ECM1400 Programming
Continuous Assessment 2

Date set: 16th November, 2019
Hand-in date: **11:59am Friday 15th November, 2019**

This continuous assessment (CA) comprises 25% of the overall module assessment.

Note that both paper (BART) and electronic submissions are required and instructions are provided at the end of the specification. The specification will be discussed in the lecture on Friday 18th October 2019. Students have until 25th October to raise any questions regarding the specification either in lectures or by email. Up to this date the module leader may make adaptations to clarify the assessment. After this date the specification will not change.

This CA is designed to test the programming concepts that we will cover in the first eight weeks of term, particularly variables, built-in types, lists, loops, operators, conditionals, modules, data structures, the event driven design pattern and exceptions.

Specification

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 knows about our schedule for the day, knows what the weather will be and can tell the time has the potential to be very useful. In this assignment you will implement a smart 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.

- **Alarms**

  The program should enable the user to schedule and cancel alarms. The alarm should notify the user when the relevant time is reached. Alarm functionality should be defined using the sched and time modules.

  [10 marks]

- **Notifications**

  A key part of an alarm is notifying the user to an event. The smart alarm will use two types of notifications. For alarm notification announcements we will use text-to-speech voice

announcements using the pyttsx3 module. For less immidiate notifications, silent notifications will be tracked through a personal notification queue.

[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 the internet and merging it with local information in the scheduled alarms. Pull in the weather and news data using the requests module and the Openweathermap API (`https://openweathermap.org`) and News API(`https://newsapi.org/`).

  [10 marks]

- **User interface**

  All smart systems are designed with automation in mind so the interface should should be minimal, however, the user needs to be able to set and cancel alarms. Implement a basic web interface using the flask module that runs on localhost. The interface should enable the user to set alarms and see basic information, such as notifications about 'passive' events.

  [10 marks]

- **Bonus features - max 10 marks**

  There are four possible bonus features for this assignment worth 5 marks each. However, the maximum mark for this section is 10 marks so not all bonus features are necessary to score full 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 aren't 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]

  – **Private web service notifications:** Many users use a range of private internet services, such as office365, youtube and netflix. Most of these services provide an API

provision so that applications can receive notifications to integrate with their services. Add third party services to your notification system by generating API keys and pulling information from these services.

[5 marks]

You should carefully follow the functionality described and use the modules recommended in each section. If you design a program with a different structure or depend on alternative modules you will not attract as many marks.

Submitting your work

The CA requires both paper and electronic submissions.

Paper You should submit a paper copy of your formatted source code and a single screenshot of your user interface to the Harrison Student Services Office in the foyer of the Harrison Building by the deadline of **11:59am Friday 15th November, 2019**.

The paper copy of the source code can be generated by exporting from PyCharm or printing from notepad++. These options will retain the formatting, line numbers and syntax highlighting.

Paper submissions should have the BART cover sheet securely attached to the front and should be anonymous (contain your student rather than your name).

Electronic You should submit your source code and any supporting files via the electronic submission system at `http://empslocal.ex.ac.uk/submit/`. Use the category containing ECM1400 and CA2. Upload a compressed version of your files as a single file, use the `zip` compression format.

After you click proceed you will be sent an email by the submit system asking you to confirm your submission by following a link. Your submission is not confirmed until you do this. It is best to do it straightaway, but there is a few hours leeway after the deadline has passed. It is also possible to unsubmit and resubmit electronic coursework — follow the instructions on the submission website.

Both paper and online submissions must be completed by the deadline else your submissions will be considered late and your mark will be capped at 40%.