Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Програмування

Лабораторна робота №3

«Робота з даними типу str, bytes та bytearray».

Виконав:
Студент групи ІО-15
Кушнерик Є. О.
Залікова книжка №1508

Перевірив:
Пономаренко А. М.

Київ – 2021

Лабораторна робота №3

Тема: «Робота з даними типу str, bytes та bytearray».

Мета: вивчити способи створення рядків та даних типу bytes і bytearray, операції над ними. Форматування рядків. Функції та методи роботи з рядками.
Налаштування локалі.

Завдання:

1. Вивчити матеріал лекцій 7, 8, 9 та 10.
2. Виконати індивідуальне завдання лабораторної роботи, вибране відповідно до варіанту.

Теоретичні основи:

Створити рядок можна такими способами:

1.За допомогою функції `str ( [ <Об'єкт> [, <Кодування> [, <Обробка помилок>]]])`

```
>>> str(), str([1, 2])
```

2. Указавши рядок між апострофами або подвійними лапками

```
>>> print ('рядок1\nрядок2')
```

3. Указавши рядок між потроєними апострофами або потроєними лапками.

```
>>> print('''Рядок1
Рядок2''')
```

Перелік спеціальних символів, припустимих всередині рядка, перед яким немає модифікатора `r`:

`\n` - перевід рядка;
`\r` - повернення каретки;
`\t` - знак табуляції;
`\v` - вертикальна табуляція;
`\a` - дзвінок;
`\b` - вибій;
`\f` - перевід формату;
`\0` - нульовий символ (не є кінцем рядка);
`\"` - лапки;
`\'` - апостроф;
`\N` - вісімкове значення N. Наприклад, `\74` відповідає символу <;
`\xn` - шістнадцяткове значення N. Наприклад, `\x6a` відповідає символу `j`;

\\ - зворотний (обернений) слеш;

\uxxxx - 16-бітний символ Unicode. Наприклад, \u043a відповідає російській букві к;

\Uxxxxxxxx–32–бітний символ Unicode

Створити об'єкт типу bytes *можна у декілька способів.*

Спосіб 1. За допомогою функції

bytes([<Рядок>,<Кодування>[,<Обробка помилок>]])

Спосіб 2. За допомогою методу рядків encode([encoding="utf-8"][, errors="strict"]).

Спосіб 3.

Указавши букву b (регістр не має значення) перед рядком в апострофах, лапках, потрійних апострофах або потрійних лапках.

Спосіб 4.

За допомогою функції bytes(<Послідовність>), яка перетворює послідовність цілих чисел від 0 до 255 в об'єкт типу bytes.

Спосіб 5

За допомогою функції bytes(<Число>), яка задає кількість елементів у послідовності. Кожний елемент буде містити нульовий символ Спосіб 6

За допомогою методу bytes.fromhex (<Рядок>).

Перетворити об'єкт типу bytes у рядок дозволяє метод decode(). decode([encoding="utf-8"][,errors="strict"]) Для перетворення можна також скористатися функцією str():

```
>>> b = bytes("рядок", "cp1251")
>>> str(b, "cp1251")
'рядок'
```

Створити об'єкт типу bytearray *можна такими способами*: Спосіб 1. За допомогою функції bytearray([<Рядок>,<Кодування>[,<Обробка помилок>]])

Спосіб 2

За допомогою функції bytearray (<Послідовність>), яка перетворює послідовність цілих чисел від 0 до 255 в об'єкт типу bytearray.

Спосіб 3

За допомогою функції bytearray(<Число>), яка задає кількість елементів у послідовності.

Спосіб 4

За допомогою методу `bytearray.fromhex(<Рядок>)`.

