Module Code
COM00015C

BEng, BSc, MEng and MMath Degree Examinations 2025–26

Department Computer Science
Title Software 1: Formative Assessment 1
Issued 06:00pm on Monday 20^{th} October 2025
Submission due 09:00am on Monday 27^{th} October 2025
Feedback & marks due 06:00pm Tuesday 28^{th} October 2025
Time Allowed 4 Hours. Which includes time for uploading and testing.

Time Recommended THREE hours

Word Limit Not Applicable

Allocation of Marks:
All five questions are worth 20% each. All questions are independent and can be answered in any order.

Instructions:
Candidates should answer **all** questions using Python 3.10 to 3.12. Failing to do so will result in a mark of 0%. All questions are independent and can be answered in any order. Download the paper and the required source files from the VLE, in the "Assessment> Formative Assessment 1" section. Once downloaded, unzip the file.

Submit your answers to the GradeScope submission point named **SOF1 2025-26 formative assessment 1**. You can find the submission point on the "Assessment>Formative Assessment 1" page on the VLE.

A Note on Academic Integrity

We are treating this online examination as a time-limited open assessment, and you are therefore permitted to refer to written and online materials to aid you in your answers.

However, you must ensure that the work you submit is entirely your own, and for the whole time the assessment is live you must not:

- communicate with departmental staff on the topic of the assessment
- communicate with other students on the topic of this assessment
- seek assistance with the assignment from the academic and/or disability support services, such as the Writing and Language Skills Centre, Maths Skills Centre and/or Disability Services. (The only exception to this will be for those students who have been recommended an exam support worker in a Student Support Plan. If this applies to you, you are advised to contact Disability Services as soon as possible to discuss the necessary arrangements.)
- seek advice or contribution from any third party, including proofreaders, online fora, friends, or family members.

We expect, and trust, that all our students will seek to maintain the integrity of the assessment, and of their award, through ensuring that these instructions are strictly followed. Failure to adhere to these requirements will be considered a breach of the Academic Misconduct regulations, where the offences of plagiarism, breach/cheating, collusion and commissioning are relevant: see AM1.2.1 *(Note this supercedes Section 7.3 of the Guide to Assessment)*.

1 (20 marks) Functions and Strings

All code for this question must be written in the file `question_1.py` provided.

Write a function `seconds_to_time(time)` that takes an `int` as a parameter representing the time in seconds, and returns a string (`str` type). The format of the returned String is as follows:

(a) [8 marks] `hh:mm:ss` where two digits are always used for hours (h), minutes (m), and seconds (s) when the time is greater than an hour. For example 3625 seconds is represented by `01:00:25`.

(b) [8 marks] `mm:ss` where two digits are always used for minutes (m), and seconds (s) when the time is less than an hour. For examples: 85 seconds is represented by `01:25`, and 16 seconds is represented by `00:16`.

(c) [4 marks] In addition the method returns `None` if the time is negative or greater than `99:59:59`.

2 (20 marks) Functions and Strings

All code for this question must be written in the file `question_2.py` provided.

Write a function `time_to_seconds(time)` which takes a `str` (string) as a parameter and returns a time in second as an `int`. The function should accept the following string format for the parameter `time` :

(a) [8 marks] `hh:mm:ss` where two digits are always used for hours (h), minutes (m), and seconds (s). For example `01:00:25` represents an hour and twenty five seconds.

(b) [8 marks] `mm:ss` where two digits are always used for minutes (m), and seconds (s). For example `01:25` represents a minute and twenty five seconds. This format is **always** used when the time is less than an hour.

(c) [4 marks] For any other format the function returns `None`.

3 (20 marks) Python's Dictionaries

All code for this question must be written in the file `question_3.py`.

The standard atomic weight of an element is the average mass of the atoms of an element. A sample of atoms and their standard atomic weight is shown in Table 1. The molar masses of molecules are calculated from the standard atomic weights of each element. One can compute the molar mass of a compound by adding the standard atomic weights of its constituent atoms. For example the molar mass of water (H_2O) is given by:

$$\begin{aligned} m_{H_2O} &= 2 \times m_H + m_O \\ &= 2 \times 1.00797 + 15.9994 = 18.01534 \end{aligned}$$

Symbol	Name	Atomic Weight (g/mol)
H	Hydrogen	1.00797
He	Helium	4.00260
C	Carbon	12.011
O	Oxygen	15.9994

Table 1: List of elements and their atomic weights.

