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'
```
Форматування рядків
```
<Рядок спеціального формату> % <Значення>
```
Всередині параметра `<Рядок спеціального формату>` можуть бути зазначені специфікатори, що мають наступний синтаксис:
```
% [(<Ключ>)][<Прапор>][<Ширина>][.<Точність>]<Тип перетворення>
```
Для форматування рядків можна також використовувати наступні методи:

`expandtabs ([<Ширина поля>])` – заміняє символ табуляції пробілами таким чином,щоб загальна ширина фрагмента разом з текстом, розташованим перед символом табуляції, дорівнювала зазначеній величині.

`center ( <Ширина> [, <Символ>])` – виконує вирівнювання рядка по центру всередині поля зазначеної ширини.

`ljust ( <Ширина> [, <Символ>] )` – виконує вирівнювання рядка по лівому краю всередині поля зазначеної ширини.

`rjust(<Ширина>[, <Символ>])` – виконує вирівнювання рядка по правому краю усередині поля зазначеної ширини.

`zfill (<Ширина>)` – виконує вирівнювання фрагмента по правому краю усередині поля зазначеної ширини.

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 в 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.

splitlines([True]) – розділяє рядок на підрядки по символу переводу рядка (\n) і додає їх у список.

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>)`

Налаштування локалі

Для установки локалі (сукупності локальних налаштувань системи) служить функция `setlocale ()` з модуля `locale`. Перш ніж використовувати функцію, необхідно підключити модуль за допомогою виразу:

```
import locale

setlocale(<категорія>[,<Локаль>));
```

Параметр `<категорія>` може набувати наступних значень:

`locale. LC_ALL` - установлює локаль для всіх режимів;
`locale.LC_COLLATE` – для порівняння рядків;
`locale. LC_CTYPE` – для переводу символів у нижній або верхній регістр;
`locale.LC_MONETARY` – для відображення грошових одиниць;
`locale. LC_NUMERIC` – для форматування чисел;
`locale.LC_TIME` – для форматування виводу дати й часу.

Одержати поточне значення локалі дозволяє функція `getlocale([<Категорія>])`.
Одержати налаштування локалі дозволяє функція `localeconv()`.

Зміна регістру символів

`upper()` – заміняє всі символи рядка відповідними прописними буквами.

```
>>> print("рядок".upper())
РЯДОК
```

`lower()` – заміняє всі символи рядка відповідними малими літерами
`swapcase()` – заміняє всі малі літери відповідними прописними буквами, а всі прописні літери – малими:
`capitalize()` – робить першу букву рядка прописною
`title ()` – робить першу букву кожного слова прописною
`casefold()` – те ж саме, що й `lower()`, але додатково перетворить усі символі з діакритичними знаками й лігатури в букви стандартної латиниці.

Функції для роботи з символами

`chr(<Код символу>)` – повертає символ по зазначеному коду.
`ord (<Символ>)` - повертає код зазначеного символу.

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>]])`

Перевірка типу вмісту рядка

`isalnum()` – повертає `True`, якщо рядок містить тільки букви й (або) цифри, у протилежному випадку – `False`.
`isalpha()` – повертає `True`, якщо рядок містить тільки букви, а якщо ні, то – `False`.
`isdigit()` – повертає `True`, якщо рядок містить тільки цифри, а якщо ні, то – `False`
`isdecimal()` – повертає `True`, якщо рядок містить тільки десяткові символи, а якщо ні, то – `False`.
`isnumeric()` – повертає `True`, якщо рядок містить тільки числові символи, а якщо ні, то – `False`.
`isupper()` – повертає `True`, якщо рядок містить букви тільки верхнього регістру, а якщо ні, то – `False`.
`islower()` – повертає `True`, якщо рядок містить букви тільки нижнього регістру, а якщо ні, то – `False`.

istitie () – повертає True, якщо всі слова в рядку починаються із великої літери, а якщо ні, то – False.
isprintable() – повертає True, якщо рядок містить символи, що друкуються, а якщо ні, то – False.
isspace() – повертає True, якщо рядок містить тільки «пропускові» символи, у протилежному випадку – False:
isidentifier()– повертає True, якщо рядок є припустимим з погляду Python ім'ям змінної, функції або класу, а якщо ні, то – False:

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

Відповідно до номера в списку групи вибрати індивідуальне завдання. Написати програму на мові Python . Забезпечити ввід даних з клавіатури комп'ютера та друк результатів обчислень. У звіті до лабораторної роботи описати алгоритм, за яким побудована програма. **При виводі даних обов'язково використати форматування.**

№вар	Завдання
1	У заданому рядку замінити пробіли, які йдуть підряд, на один пробіл.