Операції з рядками

1.Доступ до символа за індексом в квадратних дужках:

```
>>> s = "Python"
>>> s[0]
'P'
```

2. Доступ за від'ємним індексом відраховується з кінця:

```
>>> s = "Python"
>>> s[-1]
'n'
```

3. Змінити рядок по індексу неможливо

4.Операція добування зрізу для зміни рядка

```
[<Початок>:<Кінець>:<Крок>]
>>> s[2:5] # повертаються символи з індексами 2, 3 и 4
'tho'
```

Метод `format()`

```
<Рядок>=<Рядок спеціального формату>.format(*args, **kwargs)
```

У параметрі `<Рядок спеціального формату>` усередині символів фігурних дужок: `{ i }` вказуються специфікатори, що мають наступний синтаксис:

```
{ [<Поле>] [ !<Функція>] [ :<Формат>] }
>>> print("Символи {{i}} - {0}".format("спеціальні"))
Символи {i} - спеціальні
```

У параметрі `<Формат>` вказується значення, що має наступний синтаксис:

```
[[<Заповнювач>] <Вирівнювання>] [<Знак>] [#] [0]
[<Ширина>] [,][.<Точність>][<Перетворення>]
```

За замовчуванням значення усередині поля вирівнюється по правому краю. Управляти вирівнюванням дозволяє параметр `<Вирівнювання>`. Можна вказати наступні значення: <, >, ,=

`<` – по лівому краю;

`>` – по правому краю; `^` – по центру поля.

`=` – знак числа вирівнюється по лівому краю, а число по правому краю `+` - задає обов'язковий вивід знака як для від'ємних, так і для додатних чисел.

Функції

str([<Об'єкт>])– перетворює будь-який об'єкт у рядок.
repr(<Об'єкт>) – повертає строкове представлення об'єкта.
ascii(<Об'єкт>)– повертає строкове представлення об'єкта.
len(<Рядок>)– повертає кількість символів у рядку.

Методи

strip([<Символи>])– видаляє зазначені в параметрі символи на початку й наприкінці рядка.
lstrip([<Символи>])- видаляє «пропускові» або зазначені символи на початку рядка.
rstrip([<Символи>]) – видаляє «пропускові» або зазначені символи наприкінці рядка.
split ([<Роздільник>[,<Ліміт>]])– розділяє рядок на підрядки по зазначеному роздільнику й додає ці підрядки в список, який повертається як результат.
rsplit([<Роздільник>[,<Ліміт>]])– аналогічний методу split(), але пошук символу-роздільника проводиться не зліва направо, а, навпаки, справа наліво.
partition(<Роздільник>)– знаходить перше входження символу-роздільника в рядок і повертає кортеж із трьох елементів: перший елемент буде містити фрагмент, розташований перед роздільником, другий елемент – сам роздільник, а третій елемент – фрагмент, розташований після роздільника. Пошук проводиться зліва направо.
rpartition (<Роздільник>) – метод аналогічний методу partition(), але пошук символу-роздільника проводиться не зліва направо, а, навпаки, справа наліво.
join() – перетворить послідовність у рядок. Елементи додаються через вказаний роздільник. Формат методу:

```
<Рядок>=<Роздільник>.join(<Послідовність
>)
```

Пошук і заміна в рядку

find()– шукає підрядок в рядку.

```
<Рядок>.find(<Підрядок>[,<Початок>[,<Кінець>)))
```

Якщо початкова позиція не зазначена, то пошук буде здійснюватися з початку рядка. Якщо параметри <Початок> і <Кінець> зазначені, то проводиться операція добування зрізу:

```
<Рядок>[<Початок>:<Кінець>]
```

і пошук підрядка буде виконуватися в цьому фрагменті.

index() - метод аналогічний методу find(), але якщо підрядок в рядок не входить, то виконується виключення Valueerror.

```
<Рядок>.index(<Подстрока>[,<Початок>[,<Кінець>]])
```

rfind() – шукає підрядок в рядку.

```
<Рядок>.rfind(<Подстрока>[,<Початок>[,<Кінець>]])
```

rindex() – метод аналогічний методу rfind(), але якщо підрядок в рядок не входить, то виконується виключення ValueError.

```
<Рядок>.rindex(<Подстрока>[,<Початок>[,<Кінець>]])
```

count() - повертає число входжень підрядка в рядок.

```
<Рядок>.count(<Підрядок>[,<Початок>[,<Кінець>]])
```

