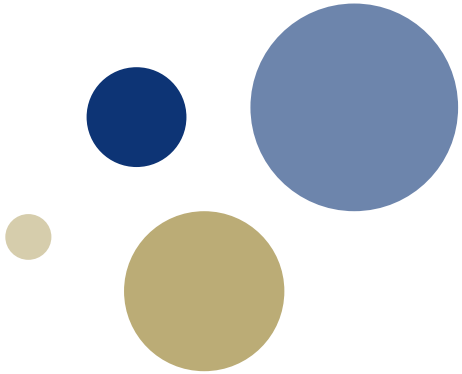

Workshop

Week 41

Warm-up

In this week's class exercise you were asked to calculate the average score for students and the average score on tests when the test scores were stored in a nested list.

Consider the following nested list, where each sublist contains the test scores for a single student on each of the three tests:

```
table = [[85, 91, 89],  # test scores for student 1
         [78, 81, 86],  # test scores for student 2
         [62, 75, 77],  # test scores for student 3
         [70, 65, 72]]  # test scores for student 4
```

1. Write a program that calculates the average score *for each student* and stores the averages in a list called `student_scores`.

2. Write a program that calculates the average *on each test* and that stores the averages in a list called `test_scores`.

Exercise 1a

Convert the program written in the warm-up exercise into functions.

Write a program consisting of two functions:

- avgRow – takes a table in the form of a nested list and returns the average for each row (sublist) in the table.
- avgCol – takes a table in the form of a nested list and returns the average for each column in the table.

The program should use the functions to print the average grades for each student and on each test.

Remember proper function documentation.

Exercise 1b

Write a program that creates a table with the tests scores of students.

The table should be in the form of a nested list where each sublist contains the tests scores for a spesific student.

The program should:

- prompt the user for the number of students and the number of tests that each student has taken.
- for each student:
  - prompt the user for the scores on each of the tests.
- store the test scores in a nested list called *table*.

Exercise 1c

Write a program that creates a table with student scores and that calculates and prints the average scores for each student and each test.

Implement the program using a main function.

The main function should:

- prompt the user for the number of students and tests.
- create the table.
- calculate the average student (row) and test (column) scores.
- display the averages.

Exercise 2a

The file nurseryrhyme.txt contains the text for the nursery rhyme «Mary had a little lamb».

Write a program that opens the text file and extracts all the words, storing them in a list called `words`.

The final list of words should look like the following:

```
words = ['Mary', 'had', 'a', 'little', 'lamb,', 'whose',
'fleece', 'was', 'white', 'as', 'snow.', 'And', 'everywhere',
'that', 'Mary', 'went,', 'the', 'lamb', 'was', 'sure', 'to',
'go.', 'It', 'followed', 'her', 'to', 'school', 'one', 'day',
'which', 'was', 'against', 'the', 'rules.', 'It', 'made', 'the',
'children', 'laugh', 'and', 'play,', 'to', 'see', 'a', 'lamb',
'at', 'school.', 'And', 'so', 'the', 'teacher', 'turned', 'it',
'out,', 'but', 'still', 'it', 'lingered', 'near,', 'And',
'waited', 'patiently', 'about,', 'till', 'Mary', 'did',
'appear.', '"Why', 'does', 'the', 'lamb', 'love', 'Mary', 'so?"',
'the', 'eager', 'children', 'cry.', '"Why,', 'Mary', 'loves',
'the', 'lamb,', 'you', 'know."', 'the', 'teacher', 'did',
'reply.']
```

Exercise 2b

Notice that some of the words stored in the list `words` in the previous exercise are capitalized and/or include special characters such as «.» and «?».

Write a function called cleanWord that takes a string as an input and that cleans the string by making all letters lowercase and removing characters that are not letters. Remember proper function documentation.

Use the function to clean the words in `words` and store the cleaned words in a new list called `cleaned_words`.

1. Write an algorithm for the cleanWord function.

2. Implement the function using Python.

Exercise 2b

Solution proposal for the algorithm:

Input: word
Output: clean_word

clean_word ← empty word
for each character in word **do**:
    if character is a letter **then**
        Add character to clean_word
    end if
end for
Return clean_word

Exercise 2c

Notice that «Mary has a little lamb» contains several duplicate words, e.g. «Mary».

Use the cleaned list of words from the previous exercise and count the number of unique words in the nursery rhyme.

1. Write an aglorithm for the solution.

2. Implement the solution using Python.

Exercise 2c

Solution proposal for the algorithm:

Input: words (a list of words)
Output: the number of unique words in words

unique_words ← an empty list
for each word in words **do**
    if word is not in unique_words **then**
        Add word to unique_words
    end if
end for
Return the number of words in unique_words