2	В заданому рядку порахувати кількість слів. Роздільником слів вважається один або кілька пробілів.
3	В заданому рядку замінити кожну кириличну літеру символом «*».
4	В заданому рядку видалити всі латинські літери.
5	Ввести рядок. Дописати в кінець рядка його довжину.
6	У заданому рядку дописати після кожного символу «*» символ «_».
7	У заданому рядку вставити перед кожним символом «!» символ «?»
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.
9	Ввести рядок і замінити кожен знайдений пробіл двома пробілами.
10	Ввести рядок. Побудувати новий рядок, в якому всі символи записано в зворотному порядку.
11	Побудувати рядок, що складається з малих букв латинського алфавіту (розташованих за алфавітом).
12	Побудувати рядок, що складається з великих кириличних літер (розташованих за алфавітом).
13	Ввести рядок і поміняти місцями символи, які стоять поруч.
14	Ввести рядок та окремий символ. Видалити введений символ, якщо він знаходиться в рядку.
15	Ввести рядок. Визначити останній символ в рядку. Видалити всі подібні символи, якщо вони знайдуться в рядку.
16	Ввести рядок. Розташувати всі його символи за зростанням їх кодів.
17	Ввести рядок. Замінити кожен символ «№» рядком «номер по порядку ».
18	Ввести рядок. Перед кожною великою латинською літерою поставити «. » (крапка, пробіл).
19	Розрядити пробілами введений рядок.
20	Ввести рядок. Замінити поєднання символів «-+» символом «0».
21	Ввести рядок. Замінити кожен символ рядка наступним за кодом символом. Букву «я» замінити пробілом.
22	Ввести рядок. Поміняти кожен символ рядка попереднім за таблицею кодування символом.
23	Ввести рядок. Після кожного символу вставити число, що відповідає коду цього символу.
24	Ввести рядок. Отримати з нього два рядки. Перший має складатися з символів введеного рядка, які мають парні і непарні індекси, а другий – з символів з непарними індексами

25	Ввести два рядки однакової довжини. Побудувати новий рядок, який складається з символів як одного, так і іншого рядка, що чергуються між собою.
26	Ввести рядок. У рядку після кожного слова дописати його довжину.
27	Ввести рядок. Отримати передостаннє слово цього рядка. Роздільником слів вважаються один або кілька пробілів.
28	Ввести рядок. Замінити парну кількість пробілів символом «П», а непарну - символом «Н».
29	Ввести рядок. Після кожної кириличної літери поставити її код, а після кожної латинської - символ «_».
30	Ввести рядок. Після кожного слова типу «sin», «cos» або «log», поставити дужки «()».

Завдання 2

Відповідно до номера в списку групи вибрати індивідуальне завдання. Написати програму на мові Python з використанням типів даних `byte` та `bytearray`. Забезпечити ввід даних з клавіатури комп'ютера та друк результатів обчислень. У звіті до лабораторної роботи описати алгоритм, за яким побудована програма. **При виводі даних обов'язково використати форматування.**

№вар	Завдання
1	Ввести послідовність символів, в якому зустрічаються структури `<s>, <#s>`. Замінити кожне входження `<s>` на `<#s>`, а кожне входження `<#s>` на `<#>`. Зауваження: в програмі слід врахувати, що `s` може бути як малим, так і великим.
2	Ввести послідовність символів, що містить число в двійковій системі числення. Перевірити правильність введення цього числа (в його записі повинні бути тільки символи `0` і `1`). Якщо число введено невірно, повторити введення. При правильному введенні перевести число в десяткову систему числення.
3	Ввести послідовність символів, що містить текст. Визначити довжину максимальної серії символів відмінних від букв, яка входить в послідовність.
4	Ввести послідовність символів. Перетворити його, замінивши крапками всі двокрапки (:), що зустрічаються серед першої половини символів, і замінивши крапками всі знаки оклику, які зустрічаються серед символів, що стоять у другій половині початкової послідовності.
