НУМЕРАЦІЯ ЕЛЕМЕНТІВ СПИСКУ

- Нумерація елементів в списку
  - Від'ємна нумерація
- Функція *len()*

1. (***Перші 10 елементів***) Написати програму, яка виводить на екран перші 10 елементів списку, якщо кількість елементів списку 10 або більше. При кількості елементів меншій за 10 вивести на екран "*Number of elements is less than 10*.".

   Вказівка: при виводі на екран бажано використати цикл

   Приклади:

   l = [1, 2, 4, "hello", "abc", 10, -1]

   Output:
   "Num"

   Джерело: Projekt MmF

2. (***Перший чи останній***) Для заданого списку чисел визначити, який елемент більший: перший чи останній. Вивести на екран більше число і вказати воно перше (*first*) чи останнє (*last*).

   Приклади

   l = [1, 4, 3, 7, 8, -1, 0]

   Вивід: 1 first

   Джерело: Projekt MmF

3. (***Сума останніх***) Для заданого списку вивести на екран суму останніх трьох чисел.

   Приклади

   l = [1, 4, 3, 7, 8, -1, 0, 4, 2]

   Вивід: 6

Джерело: Projekt MmF

4. (***Зворотній напрямок***) Вивести на екран елементи списку в зворотному напрямку.

   Приклади

   l = [1, 4, 3, 7, 8, -1, 0]

   Вивід:
   0
   -1
   8
   7
   3
   4
   1

   Джерело: Projekt MmF

5. (***Середина***) Для заданого списку визначити середній елемент, якщо кількість елементів списку непарна, або визначити два середніх елементи, якщо кількість парна.

   Приклади

   l = [1, 4, 3, 7, 8, -1, 0, 10]

   Вивід: 7 8

   Джерело: Projekt MmF

6. (***Більша половина***) Для заданого списку визначити, в якій половині сума елементів більша: в правій чи в лівій. Вивести на екран більшу суму і якій вона відповідає половині.

   Вказівка: вважаємо, що кількість елементів в списку є парною

   Приклади

   l = [1, 4, 3, 7, 8, -1, 0, 10]

   Вивід: 17 right

   Джерело: Projekt MmF

7. (***Two Sum***) Нехай задано список чисел *nums* і число *target*. Необхідно знайти в списку *nums* два числа, сума яких дорівнює *target*. Вивести на екран індекси цих чисел.

   Приклад

   nums = [1, 4, 5, 10, 2, 15]
   target = 3

   Вивід: 0 4

   Джерело: *https://leetcode.com/problems/two-sum/*
   ***Теми**: списки, математика*

8. (***Сума деяких елементів списку***) Написати функцію *summarize(nums, in_sum)*, яка приймає як параметри список чисел *nums* і список булевих значень *in_sum* такої ж довжини. Функція повертає суму тих елементів списку *nums*, яким відповідає *True* зі списку *in_sum*.

   Приклад

   summarize([1, 2, 3], [True, False, True]) → 4

   Джерело: Projekt MmF
   ***Теми**: списки, математика, boolean*

ЦИКЛ FOR ДЛЯ СПИСКІВ. ПОШУК В СПИСКУ

1. (***max and min***) Написати дві функції *maximum* і *minimum*, які знаходять відповідно найбільше і найменше числа в заданому списку.

   Вказівка: розв'язати задачу не використовуючи вбудованих функцій

   Приклад:

   maximum([0, -1, 3, 2, 10, 5]) → 10
   minimum([0, -1, 3, 2, 10, 5]) → -1

   Джерело: *https://www.codewars.com/kata/577a98a6ae28071780000989*

2. (***Додатні і від'ємні***) Знайти кількість додатних і від'ємних елементів списку.

   Приклад:

   l = [1, 4, 3, -7, 8, -1, 0, -10]

   Вивід:

Positive: 4
Negative: 3

Джерело: Projekt MmF

3. (***Потрапляння в інтервал***) Написати програму, яка знаходить для заданого списку кількість його чисел, які належать інтервалу [-10, 10].

   Приклад:

   l = [1, 11, 23, -7, -15, 0, -2, 30]

   Вивід:
   4

   Джерело: Projekt MmF

4. (***Сума чисел***) Написати програму, яка знаходить суму чисел в заданому списку.

   Приклад:

   l = [1, 2, 3, 4, 5]

   Вивід:
   15

   Джерело: Projekt MmF

5. (***Слова з літери “a”***) Написати програму, яка для даного списку слів *words* знайде кількість слів, що починаються з літери “a”, і виведе їх на екран.

   Приклад*:*

   words = [“above”, “under”, “aqua”, “hello”, “aunt”]

   Вивід:
   3
   “above”
   “aqua”
   “aunt”

   Джерело: Projekt MmF

6. (***Найдовше слово***) Напистиі програму, яка в заданому списку слів, визначить найдовше слово. Вивести слово і його довжину на екран.

