Державний торговельно-економічний університет  
Кафедра інженерії програмного забезпечення та кібербезпеки

**ЛАБОРАТОРНА РОБОТА №2**

**З ДИСЦИПЛІНИ «АЛГОРИТМИ ТА СТРУКТУРИ ДАНИХ»**

**НА ТЕМУ «ОСНОВИ СТРУКТУРИ ДАНИХ МОВИ ПРОГРАМУВАННЯ PYTHON»**

**Виконала**: студентка факультету

інформаційних технологій

групи\_курсу 3-4

Авєріна Наталія Ігорівна

**Перевірила**: Палагута К. О.

Київ 2024

**Лабораторна робота №2**

**Тема**: Основи структури даних мови програмування Python.

**Мета**: Закріпити вміння використовувати основні оператори мови Python, навчитись працювати з рядками, списками, словниками.

**Програмне забезпечення**: PyCharm, MS Word.

**1 варіант**

**Хід роботи:**

**1 задача:**

Створити програму Python, під час виконання якої:

* вводиться довільне ціле число Х в діапазоні від 10 до 20 (коректність даних, що вводяться, слід контролювати);
* створюється список з кількістю елементів Х, значення кожного елемента визначаються випадковим чином;
* визначається сума всіх парних елементів списку;
* виводяться на друк список і сума парних елементів списку.

1. Користувачеві пропонується ввести ціле число в діапазоні від 10 до 20. Для цього було використано конструкцію *while True*, щоб створити нескінченний цикл, який буде повторювати запит на введення, доки користувач не введе коректні дані.   
   Використовується функція *input* для отримання введеного тексту від користувача.  
   Щоб перехопити помилки, які виникають при некоректному введенні, наприклад, якщо користувач вводить символи замість чисел, застосовано конструкцію *try-except.*

Умова *if 10 <= x <= 20* перевіряє, чи введене число знаходиться в потрібному діапазоні. Якщо так, цикл *while* завершується і виконання програми продовжується, в іншому випадку виводиться повідомлення про некоректне введення.

1. Після введення коректного числа генерується список випадкових чисел заданої довжини. Для цього використовується модуль *random* для генерації випадкових чисел. Цикл *for* виконується x разів, де x - введене користувачем число. На кожній ітерації генерується випадкове число в діапазоні від 1 до 100, яке додається до списку *random\_list*.
2. Після створення списку випадкових чисел обчислюється сума всіх парних чисел у цьому списку. Цикл *for* перебирає всі елементи списку *random\_list*. За допомогою умови *if* перевіряється, чи є поточний елемент парним числом. Якщо так, його значення додається до змінної *sum*.
3. На екран виводиться згенерований список і сума парних чисел. Вивід результатів здійснюється за допомогою функції *print*.

import random  
while True:  
 try:  
 x = int(input("Enter an integer from 10 to 20: "))  
 if 10 <= x <= 20:  
 break  
 else:  
 print("Invalid input. Try again.")  
 except ValueError:  
 print("Invalid data type. Try again.")  
  
random\_list = []  
for i in range(0, x):  
 r = random.randint(1, 100)  
 random\_list.append(r)  
print("The random list:", random\_list)  
  
sum = 0  
for i in range(0, len(random\_list)):  
 if random\_list[i] % 2 == 0:  
 sum += random\_list[i]  
  
print("The sum of the even numbers is", sum)

**2 задача:**

Створити програму, під час виконання якої:

* вводиться текст довільної довжини;
* формується словник з кількістю повторень кожного слова (ключ – слово, значення – кількість повторень);
* визначаються 3 слова, які найчастіше повторюються;
* виводяться на друк словник і 3 найбільш вживаних слова.

1. Користувачеві пропонується ввести текст. Для цього використовується функція *input* для отримання введеного тексту від користувача. Введений текст розділяється на слова за допомогою методу *split()* і зберігається в змінній words. Всі слова перетворюються на нижній регістр за допомогою методу *lower(),* щоб уникнути випадкових відмінностей у великих і малих літерах.
2. Для кожного слова в тексті обчислюється кількість його повторень, а потім ця інформація заноситься у словник. Для цього створюється пустий словник dict, де ключі - слова з тексту, а значення - кількість повторень кожного слова. За допомогою циклу *for* перебираються всі слова у списку words. Для кожного слова перевіряється, чи вже воно є ключем у словнику dict. Якщо так, кількість його повторень збільшується на 1, якщо ні - воно додається до словника з початковим значенням 1.
3. Словник сортується за кількістю повторень кожного слова. Функція *sorted* використовується для сортування словника *dict* за значеннями (кількістю повторень) за допомогою аргументу *key=lambda item*: *item[1].* Цей аргумент дозволяє вказати, що сортування має відбуватися за значенням, а не за ключем. Результат сортування зберігається у змінній sorted\_dict.
4. На екран виводиться словник та три найбільш вживані слова. Якщо кількість унікальних слів у тексті менше трьох, то i приймає значення довжини словника sorted\_dict, інакше i приймає значення 3. Потім виводяться три найбільш вживані слова, використовуючи зріз останніх трьох елементів відсортованого словника.

text = input("Enter a text:")  
words = text.lower().split()  
  
dict = {}  
for word in words:  
 if word in dict:  
 dict[word] += 1  
 else:  
 dict[word] = 1  
  
sorted\_dict = sorted(dict.items(), key=lambda item: item[1])  
  
print("Dictionary:", dict)  
if len(sorted\_dict) >= 3:  
 i = 3  
else:  
 i = len(sorted\_dict)  
  
print("Most common words:")  
for word, count in sorted\_dict[-i:]:  
 print(word)

**Висновок**: під час виконання даної лабораторної роботи було закріплено знання операторів мови Python та було набуто навичок в роботі з рядками, списками, словниками.