**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
імені ІГОРЯ СІКОРСЬКОГО»**

**Навчально-науковий інститут прикладного системного аналізу**

**Захист практикуму №1   
з дисципліни «Інтелектуальний аналіз даних»**

Виконали:  
студенти 3 курсу, групи КІ-32

Бондарєва Анастасія,

Власенко Віолетта,

Монагарова Анна,  
Нємцов Олександр

Прийняв:

Андросов Дмитро

Київ – 2025

**Варіант №6**

**Індивідуальне завдання:** Операцiя reduce. Навести приклади.

**Теоретичні відомості**

reduce — це функція з модуля functools у Python, яка послідовно застосовує задану двомісну функцію до елементів ітерованого об’єкта, «згортаючи» послідовність до єдиного результату.

Синтаксис функції reduce() у Python:

from functools import reduce

reduce(function, iterable[, initializer])

де:

* function: функція, що приймає два аргументи, виконує над ними певну операцію та повертає результат.
* iterable: ітерований об’єкт (наприклад, список, кортеж), елементи якого будуть оброблені.
* initializer (необов’язковий): початкове значення. Якщо воно вказане, то передається як перший аргумент function під час першої ітерації. Якщо initializer не вказаний, то першими двома аргументами function стають перші два елементи послідовності iterable.

Принцип роботи reduce. Є послідовність значень (наприклад, список) та функція, яка бере два аргументи та повертає один результат. reduce бере перші два елементи послідовності та застосовує до них цю функцію. Отриманий результат стає накопиченим значенням, яке потім знову обчислюється з наступним елементом. Процес триває, доки не залишиться один кінцевий результат. Якщо вказано initializer, то він використовується як стартове значення перед початком.

**Приклади використання**

Приклад 1. Визначення найбільшого числа в списку.

from functools import reduce

numbers = [7, 2, 9, 4, 6]

# функція порівнює два числа і повертає більше

result = reduce(lambda x, y: x if x > y else y, numbers)

print(result)  # 9

Спочатку порівнюємо 7 і 2, отримуємо 7. Далі 7 і 9 — отримуємо 9. Потім 9 і 4, маємо 9. І 9 та 6 — все ще 9. У підсумку весь список «згорнувся» в одне число 9.

Приклад 2. Конкатенація рядків у речення.

from functools import reduce

words = ["I", "like", "data", "mining!"]

sentence = reduce(lambda x, y: x + " " + y, words)

print(sentence)  # I like data mining!

Ми маємо список слів ["I", "like", "data", "mining!"]. Функція reduce поступово об’єднує їх у один рядок, додаючи пробіл між словами. У результаті весь список «згорнувся» в одне речення.

Приклад 3. Підрахунок загальної вартості товарів у кошику.

from functools import reduce

cart = [

    {"name": "яблука", "price": 15, "qty": 2},   # 15 \* 2 = 30

    {"name": "хліб", "price": 25, "qty": 1},    # 25 \* 1 = 25

    {"name": "молоко", "price": 30, "qty": 3}   # 30 \* 3 = 90

]

total = reduce(lambda acc, item: acc + item["price"] \* item["qty"], cart, 0)

print("Total cost:", total)  # 145

У нас є список словників, де кожен елемент — товар із ціною та кількістю. Ми хочемо знайти підсумкову вартість усіх покупок. initializer = 0 (стартова сума). До 0 додаємо 15 \* 2 = 30, отримуємо 30. До 30 додаємо 25 \* 1 = 25, отримуємо 55. До 55 додаємо 30 \* 3 = 90, отримуємо 145. У результаті — вся вартість кошика дорівнює 145.

Приклад 4. Групування людей за віком.

people = [

    {"name": "Oleh", "age": 15},

    {"name": "Maria", "age": 22},

    {"name": "Ivan", "age": 67},

    {"name": "Anna", "age": 34},

    {"name": "Natalia", "age": 70},

]

def group\_people(acc, person):

    age = person["age"]

    if age < 18:

        acc["child"].append(person["name"])

    elif age < 65:

        acc["adult"].append(person["name"])

    else:

        acc["senior"].append(person["name"])

    return acc

groups = reduce(group\_people, people, {"child": [], "adult": [], "senior": []})

print(groups)

Результат:

{'child': ['Oleh'], 'adult': ['Maria', 'Anna'], 'senior': ['Ivan', 'Natalia']}

У нас є список людей, і ми хочемо згрупувати їх за категоріями: child (до 18), adult (18-64), senior (65+). Початкове значення (initializer) — це словник з трьома порожніми списками. Кожна людина додається у відповідний список залежно від віку. reduce поступово наповнює категорії.