***Приклад**:*

words = [“above”, “under”, “aqua”, “hello”, “aunt”, “school”]

Виєід:
school 6

Джерело: Projekt MmF

7. (***Longest common prefix***) Написати програму, яка для заданого списку слів *words* визначає найдовший початок, який є спільним для всіх слів списку *words*.

   Приклади:

   words = [“hi”, “hello”, “hundred”])
   Output: “h”

   words = [“fridge”, “friend”, “fries”, “friday”])
   Output: “fri”

   Джерело: *https://leetcode.com/problems/longest-common-prefix/*

ПЕРЕТВОРЕННЯ СПИСКІВ

- Функції *append()* і *remove()*

1. (***Swap first and last***) В заданому списку змінити місцями перший і останній елементи. Вивести оновлений список на екран.

   Приклади

   l = [1, 4, 3, 7, 8, -1, 0]

   Виєід:
   [0, 4, 3, 7, 8, -1, 1]

   Джерело: Projekt MmF

2. (***Алгоритм сортування***) Написати програму, яка відсортує заданий список в порядку зростання.

   Вказівка: для цієї задачі придумати свій алгоритм. Не використовувати вбудованих функцій.

   Приклад

l = [3, 1, 4, 2, 7]

Вивід: [1, 2, 3, 4, 7]

Джерело: Projekt MmF

3. (***Створення вектора***) Написати функцію, яка повертає вектор з наступною характеристикою.
    a. Вектор довжини *n*, що складається з нулів та однієї одиниці на *k*-ому місці.
    b. Вектор, що містить числа від 0 до *n-1*

***Приклади**:*

zero_one(5, 2) → [0, 0, 1, 0, 0]

Джерело: Projekt MmF, *https://www.codewars.com/kata/5a00e05cc374cb34d100000d*

4. (***Дільники числа***) Для заданого числа створити список його дільників.

Приклад

num = 24

Вивід:
[1, 2, 3, 4, 6, 12, 24]

***Джерело**: Projekt MmF*
***Тема**: списки, математика, теорія чисел*

5. (***Прості числа***) Скласти список простих чисел, які менші за 1000.

Вказівка: використати результат із задачі “дільники числа” для функції, яка перевіряє число на простоту

Джерело: Projekt MmF

6. (***Попадання в інтервал 2***) Створити функцію *is_in_interval(nums)*, яка для заданого вектора чисел *nums* повертає список такої ж довжини, де для кожного числа зі списку *nums* стоїть *True*, якщо воно потрапляє в інтервал [-10, 10] і *False* в інакшому випадку.

***Приклади**:*

is_in_interval([0, 5, -12, 20, 3, -2, 100]) → [True, True, False, False, True, True, False]

Джерело: Projekt MmF
***Теми**: списки, математика, boolean*

7. (***Слова з тексту***) Написати програму, яка створює список слів, які використано в заданому тексті (слова можуть повторюватися).

   Вказівка: вважаємо, що між словами знаходиться рівно один пробіл

   Приклад:

   s = 'My name is Ivan'

   Вивід:
   ['My', 'name', 'is', 'Ivan']

   Джерело: Projekt MmF

8. (***Remove duplicates***) Написати програму, яка для заданого списку чисел *nums* видалить повторення елементів.

   Приклад

   l = [1, 2, 1, 3, 4, 3, 4]

   Вивід: [1, 2, 3, 4]

   Джерело: *https://www.codewars.com/kata/57a5b0dfcf1fa526bb000118*

9. (***Pascal triangle***) Трикутник Паскаля будується наступним чином. В першому ряді трикутника знаходяться два числа: 1 і 1. Кожен наступний ряд починається і закінчується числом 1 та містить всередині суми сусідніх чисел з попереднього ряду (порядок чисел в кожному ряді важливий):

   ```
         1 1
       1 2 1
     1 3 3 1
   1 4 6 4 1
 1 5 10 10 5 1
1 6 15 20 15 6 1
   ```

   Написати фукнцію *pascal_triangle(n)*, яка приймає натуральне число *n* і друкує на екрані перші *n* рядків трикутника Паскаля.

   Приклади:

   pascal_triangle(3)

[1, 1]
[1, 2, 1]
[1, 3, 3, 1]

Джерело: https://leetcode.com/problems/pascals-triangle/

ДВОВИМІРНІ СПИСКИ

- Зауваження про створення двовимірних списків. Копіювання списків

1. (***Розмір матриці***) Написати функцію *dims(matr)*, яка для заданої матриці (двовимірного списку) повертає її розмірність: кількість рядків і кількість стовпчиків.

   Приклад

   dims([
   [1, 2],
   [2, 3],
   [5, 6]
   ]) → [3, 2]

   Джерело: Projekt MmF

2. (***Сума в рядку/стовпчику***) Написати код, який для заданої матриці виводить суму її елементів в заданому стовпчику або в заданому рядку.

