Цікаві коди Python

**1️⃣ Створення списку з вмісту файлу**

print([line.strip() for line in open("text.txt")])

**2️⃣ Отримання словника зі списку кортежів і списку з назвами колонок**

column\_names = ['name', 'salary', 'job']

db\_rows = [('Alice', 180000, 'data scientist'),

('Bob', 99000, 'mid-level manager'),

('Frank', 87000, 'CEO')]

db = [dict(zip(column\_names, row)) for row in db\_rows]

print(db) # ->

'''

[{'name': 'Alice', 'salary': 180000, 'job': 'data scientist'},

{'name': 'Bob', 'salary': 99000, 'job': 'mid-level manager'},

{'name': 'Frank', 'salary': 87000, 'job': 'CEO'}]

'''

**3️⃣ Обчислення факторіалу**

from functools import reduce

n = 5

print(reduce(lambda x, y: x \* y, range(1, n+1)))

**4️⃣ Алгоритм швидкого сортування**

unsorted = [33, 2, 3, 45, 6, 54, 33]

quick = lambda l: quick([x for x in l[1:] if x <= l[0]]) + [l[0]] + quick([x for x in l if x > l[0]]) if l else []

print(quick(unsorted))

**5️⃣ Сума парних чисел у списку**

nums = [1, 2, 3, 4, 5, 6]

sum\_even = sum([num for num in nums if num%2 == 0])

print(sum\_even)

**6️⃣ Заміна зіпсованих рядків у списку**

visitors = ['Firefox', 'corrupted', 'Chrome', 'corrupted',

'Safari', 'corrupted', 'Safari', 'corrupted',

'Chrome', 'corrupted', 'Firefox', 'corrupted']

visitors[1::2] = visitors[::2]

print(visitors) # ->

'''

['Firefox', 'Firefox', 'Chrome', 'Chrome', 'Safari', 'Safari',

'Safari', 'Safari', 'Chrome', 'Chrome', 'Firefox', 'Firefox']

'''

Швидше за все, ви добре знайомі з функціями strip() , lstrip() і rstrip() в Python. Вони використовуються для видалення пробілів або інших символів з рядків. В основному це виглядає так:

" Andrew Wegner ".lower().strip() # -> 'andrew wegner'

" Andrew Wegner ".lower().lstrip() # -> 'andrew wegner'

" Andrew Wegner ".lower().rstrip() # -> ' andrew wegner'

Але яким буде висновок?

"Andrew Wegner".lower().rstrip("wegner")

Якщо ви очікували '**andrew**' , то ви не маєте рації. Вірний висновок '**and**'

❓ Чому? Справа в тому, що функції працюють зі списком символів , а не рядками. Тобто видаляють з початкового рядка всі символи, які ви передали функції.

При присвоюванні змінній значення іншої змінної у більшості випадків новий об'єкт не створюється, а копіюється посилання на вже існуючий.

import copy

a = [1, 2, 3]

b = [4, 5, 6]

c = [a, b]

# Звичайне присвоєння

d = c

print(id(c) == id(d)) # True — d is the same object as c

print(id(c[0]) == id(d[0])) # True — d[0] is the same object as c[0]

# Поверхневе копіювання

d = copy.copy(c)

print(id(c) == id(d)) # False — d is now a new object

print(id(c[0]) == id(d[0])) # True — d[0] is the same object as c[0]

# Глибоке копіювання

d = copy.deepcopy(c)

print(id(c) == id(d)) # False — d is now a new object

print(id(c[0]) = id(d[0])) # False — d[0] is now a new object

Якщо використовувати функцію copy зі стандартної бібліотеки, новий об'єкт буде створено, але його посилання на інші об'єкти залишаться такими ж.

У випадку з deepcopy відбудеться рекурсивне копіювання. Наприклад, при такому копіюванні списку всі його елементи також копіюються як нові об'єкти.

***Python-скрипт для озвучування PDF***

# Importing necessary libraries

import PyPDF2

import pyttsx3

# Prompt user for the PDF file name

pdf\_filename = input("Enter the PDF file name (including extension): ").strip()

# Open the PDF file

try:

with open(pdf\_filename, 'rb') as pdf\_file:

# Create a PdfFileReader object

pdf\_reader = PyPDF2.PdfReader(pdf\_file)

# Get an engine instance for the speech synthesis

speak = pyttsx3.init()

# Iterate through each page and read the text

for page\_num in range(len(pdf\_reader.pages)):

page = pdf\_reader.pages[page\_num]

text = page.extract\_text()

if text:

speak.say(text)

speak.runAndWait()

# Stop the speech engine

speak.stop()

print("Audiobook creation completed.")

except FileNotFoundError:

print("The specified file was not found.")

except Exception as e:

print(f"An error occurred: {e}")