Homework - Serie 7

Kevin Sturm

Python 3: hand-in on May 12

Test your code with examples!

Problem 1. Strings.

(a) Write a function which gets as an input three strings `s_in`, `s_find` and `s_replace`. The return value of the function is `s_out` where the string `s_out` consists of all words of `s_in` in which all occurring words `s_find` are replaced by `s_replace`.

(b) Write a function `f` which takes a string s and removes all its spaces and puts the resulting words in list. Let the function have an optimal argument which if set to `True` capitalises the input string.

Problem 2. Classes. I.

- Write a class `Complex` with methods `add`, `multiply` and `divide` which realises the addition, multiplication and division of two complex numbers `z1` and `z2`. Define complex numbers by its real and imaginary parts and use python tuple, i.e., `z = (imag, compl)`. (Do not use the build in complex numbers of python). The constructor `__init__` should initialise `z1` and `z2`.

Problem 3. Classes. II.

(a) Write a class `Vector` with methods `add(z1, z2)` and `scalar(a, z1)` which realise the addition and scalar multiplication of two lists `z1` and `z2` and the scalar `a` and the vector `z1`, respectively.

(b) Write an inherited class of `Vector` named `VectorPlus` which additionally has the functions `vector_prod(z1, z2)` and `tensor(z1, z2)` realising the tensor and vector product of two lists `z1` and `z2`.

Problem 4. Decorators. I.

(a) Write a decorator `dec(ev, fun)` which gets a function `fun` and returns a function that evaluates `fun` at `ev`. Test your code with the functions `sin` and `exp` of the standard library `math`.

(b) Write a decorator `comp(l)` which takes a list of functions `l` and returns a function which is the composition of all the functions in the list. Example: if `l` is a list containing f_1 and f_2 the decorator should return the function $f_1 \circ f_2$. Test your code with some functions of the `math` library.

Problem 5. Decorators. II.

- Let `f` be a python function. Write a decorator `count` which counts how often the function `f` was called. Test your program with `sin` and `cos` of the math library. Example: with `f = count(sin)` the call `f(0.1)` should return 1 and `sin(0.1)` and another call `f(0.2)` would return 2 and `sin(0.2)`. HINT: in the inner definition of the decorator define the 'counter' variable which counts the function calls as `nonlocal` (syntax: `nonlocal counter`). This makes the variable 'counter' available in the outer function definition.

Problem 6. Doc String

- Write a detailed doc string documentation for the classes of exercise 2 and 3. Test your code by call help in the console as well as calling the functions and the module with ".__doc__"!

Problem 7. (a) Given an ordered dict, write a programm to insert items in beginning of an ordered dict.
Sample input: `original_dict = {'a':1, 'b':2} item to be inserted ('c', 3)`
Ouput: `{'c':3, 'a':1, 'b':2}`

(b) Given three lists sorted in non-decreasing order, print all common elements in these lists.
Sample input: `l1 = [1, 5, 5], l2 = [3, 4, 5, 5, 10], l3 = [5, 5, 10, 20]`
Output: `[5,5]`

(c) Convert a key-value list dictionary to list of lists.
Sample input: `{'gfg': [1, 3, 4], 'is': [7, 6], 'best': [4, 5]}`
Output: `[['gfg', 1, 3, 4], ['is', 7, 6], ['best', 4, 5]]`

(d) Given a nested dictionary (see example below), perform an inversion of the keys, i.e the innermost nested becomes outermost and the other way around.
Sample input:
`test_dict = {"a" : {"b" : {}}, "d" : {"e" : {}}, "f" : {"g" : {}}};`
Output: `{'b': {'a': {}}, 'e': {'d': {}}, 'g': {'f': {}}}`

Problem 8. Let $f : [a,b] \to \mathbf{R}$ be a continuous function with $a < b$. For $N \in \mathbf{N}$ we subdivide $[a,b]$ into equidistant intervals $a = x_0 < x_1 < \cdots < x_{N-1} < x_N = b$, where

$$x_j := a + j\,\frac{(b-a)}{N}, \qquad \text{for } j = 0, \dots, N$$

We then define the *composite midpoint rule*

$$I_N := \frac{b-a}{N} \sum_{j=1}^{N} f\big((x_{j-1} + x_j)/2\big).$$

Since I_N is a Riemann sum, we know that

$$\lim_{N\to\infty} I_N = \int_a^b f\,dx.$$

If f is two time continuously differentiable function $f : [a,b] \to \mathbf{R}$, then one can even show that there is $C > 0$, such that

$$\Big|\int_a^b f\,dx - I_N\Big| \le C N^{-2} \qquad \text{for } N \to \infty. \tag{1}$$

(a) Write a python function

```
midpointrule(a,b,f,n)
```

which, for the sequence $N = 2^k$ and $k = 0, \dots, n$, computes and returns the vector `V` of the corresponding values I_N.

(b) Think about how you can test your code! What are suitable test examples? Experimentally verify the order of convergence order given in (1).

Hint: Test your quadrature with polynomials of different degree. Calculate the result analytically (with pen and paper). What do you notice?