   Джерело: Projekt MmF

3. (***Matrix trace***) Слідом квадратної матриці називається сума її елементів, які стоять на головній діагоналі. Написати функцію, яка знаходить слід заданої квадратної матриці (двовимірного списку).

   Джерело: *https://www.codewars.com/kata/55208f16ecb433c5c90001d2*

4. (***Одинична матриця***) Написати функцію *unit_matrix(n)*, яка повертає одиничну (квадратну) матрицю заданого розміру.

   Приклад*:*

   unit_matrix(2) →
   [[1, 0],
   [0, 1]]

   Джерело: *https://www.codewars.com/kata/5a34da5dee1aae516d00004a*

5. (***Diagonals sum***) Написати функцію, яка повертає суму елементів головної і побічної діагоналей квадратної матриці.

   Вказівка: спільний елемент врахувати двічи

   Приклад:

   sum_diagonals([
   [1, 2, 3],
   [4, 5, 6],
   [7, 8, 9]
   ]) → 30

   Джерело: *https://www.codewars.com/kata/5592fc599a7f40adac0000a8*

6. (***Diagonals***) Написати функцію, яка для заданої квадратної матриці повертає список всіх її можливих діагоналей, записаних згори вниз.

   Приклад:

   diagonals([
   [1, 2],
   [4, 5]
   ]) → [[1, 5], [2], [4], [1], [2, 4], [5]]

   Джерело: *https://www.codewars.com/kata/5592dd43a9cd0e43a800019e*

7. (***Максимум в матриці***) Для заданої матриці (двовимірного списку) знайти найбільше число, яке він містить.

   Приклад

   matrix =
   [[1, 4, 7],
   [2, 5, 1],
   [10, 2, 2]]

   Вивід: 10

   Джерело: Projekt MmF

8. (***Найдовший вектор***) Для заданого двовимірного списку визначити вектор-рядок із найбільшою довжиною.

Вказівка: написати функцію, яка рахує довжину вектора.

Приклад

matrix =
[[1, 0, 0],
[1, 2, 1],
[2, 1, 1],
[3, 4, 5],
[0, -1, -2]]

Вивід: [3, 4, 5]

Джерело: Projekt MmF

9. (***Сусідні елементи***) Написати функцію, яка для заданої матриці і для її елемента, заданого координатами, повертає список сусідніх значень зверху, знизу, справа і зліва, якщо вони існують.

   Приклад

   matr = [
   [1, 2, 3, 4],
   [5, 6, 7, 8],
   [9, 3, 2, 1]
   ]

   neighbours(matr, [1, 1]) → [2, 7, 3, 5]

   Джерело: *https://www.codewars.com/kata/5b2e5a02a454c82fb9000048*

10. (***Matrix expansion***) Для заданої квадратної матриці побудуємо її розширення за наступними правилами.

    Наприклад, задано матрицю 2 на 2.
    [
    [1, 2],
    [5, 3]
    ]

    Її розширенням назвемо матрицю

    [
    [1, 2, a],
    [5, 3, b]
    [c, d, e]
    ]

де
a = сумі елементів першого рядка
b = сумі елементів другого рядка
c = сумі елеменітів першого стовпчика
d = сумі елементів другого стовпчика
e = сумі діагональних елементів

Написати функцію, яка для заданої квадратної матриці повертає її розширення.

Джерело: *https://www.codewars.com/kata/614adaedbfd3cf00076d47de*

11. (***Транспонування матриці***) Написати функцію *transpose(matrix)*, яка приймає як параметр двовимірний список *matrix* і повертає транспонований список.

    ***Приклади**:*

    matrix =
    [[1, 2],
    [3, 4]]

    transpose(matrix) →
    [[1, 3],
    [2, 4]]

    Джерело: *https://www.codewars.com/kata/559656796d8fb52e17000003*

12. (***Rotate an array matrix***) Написати функцію rotate(matr, dir), яка повертає задану квадратну матрицю на кут 90 градусів за годинниковою стрілкою або проти годинникої стрілки (залежно від параметру dir) навколо її центру.

    Приклад

    m = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
    ]

    rotate(m, “clockwise”) →

    [
    [7, 4, 1],
    [8, 5, 2],
    [9, 6, 3]
    ]

Джерело: *https://www.codewars.com/kata/525a566985a9a47bc8000670*

13. (***Sudoku board validator***) Судоку - це гра, в якій необхідно заповнити таблицю 9 на 9 числами від 1 до 9 за наступними правилами.

    - В кожному рядку і стовпчику числа не можуть повторюватися
    - В кожній з 9 менших таблиць 3 на 3 числа також не можуть повторюватися

    Написати функцію, яка для заданого розв'язання судоку перевіряє, чи воно правильне.

    Джерело: *https://www.codewars.com/kata/63d1bac72de941033dbf87ae*