine that is stored as part of the scheduled alarms.

  Access current weather and news data using the `requests` module and the Openweathermap API (https://openweathermap.org) and News API(https://newsapi.org/). Access current COVID-19 data using the `uk-covid-19` module provided by Public Health England (https://publichealthengland.github.io/coronavirus-dashboard-api-python-sdk/). All these information sources should also be available as part of alarm updates. Create a default behaviour for alarms that gives a daily briefing including top news stories, current local weather and local COVID infection rates.

  Note: the data handling functionality should be independent from the data access functionality. You should get started by using sample data that is downloaded and stored locally while developing the functions before linking it up to the third party web services.

  [10 marks]

- **Project delivery**

  This specification is for a software product for users. This means there are more detailed consideration about how the project is delivered. The project needs an interface, it needs instructions for deployment and customisation, it will run continuously so will need to be moniored and it depends on third party services so will need to be regularly tested.

  – User interface: All smart systems are designed with automation in mind so there is an emphasis on minimal user input, however, the user needs to be able to set and cancel alarms. A template will be provided for the html interface although you will need to map the interface with functionality. The interface should use the flask module to enable the user to set alarms and see basic information, such as notifications about passive events. **[5 marks]**

  – Configuration file: Sensitive information, such as api keys and other user credentials, and filepaths for application resources should be stored in a configuration file rather than in the source code. Passwords should also not be needed and should never be included in source code. Create a json file called config.json that contains the API keys and credentials for any web services and filepath information for any local resources, such as a log file. **[5 marks]**

  – Logging: A central functionality in any event driven architecture is tracking and handling events. This means the system should be executing tasks remotely, while you arent watching. Logging is therefore an important feature that will make the program accountable and make it possible to trace behaviour and recover state. You should log all events that happen in your smart alarm and categorise different types of event that may be treated in different ways. **[5 marks]**

  – Testing: As your software will depend on third party services and is designed to run continuously there is a good chance errors will occur while the program is deployed. As part of your code you should include unit testing for each of your functions and include a deployment test cycle as well as scheduling tests to regularly check the functionality of your program. **[5 marks]**

  – Deployment and Documentation: The service you develop may be useful for more than just one user and it may inspire others to extend the code. The code may be hosted publicly and pushed to as a public module in a repository that others to import. In both of these scenarios the documentation is foremost. A user must know how to use the system and a developer must know how the code is structured, what it does and how to extend it. **[5 marks]**

You should carefully follow the functionality described. Note the requirements intentionally do not include some details and you should make design decisions carefully.

Submitting your work

The CA requires electronic submission to the BART online submission platform.

Electronic You should submit your Python code via the electronic submission system at `https://bart.exeter.ac.uk/` under CEMPS and Harrison. Use the category containing ECM1400 and 2020-21 Continuous Assessment 3. Upload a compressed version of your files as a single file using the `zip` compression format.