Лабораторна робота №11

Тема: Робота з рядками. Використання функцій та методів.

Приклад 1. Вивести на екран всі великі літери латинського алфавіту.

Розв'язок. Скористаємося тим фактом, що латинські літери в таблиці кодування розташовані у алфавітному порядку. Скористаємося циклом по проміжку від коду символа *"𝒜"* до коду символа *"𝒵"*:

```
for i in range(ord("A"), ord("Z") + 1):
    print(chr(i), end = '')
```

Нагадаємо, що параметр end в інструкції print вказує яким символом завершувати інструкцію виведення на екран. Типовим значенням є символ '\n' – символ нового рядка. У цій програмі ми використали порожній символ, для того, щоб символи алфавіту були виведені в одному рядку.

Результат виконання програми

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Приклад 2. Визначити, чи є даний символ латинською літерою (великою або маленькою), цифрою або ні тим ні іншим. Розв'язок. Зауважимо, що цю задачу легко розв'язати, використовуючи стандартні функції Python. Проте для глибшого розуміння роботи з символами напишемо програму без їхнього використання. Скористаємося тим фактом, що літери у таблиці кодування розташовані в алфавітному порядку, а цифри – в порядку зростання.

```
Ch = input("Задайте символ ")
if Ch >= 'a' and Ch <= 'z':
    print("%c - маленька літера" % Ch)
elif Ch >= 'A' and Ch <= 'Z':
    print("%c - велика літера" % Ch)
elif Ch >= '0' and Ch <= '9':
    print("%c - цифра" % Ch)
else:
    print("%c - ні літера, ні цифра" % Ch)
```

Приклад 3. Написати програму, що переводить маленькі латинські літери у відповідні великі літери (у верхній регістр). Розв'язок. Аналогічно до попередньої задачі, розв'яжемо поточну задачу, не використовуючи готові функції Python. Аналогічно до попереднього прикладу скористаємося властивостями таблиці

кодування. Помітимо, що різниця кодів між символами маленької і великої літери однакова і становить ord('A') – ord('a')

Тоді програма має такий вигляд:

```
Ch = input("Задайте символ ")
if Ch >= 'a' and Ch <= 'z':
    UpperCh = chr(ord(Ch) + ord('A') - ord('a'))
    print("%c -> %c" % (Ch, UpperCh))
else:
    print("Введений символ не є латинською літерою")
```

Приклад 4. Перевірити, чи є рядок симетричним

Розв'язок. Рядок є симетричним якщо його запис справа на ліво збігається з записом з ліва на право. Для розв’язання цієї задачі використаємо те, що до рядка можна будувати зріз (з від’ємним кроком). Отже, отримаємо таку просту програму.

```
S = input("Введіть рядок ")

if S == S[::-1]:
    print ("Заданий рядок є симетричним")
else:
    print ("Заданий рядок НЕ є симетричним")
```

Приклад 5. Обчислити кількість входжень символа a у рядок S.

Розв'язок. Для розв’язання цієї задачі проітеруємо рядок по-символьно за допомогою циклу по колекції. Використаємо лічильник, який будемо збільшувати на один кожного разу коли будемо зустрічати символ a. Отже програма матиме вигляд

```
S = input("Введіть рядок ")
a = input("Введіть символ ")

n = 0              # Змінна лічильник
for ch in S:       # Змінна ch пробігає всі символи рядка S
    if ch == a:    # Якщо ch == a
        n += 1     # збільшуємо лічильник на одиницю

print("\"%s\" входить у рядок \"%s\" %d разів" % (a, S, n))
```

Приклад 6. Перевірити, чи правильно в заданому виразі розставлені круглі дужки (тобто, чи стоїть справа від кожної відкритої дужки відповідна до неї закрита дужка, а зліва від кожної закритої – відповідна до неї відкрита).

Розв'язок. Використаємо змінну k для підрахунку різниці кількості відкритих і закритих дужок. Додатково використаємо змінну r, яка буде набувати значення False, якщо закритій дужці не відповідає жодна із відкритих дужок, що їй передують. Наприклад, якщо текст має вигляд

```
"a * ( b + c ) ) * ( d"
```

то, r = False, хоча при цьому k = 0, бо кількість відкритих і закритих дужок однакова.

```
expresion = input("Введіть вираз ")
k = 0    # різниця кількості відкритих і закритих дужок
r = True # закритій дужці відповідає відкрита
for c in expresion:
    if c == '(':
        k += 1
    if c == ')':
        k -= 1
    if k < 0: # закритій дужці не відповідає відкрита
        r = False
        break

if (k == 0 and r):
    print("Дужки розставлені правильно")
else:
    print("Дужки розставлені неправильно")
```

Приклад 7. У заданий текст входять тільки цифри та літери. Визначити, чи правда, що сума числових значень цифр, які входять у текст, дорівнює довжині тексту.

