Gen
AI
AI tools
can be used in an
assistive role

COMP1850 Programming

School of Computer Science, University of Leeds

Worksheet 1.3

This worksheet contains a combination of formative activities and activities that contribute towards your portfolio.

- Portfolio activities are indicated in the title of the task.
- Activities marked by (*) are advanced activities and may take more time to complete.

Expectations:

1. Timeliness: You should complete all the tasks in the order provided and submit your portfolio task to Gradescope before the deadline.
2. Presentation: You should present all your work clearly and concisely following any additional guidance provided by the module staff in the module handbook.
3. Integrity: You are responsible that the evidence you submit as part of your portfolio evidence is **entirely your own work**. You can find out more about Academic integrity on the Skills@library website. All work you submit for assessment is subject to the academic integrity policy.

Feedback: Feedback on formative activities will be provided via Lab classes and tutorials. Feedback on evidence submitted as part of the portfolio will be available on Gradescope.
Support opportunities: Support with the activity sheet is available in the Lab classes and tutorials. Individual support is available via office hours.
Expected complete date: Friday 17th October

All portfolio tasks are rated AMBER by the university: AI tools can be used in an assistive role. You are permitted to use AI tools for specific defined processes within the assessment.

Within this assessment you **may** use Generative AI to:

- help you start to understand more complex ideas by providing accessible summaries
- test your knowledge against a piece of content
- help identify and correct spelling mistakes and grammatical errors in your work
- provide feedback and advice on your overall coding style

You must **not** use Gen AI to:

- produce written content (code, comments or any other content) which you then submit as your work, regardless of whether you make changes to it.
- rewrite or make changes to any of your work.

Learning Outcomes:

- Develop programs that use the fundamental programming constructs: assignment and expressions, basic I/O, conditional and iterative statements
- Develop programs that effectively use the different structured data types provided in the language like lists, dictionaries, and sets

Practice task

The Recaman Sequence is a series of integers a_0, a_1, a_2, ... generated by the following formula:

- $a_0 = 0$

We generate further numbers at k = 1, 2, 3, ... using :

- $a_k = a_{k-1} – k$ if $a_{k-1} – k > 0$, and this number is not already in the sequence
- $a_k = a_{k-1} + k$ otherwise

For example: k=1, $a_{k-1}–k = a_0-1 = -1 < 0$, hence $a_1 = a_{k-1}+k = a_0+1 = 1$

The first 10 numbers in the sequence are: 0, 1, 3, 6, 2, 7, 13, 20, 12, 21

You should:

- choose an appropriate iteration approach for generating the sequence ;
- construct appropriate logical expressions for generating the sequence ;
- choose an appropriate data structure for storing the sequence.

The template code requires only:

- Input of the sequence length
- Output of the generated sequence as integer values

It is recommended that you use further output during development of your code, to test the logic of your application. You can use the data provided above to test sequence lengths up to 10.

(Wikipedia provides some interesting facts about this sequence including an audio representation.)

Portfolio Task

A road crossing is controlled by a traffic light. A red light means traffic must stop, a green light means traffic may go. The traffic light shows red and amber when it is about to change to green, and amber when it is about to change to red.

The traffic light passes through a repeating set of states, with a fixed duration, as defined by the dictionary below:

```
states = {  “red”: 4,  “red_amber”: 3,  “green”: 5 ,  “amber”: 3 }
```

The “traffic_light.py” code provides a skeleton for the simulation of the traffic light system for a given number of time units.

There is one input to the code – the number of steps required (>0).

You should complete the code so that at each step the state of the system is printed to screen. For your submission that should be the only output – as shown in the first print statement.

For example the output after the first 7 steps should be:

```
Time 000 State red

Time 001 State red

Time 002 State red

Time 003 State red

Time 004 State red_amber

Time 005 State red_amber

Time 006 State red_amber

Time 007 State green
```

You will need to use a combination of iteration and logical control of the state.

Submitting to Gradescope will test the state transitions through the defined states.

Sample Exam Question

Consider the Python code ‘cmdloop.py’ in the exam folder of the Github resources.

Draw a flowchart representing the execution of this program for any input data.