Advanced Python: Homework set 1

2023/2024

Problem 1

In Poland, the tax on goods and services (VAT) is calculated in two ways: in the case of invoices, the net values are summed and multiplied by 23%, and in the case of cash register fiscal records and receipts, the VAT of 23% is calculated separately for each item and at the end it is summed. Program two functions in Python to return VAT for given shopping list

1. `vat_invoice(list)`
2. `vat_receipt(list)`

where `list` is a list of numbers representing the net price. We usually expect that the following program will print `True`

```
print(vat_invoice(shopping) == vat_receipt(shopping))
```

where `shopping` is a list of float numbers. Look for a shopping list for which the above program will print `False` and place this list in the source file. Do a number swapping experiment: convert `float` in the shopping list to their `Decimal` class counterparts and check if the program in the frame still prints `False`

Problem 2

Write a function `is_palindrome(text)` that returns `True` if the argument is a palindrome. We assume that text can be either a single word (e.g. “rotor” or “eye”), but also with a longer expression: ”Kobyła ma mały bok.”; In this case, we ignore punctuation marks, spaces, and case. Check if the function works correctly for foreign language texts:

```
    is_palindrome("Eine g ldne, gute Tugend  L ge nie!")
    is_palindrome(" M     omo   m.")
```

Sorry, it seems that I have a problem with utf-8 here, those two sentences are: “Eine güldne, gute Tugend: Lüge nie!” and “Míč omočím.”.

Problem 3

On November 17 this year will be World Multiplication Table Day. Program the table function (x1, x2, y1, y2, d), which will print the multiplication table for numbers $[x_1, x_1 + d, x_1 + 2 * d, \ldots, x_2] \times [y_1, y_1 + d, y_1 + 2 * d, \ldots, y_2]$ where x_1, x_2, y_1, y_2 and d are `float` numbers. For example,

```
array(3.0, 5.0, 2.0, 4.0, 1.0)
```

should print

```
         3.0   4.0   5.0
   2.0   6.0   8.0  10.0
   3.0   9.0  12.0  15.0
   4.0  12.0  16.0  20.0
```

Make sure that the column widths are the same and appropriate to the number of digits in the numbers. We assume that x_1, x_2, y_1, y_2 can also be negative numbers.

Problem 4

The number π (or rather its subsequent approximations) can be calculated in many ways. One of them is to repeatedly throw a dart at a square target with a circle inscribed on it and count the number of hits inside the circle (ltwo) and the total number of hits in the target (cltwt). These two numbers will allow us to calculate an approximation of π:

$$\pi \approx \frac{4 * \text{ ltwo}}{\text{cltwt}}$$

Write a simulation of throwing a dart on a dartboard by randomly selecting the coordinates of a point on the dartboard. The program should print subsequent approximations of π after each throw. The program may end after the specified number of draws has been completed or when the difference between the obtained approximation and the `math.pi` value is smaller than the specified value.

Problem 5

Write a function that, for a given list of strings, `word_list`, returns the longest common prefix for at least three elements `word_list`. For example,

```
common_prefix(["Cyprian", "cyberotoman", "cynik", "ceniac", "czule"])
```

should return `"cy"` Letter case does not matter to us.