Розв'язок. Використаємо змінну m для підрахунку суми цифр, що містяться у заданому тексті.

```
S = input("Введіть вираз ")
m = 0
for c in S:
    if c >= '0' and c <= '9':
        m = m + int(c) # тут використовується операція int(c)
                       # перетворення символа c у число

if len(S) == m:
    print("Так")
else:
    print("Ні")
```

Задачі для самостійного виконання

Завдання 1. Створити програму перетворення рядка, замінивши в ньому кожну крапку трьома крапками.

Завдання 2. Надрукувати лише великі латинські літери, що входять до заданого рядка.

Завдання 3. Нехай у рядку міститься послідовність слів, що розділена одним чи декількома символами пропуску. Необхідно видалити зайві пропуски між словами, так, щоб слова розділялися лише одним символом пропуску. Також видалити всі пропуски на початку та вкінці рядка. Розв'язок. Найпростіший спосіб розв’язання цієї задачі – розбили рядок на слова, а далі склеїти отриманий список за допомогою функції join застосувавши її до символу пропуска у якості розділювача.

Завдання 4. У заданому рядку всі символи '0' замінити на '1', а символи '1' на '0'. Розв’язок. У цій задачі ми не можемо скористатися операцією replace, Якщо спочатку замінити символи '0' на '1', то рядок не буде містити нулів і відповідно, коли потім навпаки замінити '1' на '0', то рядок не буде містити одиниць. Отже, потрібно написати програму, що реалізує поставлену задачу здійснюючи заміну для кожного окремого символу рядка. Нагадаємо, рядок належить до незмінюваних типів, тому ми не можемо безпосередньо змінювати символи у рядку. Застосуємо підхід, а саме перетворимо рядок у список символів, проведемо необхідну заміну у отриманому списку символів і склеїмо символи до купи використовуючи операцію join. Заміну будемо здійснювати таким чином: будемо пробігати список символів і якщо поточний символ буде '0' або '1', то будемо змінювати його на '1' або '0' відповідно.

ПОРІВНЯННЯ РЯДКІВ (законспектувати)

Рядки у Python можна порівнювати. Порівняння рядків проводиться лексикографічно.

Будемо казати, що послідовність $a = \{a1a2...an\}$ лексикографічно менша (або просто менша) за послідовність $b = \{b1b2...bn\}$, якщо існує таке натуральне k, що для всіх $i \leq k$ виконується рівність $ai = bi$, а для $i = k + 1$ справедлива нерівність $ak+1 < bk+1$. Якщо для всіх $i \leq n$ виконується рівність $ai = bi$, то будемо казати, що послідовності a і b лексикографічно рівні (або просто рівні).

Наприклад, послідовність {012} менша за послідовність{021}. Також якщо згадати, що символи можна порівнювати, порівнюючи їхні номери в алфавіті, то послідовність символів $\{aaa\}$ менша за послідовність $\{aba\}$. Вищенаведене означення можна узагальнити на випадок послідовностей, у які входить неоднакова кількість членів. Для цього будемо вважати, що порожній член послідовності (тобто

його відсутність) менший за будь-який наявний. Отже, з цього можемо зробити висновок, що, наприклад, послідовність символів $\{aaa\}$ менша за послідовність символів $\{abcd\}$.

Лексикографічний порядок послідовностей – це порядок, при якому послідовності відсортовані за (лексикографічним) зростанням.

Прикладом лексикографічного порядку є послідовність слів у словнику. Для порівняння рядків, у Python як і для символів визначені 6 стандартних відношень ==, !=, >, <, >=, <=

Приклад 8. Перевірити чи слова, що записані у рядку знаходяться у словниковому порядку.

Розв'язок. Для розв'язання цієї задачі достатньо розбити рядок на слова і далі, скористатися операцією порівняння рядків для слів отриманого списку. Цю задачу можна віднести до задач пошуку. Дійсно, припустимо спочатку, що слова у рядку розташовані у порядку зростання. Далі будемо шукати послідовну пару слів, таку, що перше слово більше за друге. Якщо таку пару ми знайдемо, то слова у рядку не впорядковані. Отже програма буде мати вигляд

```
S = input("Введіть рядок ")
l = S.split()
ok = True
for i in range(1, len(l)):
    if l[i - 1] > l[i]:
        ok = False
        break

if ok:
    print("Слова в алфавітному порядку")
else:
    print("Слова НЕ в алфавітному порядку")
```

Задачі для самостійного виконання

Завдання 8. Вивести на екран таку таблицю символів: Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx

Завдання 9. Визначити, яка з двох заданих літер у даному тексті трапляється частіше.

Завдання 10. Визначити, чи входить до даного тексту

a) кожна з літер слова "key";
b) кожна з літер заданого слова.

Завдання 11. Скласти програму підрахунку загального числа входжень символів основних арифметичних операцій ('+', '-', '*', '/') у рядку.