For this question, the table of atomic weight is stored in a global variable `ATOMS`. The variable `ATOMS` is a dictionary where the keys are the atoms' symbol, and the values are another dictionary. This second dictionary has two keys, the `'name'` mapped to the full name of the atom and the `'weight'` mapped to the atomic weight of that atom. An extract from the dictionary is given below.

```
ATOMS = {'H':{'name':'Hydrogen', 'weight':1.00797},
         'He':{'name':'Helium', 'weight':4.00260},
         ...}
```

A molecule is represented by a list of tuples, where the first element of a tuple is the string corresponding to an atom symbol, and the second element is the number of occurrences of that atom at the given position in the structure of the molecule.

For example, the molecule H_2O is given by `[('H',2),('O',1)]`, whereas the Acetic acid molecule CH_3COOH is represented by
`[('C',1), ('H',3), ('C',1), ('O',1), ('O',1), ('H',1)]`.

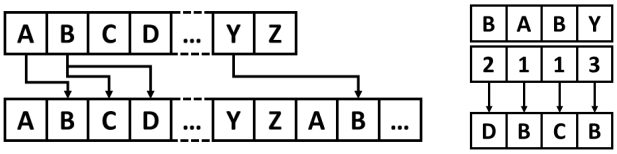
Implement a function `molar_mass(molecule)` that takes a list of tuples representing a molecule and returns its molar mass.

The function returns `None` if the molecule contains an unknown atom, that is an atom symbol that is not in the dictionary `ATOMS`.

4 (20 marks) Built-in Data Structures.

All code for this question must be written in the file `question_4.py`.

The Caesar cipher is a very simple encryption method and it is easily breakable. In this question we use an improved encryption method. Given a plain text message T of n characters using an alphabet A of m letters, and a list S of n positive integers from the set $\{0, 2, .., k-1\}$, T is encrypted by shifting the k^{th} character $T[k]$ by an amount $S[k]$. An example is given in Figure 1 where $T = [B, A, B, Y]$ and $S = [2, 1, 1, 3]$. The first character 'B' is shifted by 2 giving the character 'D', then the second character 'A' is shifted by 1 giving 'B' and so on. The encrypted message is "DBCB".



(a) (b)

Figure 1: Example of the encryption method proposed in this question.(a) the top row represents the character to be encrypted, the bottom row is the encrypted character given a certain shift. Note how the alphabet is repeated in the bottom row. (b) shows the encryption of the message "BABY" using a shift sequence 2113 resulting in the encrypted message "DBCB".

Implement the function `encrypt(message, shifts, alphabet)` that takes a plain text `message` as a string, the `shifts` sequence as a list of integers, and the `alphabet` as a string. The alphabet can be any sequence of symbols, not necessarily the English alphabet. The function returns a string containing the encrypted message.

The function returns `None` if:

- the message contains characters that are not in the alphabet,
- the size of the `shifts` is not the same as the size of `message`,
- `shifts` contains negatives values or values greater or equal to the size of the alphabet.

5 (20 marks) Basic Programming Structure

All code for this question must be written in the file `question_5.py`.

The aim of this exercise is to transform a string representing a molecule into a list of tuples, where the first element of a tuple is the string corresponding to an atom, and the second element is the number of occurrences of that atom at the given position in the structure of the molecule. For example, the molecule CO_2 given by the string `'CO2'` is represented by the list of tuples `[('C',1),('O',2)]`.

Glucose $C_6H_{12}O_6$ is represented by `'C6H12O6'` and `[('C',6),('H',12),('O'.6)]`. The Acetic acid molecule CH_3COOH is given by the string `'CH3COOH'` and the list of tuples `[('C',1), ('H',3), ('C',1), ('O',1), ('O',1), ('H',1)]`.

Some atoms are described by more than one character, like calcium Ca. The first character is uppercase and the second character is lowercase. For example, the calcium carbonate molecule $CaCO_3$ is represented by the string `'CaCO3'`. Its list representation is `[('Ca',1), ('C',1), ('O',3)]`.

Implement a function `molecule_to_list(molecule)` that takes a string representation of a molecule as described above and returns its list of tuples representation.

The function returns `None` if:

- `molecule` does not start with a uppercase letter from the English alphabet,
- `molecule` contains characters that are not alphanumeric.
- `molecule` has an invalid format, for example the symbol of an atom is lowercase like ca instead of Ca or h instead of H.

End of examination paper