Методи bytearray

append (<Число>)– додає один елемент у кінець об'єкта

extend(<Послідовність>) - додає елементи послідовності в кінець об'єкта

+ і += використовувати оператори для додавання декількох елементів Присвоювання значення зрізу:

```
>>> b = bytearray( "string", "ascii")
>>> b[len(b):] = b"123"
# Додаємо елементи в послідовність
>>>b
bytearray(b'string123')
```

insert(<Індекс>, <Число>) – додає один елемент у зазначену позицію.

pop([<Індекс>]) – видаляє елемент, розташований по зазначеному індексу, і повертає його.

Вилучити елемент списку дозволяє також оператор del.

remove(<Число>)– видаляє перший елемент, що містить зазначене значення. reverse () - змінює порядок проходження елементів на протилежний.

decode()- перетворює об'єкт типу bytearray в рядок

```
decode ( [encoding="utf-8"] [,
errors="strict"])
```

Індивідуальні завдання:

Завдання 1

1.Відповідно до номера в списку групи вибрати індивідуальне завдання.

2.Написати програму на мові Python.

3.Забезпечити ввід даних з клавіатури комп'ютера та друк результатів обчислень.

4.У звіті до лабораторної роботи описати алгоритм, за яким побудована програма.

При виводі даних обов'язково використати форматування.!!!!

8	Задати два однакові за довжиною рядки. Побудувати новий рядок, в якому на парних місцях розташовані елементи першого рядка, а на непарних – елементи другого рядка.

Код програми

```
print("Введіть два однакові за довдиною рядка")
first=input("Введіть перший рядок:\n")
second=input("Введіть другий рядок:\n")
while(len(first)!=len(second)):
    print("{}".format("Ви ввели різні за довжиною рядки"))
    first = input("Введіть перший рядок:\n")
    second = input("Введіть другий рядок:\n")
for i in range(len(first)):
    print("{0}{1}".format(first[i],second[i]),end="")
```

Результат програми

```
D:\ProgramData\Anaconda3\envs\pythonProject\python.e
Введіть два однакові за довдиною рядка
Введіть перший рядок:
abcdef
Введіть другий рядок:
abcdefg
Ви ввели різні за довжиною рядки
Введіть перший рядок:
University
Введіть другий рядок:
Motivation
UMnoitvievrastiitoyn
```

Завдання 2

1.Відповідно до номера в списку групи вибрати індивідуальне завдання.

2.Написати програму на мові Python з використанням типів даних `bytes` та `bytearray`.

3.Забезпечити ввід даних з клавіатури комп'ютера та друк результатів обчислень.

4.У звіті до лабораторної роботи описати алгоритм, за яким побудована програма.

При виводі даних обов'язково використати форматування!!!
Всі операції в завданні 2 обов'язково виконувати тільки зі змінними типу bytes або bytearray!!!

8	1.Ввести послідовність символів. 2.Перетворити її, видаливши кожен символ *, і повторивши кожен символ, відмінний від *.

Код програми

```
a_string = input("Enter symbols")
byte_array = bytearray(a_string.replace('*', ''), 'utf-8')
emptystr = ''
str_array = byte_array.decode()
for i in str_array:
    emptystr += i + i
print("{}".format(emptystr))

```

Результат програми

```
D:\ProgramData\Anaconda3\envs\pythonPr
Enter symbols@*#*$*%*
@@##$$%%

Process finished with exit code 0
```

Висновки:

Під час виконання лабораторної роботи я навчилась: оперувати методами та функціями для рядків (які дозволяють значно скоротити код), типів даних bytes, bytesarray та використовувати форматування тексту при виведенні даних. Написані мною програми працюють коректно і повністю виконують поставлену в них задачу.