5	Ввести послідовність символів. Вказати ті слова, які містять хоча б одну букву k.

6	Ввести послідовність символів, що містить текст. У рядку між словами вставити замість пробілу кому і пробіл.
7	Ввести послідовність символів, що містить текст, який закінчується крапкою. Вивести на екран слова, що містять три букви.
8	Ввести послідовність символів. Перетворити її, видаливши кожен символ * і повторивши кожен символ, відмінний від *.
9	Ввести послідовність символів, що містить текст. Підрахувати кількість букв k в останньому слові послідовності.
10	Ввести послідовність символів. Підрахувати, скільки різних символів зустрічається у ньому. Вивести їх на екран.
11	Ввести послідовність символів. Підрахувати найдовшу послідовність букв a, які йдуть підряд.
12	Ввести послідовність символів, серед яких є дужки, які відкриваються та закриваються. Вивести на екран послідовності символів, що розташовані всередині цих дужок.
13	Ввести послідовність символів, що містить текст. Визначити процентне відношення малих і великих літер до загальної кількості символів в ньому.
14	Ввести послідовність символів, серед яких є одна дужка, яка відкривається і одна дужка, що закривається. Вивести на екран всі символи, розташовані всередині цих дужок.
15	Ввести послідовність символів, що містить букви латинського алфавіту і цифри. Вивести на екран довжину найбільшої послідовності цифр, що йдуть підряд.
16	Ввести послідовність символів з 11 елементів. Прибрати зайві пробіли (Більше одного поспіль).
17	Введіть 5 послідовностей символів довжиною 8 елементів. Розташувати послідовності в алфавітному порядку (як в словнику).
18	Ввести послідовність символів. Визначити символ який найчастіше зустрічається і кількість його повторень.
19	Ввести послідовність символів. Розташувати слова в алфавітному порядку за першою літерою слова.
20	Вивести слова в зворотному порядку, не використовуючи додаткову пам'ять.
21	Ввести послідовність символів, що містить пробіли. Знайдіть найдовше слово, виведіть на екран це слово і його довжину.
22	Ввести послідовність символів, що містить дві одинакові літери. Визначте ці літери.
23	Введіть список символів з 7 елементів. Визначте, чи є він

	симетричним (Симетричним вважається список, який однаково читається зліва направо і справа наліво).
24	Ввести дві послідовності символів. Визначте однакові символи, які містять обидва списки і їх кількість.
25	Введіть послідовність символів з 17 елементів. Визначте символ з найбільшим числом повторень.
26	Введіть послідовність символів з 11 елементів. Обчислити добуток цілих чисел, які входять в нього (без урахування їх знаків).
27	Введіть послідовність символів. Вивести всі слова, що містить послідовність, як окремі списки символів (без пробілів)
28	Введіть послідовність символів та ціле число. Вивести слова з кількістю символів, яка відповідає введеному числу.
29	Введіть послідовність символів. Вивести всі слова, які починаються з даної літери.
30	Введіть послідовність символів і окреме слово. Визначити, скільки разів задане слово зустрічається в даній послідовності.

Зміст звіту:

1. Титульний лист повинен мати такий вигляд:

Національний технічний університет України
«Київський політехнічний інститут»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Програмування
Лабораторна робота №3
«Робота з даними типу str, bytes та bytearray ».

Виконав:
студент групи ІО-ХХ
Іванов І. І.
Залікова книжка № _____
Перевірив Новотарський М.А.

Київ 2016р.

2. Мета лабораторної роботи та загальне завдання
3. Короткі теоретичні відомості, які відображають операції, функції та методи обробки символьних даних, які були використані при написанні лабораторної роботи
4. Роздруківка того фрагменту тексту програми, який написаний індивідуально.
5. Роздруківка результатів виконання програми з контрольним прикладом.
6. Аналіз результатів та висновки.

Контрольні питання (в звіті не відображаються)

1. Способи створення рядків.
2. Спеціальні символи, що застосовуються при обробці рядків.
3. Операції над рядками.
4. Математичні функції модуля math
5. Оператори розгалуження та циклу.
6. Форматування рядків за допомогою оператора %
7. Метод форматування format()
8. Функції та методи для перетворень символьних типів даних