INDEX PAGE
Pages
Lab 1
FAQ
Lab 2
Lab 3
Lab 7
Lab 6
Testing tips
Lab 4
Lab 10
Lab 5
Module overview
Support
Installing and finding Python
Snippets
Lecture slides
Lab 8
Lab 9
First steps
Testing demo

Home | Lab 5

Laboratory 5: Lists, Files, Exceptions, Strings

Prerequisites: +exceptions, +string handling

Contents

PEP8

To monitor the coding style we turn on the PEP8 style guide monitoring in Spyder by:

1. Going to preferences in Spyder menu
2. Clicking on `Editor` in the selection list on the left
3. Clicking on `Code Introspection/Analysis` (top right corner)
4. Ticking the box `Real-time code style analysis`
5. Clicking `Apply` and `OK` (bottom right corner)

Exercises

Provide the following functionality in a file `lab5.py`:

1. Write a function `count_sub_in_file(filename,s)` that takes two arguments: the substring `s` (of type string) and a `filename` (of type string). The function should return the number of occurrences of `s` in the file given through filename. (Note that each string object has a method `count` that should be used here.)

    - You can test your function `count_sub_in_file` on *Alice's Adventures in Wonderland* which you can download from http://www.gutenberg.org/files/28885/28885-8.txt

        - How often does the substring "Alice" occur? Expect several hundred.
        - How often does the substring "alice" occur? Expect a very small number.

      (On Mac OS X and Linux, you can use the `wget http://www.gutenberg.org/files/28885/28885-8.txt` command from a shell to download the file. Otherwise use your browser and save the file in the same directory as your `lab5.py` file.)

2. Modify the function `count_sub_in_file(filename, s)` so that if the file with name `filename` cannot be opened, the value -1 is returned (instead of the number of substrings `s`).

3. Write a function `count_vowels(s)` that returns the number of letters 'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U' in a given string `s` (the return value is of type integer).

    Examples:

    ```
    In [ ]: count_vowels('This is a test')
    Out[ ]: 4

    In [ ]: count_vowels('aoeui')
    Out[ ]: 5

    In [ ]: count_vowels('aoeuiAOEUI')
    Out[ ]: 10

    In [ ]: count_vowels('N0 v0w3ls @t @ll ln thls strlng')
    Out[ ]: 0
    ```

4. Write a function `vector_product3(a, b)` that takes two sequences of numbers. Both sequence `a` and sequence `b` have three elements. With inputs `a=[ax, ay, az]` and `b=[bx, by, bz]`, the function

should return a list which contains the vector product of 3d-vectors `a` and `b`, i.e. the return value is the list:

```
[ay * bz - az * by, az * bx - ax * bz, ax * by - ay * bx].
```

Examples:

```
In [ ]: vector_product3([1, 0, 0], [0, 1, 0])
Out[ ]: [0, 0, 1]

In [ ]: vector_product3([1, 2, 4], [3, 5, 6])
Out[ ]: [-8, 6, -1]
```

5. Write a function `seq_mult_scalar(a,s)` which takes a list of numbers `a` and a scalar (i.e. a number) `s`. For the input `a=[a0, a1, a2,.., an]` the function should return `[s * a0, s * a1, s * a2, ..., s * an]`.

   Example:

   ```
   In [ ]: seq_mult_scalar([-4, 9, 1], 10)
   Out[ ]: [-40, 90, 10]
   ```

6. A function `powers(n,k)` that returns the list `[1,n,n^2,n^3,...,n^k]` where `k` is an integer. Note that there should be k+1 elements in the list.

   Example:

   ```
   In [ ]: powers(2, 3)
   Out[ ]: [1, 2, 4, 8]

   In [ ]: powers(0.5, 2)
   Out[ ]: [1.0, 0.5, 0.25]
   ```

7. A function `traffic_light(load)` that takes a floating point number `load`. The function should return the string:

   - `"green"` for values of `load` below 0.7.
   - `"amber"` for values of `load` equal to or greater than 0.7 but smaller than 0.9
   - `"red"` for values of `load` equal to 0.9 or greater than 0.9

   Example:

   ```
   In [ ]: traffic_light(0.5)
   Out[ ]: 'green'
   ```

Then submit `lab5.py` by email with the subject lab 5 for automatic testing of this laboratory session.

Last updated: 2019-01-06 at 17:34

Return to Top

Page last modified Fri Nov 9 15:49:23 2018. |