Exercise 03: Lists, For-loops, Dictionaries

PCL 1 / PfL, Fall Semester 2022

Deadline: 02.11.22, 18:00

Remarks on submission

- Always name your submissions as follows: `olatusernames_(pcl1|pfl)_exnumber.txt/pdf/py/zip`
- Learning partnerships in pairs are **mandatory**. Both students have to submit the exercise and name it `username1_username2_(pcl1|pfl)_exnumber.txt/pdf/py/zip`
- Please state first and last names of both students on the submission form.
- Write many comments! They don't just help the tutors to understand what you're thinking, but can also help you to find and fix possible bugs.
- Please number the tasks on the submission sheet/Python programs the same as on the task sheet.
- **Hand in:** by **November 2nd at 18:00**. Make sure to hand in on time!

1 Python Lists

1.1 Manipulating Lists

Have a look at the official Python documentation of lists for this exercise (3.1.3 Lists and 5.1 More on Lists). For this task, a program `task1_1.py` is available, which contains the list `aquarium`. Complete the script so that it:

a) counts the occurrences of the token 'castle' and displays the number,

b) removes the second occurrence of 'castle' from the list `aquarium`,

c) adds the token 'golfball' at the end of the list `aquarium`,

d) adds the token 'cat' at the fifth position of the list `aquarium`,

e) will output the index number of 'goldfish',

f) replaces 'goldfish' with 'oh no' in `aquarium`,

g) displays the number of elements in the list `aquarium`,

h) print your list `aquarium` to the command line,

Additional tasks: The following exercises are optional, but give good practice using lists

i) defines the (new) list `terrarium`,

j) adds the sixth element of the list `aquarium` into the list `terrarium`,

k) adds the token 'python' at any position in the list `terrarium`,

l) sorts the list `terrarium`. You now should have the description of two beautiful exhibits. Print both lists to the command line.

Try to keep the solution as general as possible, e.g. d) should still function correctly if additional elements are inserted at the beginning of the list.

Comment your script reasonably. Please submit a complete, executable progr.

1.2 Building Lists & Comparison Operators

This task will involve receiving user input, comparing it using operators, and saving it in lists. Finally, the lists must be sorted and outputted.
Your program `task1_2.py` should do the following:

- It asks the user for any number.
- It ensures that no letters or words are accepted (hint: utilize the `.isalpha()` method. This method checks whether a given character is a letter and returns true or false. For example: `'d'.isalpha() = True` but `'1'.isalpha() = False`)
- It checks whether the number is positive (count 0 as a positive number).
- It repeats the query until more than five negative numbers have been entered.
- If this threshold is reached, the script returns both the list of positive and the list of negative numbers. Both lists should be sorted by size, starting with the number closest to zero.

Comment your script reasonably. Please submit a complete, executable program.

2 Python For-Loops and Strings

Now you should write a program `task2.py` that accepts a string via the standard input and then checks if the given string is a palindrome. A palindrome is a word, phrase, or sequence that reads the same backwards as forwards, e.g. 'madam' or 'Sugus'. Solve this exercise with a for-loop and make sure to make the palindrome-checker **case-insensitive**. The final output could look like this:

```
»Enter a word:  Sugus
»This word is a palindrome.

»Enter a word:  computer
»This word is not a palindrome.
```

Comment your script reasonably. Please submit a complete, executable program `task2.py`.

3 Python Dictionaries

Imagine you are a computer collector with a special taste for old and original machines. As your collection grows, you realize that you need a system to keep track of your possessions. To do this, you create a Python dictionary where the name of the product is the key. The value is the year of the introduction and the company that released it (e.g., Canon Inc.).

```
computercollection = {'Apple II Plus':  [1979,'Apple Inc.'], 'Lisa':  [1983,'Apple Inc.'],
'Canon Cat':  [1987,'Canon Inc.'], 'Tandy 1000':  [1984,'Tandy Corporation'], 'Apple
Newton':  [1993,'Apple Inc.']}
```

a) You just got lucky at an auction! Add two more entries (key-value pairs) to your dictionary. [You can invent them or google and see which ones you would like to add.]

b) The question people most commonly ask you is: Which is the oldest computer in your collection? Because you get tired of repeating yourself, you create an automated answer. Print a sentence that says: 'I currently have [n] computers in my collection. The oldest is the [name of product] which was introduced in [year of introduction].', e.g.:

```
I currently have 17 computers in my collection.  The oldest is the Fondue500 which
was introduced in 1991.
```

c) Finally, you want to sort your collection by company. Output a simple table that shows the company name first and then the name of the product, ordered alphabetically by company names. The year of introduction is not shown. Your output should look something like this:

```
flindows   SmartProductName
flindows   AnotherProductName
schmapple   FruitRelatedProductName
```

Additional challenge: To add to c): Not only sort your collection automatically, but make it count the machines per company for you too. Your output should be the sentence 'The company I own most machines of is [company name]: I own [number] computers by them.'.

Comment your script reasonably. Please submit the complete, executable program `task3.py`.

4 Read files and determine word frequencies

For this task, the book 'Dune' by Frank Herbert is available as a `.txt`-file (`dune.txt`). Write a Python program that will expect the file path or the name (if the file is in the same folder as the program) of a file as a command-line argument. The goal is to find words that frequently occur in the context of another word. We are looking to find words in the context of the word 'dune'. Thus, the program call should look like this:

```
$ python3 ex4.py dune.txt
```

Use the provided book, but your program should work for every file in this format.

Execute the following operations:

a) Tokenize each line with the function `.split()`. Get rid of any punctuation and whitespace (hint: use the methods `.lstrip()`/`.rstrip()`) and convert every word to lowercase.

b) Not all words in a text have the same semantic importance. Words that add only little or no semantic information are called stop words and they can easily be removed from a text without any loss of semantic information. Use the provided set `stopwords` to remove all stop words from the tokenized text.

c) Now create a dictionary named 'frequency'. Iterate over the tokenized text. For every token, count its total occurrences within the text. Hint: If you find a new word, set the word as a new key and set its value to 1.

d) Sort the dictionary by values and output it to the command line. You can achieve this either by using UNIX-Commands or by putting the key-value pairs into a list in python and sorting that list.

Additional challenge: Create a context-word dictionary for the word 'desert'. A context-word is any word that appears to the left or right of the word within a text. Save all context-words of 'desert' in a dictionary and count how often they appear in context with the word 'desert'.

Comment your script reasonably. Please submit the complete, executable program `task4.py`.

Reflection/Feedback

a) Summarize your discoveries and your learning progress in two sentences.

b) How long have you worked on the tasks of this exercise?