Практическое занятие №2.

Работа с вещественными числами и векторами

При решении следующих задач предлагается выполнять автоматическую проверку синтаксиса кода на соответствие стандарту PEP8. Такая практика поможет Вам научиться писать код на языке Python сразу без синтаксических ошибок!

In [ ]:
```
# установка библиотеки для проверки кода на соответствие PEP8
!pip install flake8 pycodestyle_magic
```

In [ ]:
```
# теперь код будет проверяться на соответствие PEP8
%load_ext pycodestyle_magic
%pycodestyle_on
```

1. Вычислить значение итерационной функции:

$$f(n, m, a) = \prod_{c=1}^{a} \prod_{j=1}^{m} \sum_{i=1}^{n} \left(\frac{(28c^2)^6}{5} + 16 \left(\frac{j^3}{44} + i^2 \right)^5 \right), \tag{1}$$

при $n = 4, m = 2, a = 8$. Код должен соответствовать PEP8.

In [ ]:

Пусть даны два вектора $\vec{y} \in \mathbb{R}^n$ и $\vec{z} \in \mathbb{R}^n$, векторы имеют вид $\vec{y} = (y_1, y_2, \ldots, y_n)$ и $\vec{z} = (z_1, z_2, \ldots, z_n)$. С помощью Python:

2. Вычислить евклидово расстояние между двумя векторами $\vec{y}$ и $\vec{z}$:

$$d_1(\vec{y}, \vec{z}) = \sqrt{\sum_{i=1}^{n} (y_i - z_i)^2}, \tag{2}$$

при $\vec{y} = (1, 0.5, 1)$, $\vec{z} = (0.5, 2, 1)$. Код должен соответствовать PEP8.

In [ ]:

3. Вычислить манхэттенское расстояние между двумя векторами $\vec{y}$ и $\vec{z}$:

$$d_2(\vec{y}, \vec{z}) = \sum_{i=1}^{n} |y_i - z_i|, \tag{3}$$

при $\vec{y} = (1, 0.5, 1)$, $\vec{z} = (0.5, 2, 1)$. Код должен соответствовать PEP8.

In [ ]:

4. Вычислить расстояние Чебышева между двумя векторами $\vec{y}$ и $\vec{z}$:

$$d_3(\vec{y}, \vec{z}) = \max |y_i - z_i|, \tag{4}$$

при $\vec{y} = (1, 0.5, 1)$, $\vec{z} = (0.5, 2, 1)$. Запрещается использование функции `max()`. Код должен соответствовать PEP8.

In [ ]:

5. Вычислить квадрат евклидового расстояния между двумя векторами $\vec{y}$ и $\vec{z}$:

$$d_4(\vec{y}, \vec{z}) = \sum_{i=1}^{n} (y_i - z_i)^2, \tag{5}$$

при $\vec{y} = (1, 0.5, 1)$, $\vec{z} = (0.5, 2, 1)$. Код должен соответствовать PEP8.

In [ ]:

6. Вычислить расстояние Минковского между двумя векторами $\vec{y}$ и $\vec{z}$:

$$d_5(\vec{y}, \vec{z}) = \left(\sum_{i=1}^{n} |y_i - z_i|^h \right)^{1/h}, \tag{6}$$

при $\vec{y} = (1, 0.5, 1)$, $\vec{z} = (0.5, 2, 1)$ для $h = 5$. Код должен соответствовать PEP8.

In [ ]:

7. `matplotlib` — библиотека на Python для визуализации данных. Полученные с помощью библиотеки `matplotlib` изображения часто используются в научных публикациях, в графическом интерфейсе пользователя в веб-приложениях, мобильных приложениях, приложениях для ПК. Функция, позволяющая сравнить абсолютные значения штрафов реализованных мер расстояния при удалении векторов друг от друга, реализованная с помощью `matplotlib`, имеет вид:

In [ ]:
```
import matplotlib.pyplot as plt


def visualize(distance_metrics, y, z, move=1):
    moved_z = [i + move for i in z]
    distance_differences = []
    for distance in distance_metrics:
        distance_before_move = distance(y, z)
        distance_after_move = distance(y, moved_z)
        distance_difference = distance_after_move - distance_before_move
        distance_differences.append(distance_difference)
    x = range(0, len(distance_differences))
    figure, axis = plt.subplots()

    # Построение гистограммы с подписями.
    axis.bar(x, distance_differences)
    axis.set_xticks(x, labels=[f'd_{i + 1}' for i in x])
    # для тех, кто работает в средах, отличных от jupyter-подобных - раскомментировать строку ниже
    # plt.show()
```

Вызовите функцию `visualize`, передав в качестве аргументов реализованные меры расстояний и два вектора:

$\vec{y} = (1, 0.5, 1)$, $\vec{z} = (0.5, 2, 1)$.

Сравните результаты при различных значениях параметра `move`, например, при $1, 0.01, 1.5$.

In [ ]:

Работа со строками

Пусть дан набор `words` из n строк $(s_1, s_2, \ldots, s_n)$.

In [ ]:
```
words = ["language!", "programming", "Python", "the", "love", "I"]
```

8. Реализовать функцию, разворачивающую подаваемый на вход список и преобразующую результат в строку, разделённую пробелами. Полученная строка не должна заканчиваться пробелом. Код должен соответствовать PEP8.

In [ ]:

9. Реализовать функцию $f(s, c)$, вычисляющую, сколько раз символ c встречается в полученной в задаче №8 строке s. Используя полученную функцию $f(s, c)$ реализовать функцию `count_characters`, возвращающую словарь `{}`, в котором ключами являются символы c_i, а значениями являются результаты выполнения функции $f(s, c_i)$. Буквы в разном регистре (например, `I` и `i`) считаются одним и тем же символом, пробелы не учитываются.

In [ ]:

10. Каким образом поисковым системам удаётся находить подходящие результаты даже если пользователь допускает одну или несколько опечаток? Каким образом можно определить, насколько похожи две различные последовательности символов? Одним из способов определения сходства последовательностей является расстояние Левенштейна.

Формула вычисления расстояния Левенштейна имеет вид:

$$f(\mathbf{a}, \mathbf{b}, i, j) = \begin{cases} \max(i, j), & i = 0 \text{ или } j = 0 \\ f(\mathbf{a}, \mathbf{b}, i-1, j-1), & a_{i-1} = b_{j-1} \\ 1 + \min\left(f(\mathbf{a}, \mathbf{b}, i, j-1), f(\mathbf{a}, \mathbf{b}, i-1, j), f(\mathbf{a}, \mathbf{b}, i-1, j-1)\right), & \text{иначе} \end{cases}$$

где $\mathbf{a}$ и $\mathbf{b}$ — сравниваемые последовательности символов, например, строки;
i и j — номера удаления и вставки соответственно;
a_{i-1} — $(i-1)$-й символ в строке $\mathbf{a}$;
b_{j-1} — $(j-1)$-й символ в строке $\mathbf{b}$.

Реализуйте расстояние Левенштейна на Python по рекуррентной формуле выше.

Сравните попарно следующие строки:

- `Hello, world!` и `Goodbye, world!`,
- `Hello, world!` и `Bye, world!`,
- `I love Python` и `I like Python`,
- `100101` и `100011`.

In [ ]:

11. С помощью модуля `timeit` измерьте производительность реализованного алгоритма для строк различной длины. Попробуйте улучшить производительность полученного кода, заменив рекурсивные вызовы циклами. Установите пакет `python-Levenshtein`, содержащий реализацию расстояния Левенштейна на языке C (дополнительную информацию можно почитать здесь), импортируйте функцию `from Levenshtein import distance` и измерьте производительность библиотечной реализации.

In [ ]:

Выполнение данных заданий в течение практического занятия №2 даёт возможность получить следующие баллы:

- **Любые 4 задачи: 0,5 балла**
- **Любые 7 задач: 1 балл**

- **Любые 9 задач: 2 балла**
- **Все 11 задач: 3 балла**