**Лабораторна робота №3**

**Тема. Словники та робота з ними. Методи словників. Множина (set и frozenset). Функції та їх аргументи.**

**Мета роботи**. Отримати навички роботи з словниками, множинами, функціями

**Зміст**.

1. Вивчення відомостей про роботу з словниками, множинами, функціями.

2. Виконання роботи.

3. Отримання результату.

**Ключові положення.**

Словники в Python - невпорядковані колекції довільних об'єктів з доступом по ключу. Їх іноді ще називають асоціативними масивами або хеш-таблицями. Щоб працювати зі словником, його потрібно створити. Створити його можна кількома способами.

1. **По-перше**, за допомогою литерала:

Наприклад:

>>> d = {}  
>>> d  
{}  
>>> d = {'dict': 1, 'dictionary': 2}  
>>> d  
{'dict': 1, 'dictionary': 2}

1. **По-друге**, за допомогою функції **dict**:

Наприклад:

>>> d = dict(short='dict', long='dictionary')  
>>> d  
{'short': 'dict', 'long': 'dictionary'}  
>>> d = dict([(1, 1), (2, 4)])  
>>> d  
{1: 1, 2: 4}

1. **По-третє**, за допомогою методу fromkeys:

Наприклад:

>>> d = dict.fromkeys(['a', 'b'])  
>>> d  
{'a': None, 'b': None}  
>>> d = dict.fromkeys(['a', 'b'], 100)  
>>> d  
{'a': 100, 'b': 100}

1. **По-четверте**, за допомогою генераторів словників, які дуже схожі на генератори списків.

Наприклад:

>>> d = {a: a \*\* 2 for a in range(7)}  
>>> d  
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36}

**Методи словників**

dict.clear () - очищає словник.

dict.copy () - повертає копію словника.

classmethod dict.fromkeys (seq [, value]) - створює словник з ключами з seq і значенням value (за замовчуванням None).

dict.get (key [, default]) - повертає значення ключа, але якщо його немає, не кидає виняток, а повертає default (за замовчуванням None).

dict.items () - повертає пари (ключ, значення).

dict.keys () - повертає ключі в словнику.

dict.pop (key [, default]) - видаляє ключ і повертає значення. Якщо ключа немає, повертає default (за замовчуванням кидає виняток).

dict.popitem () - видаляє і повертає пару (ключ, значення). Якщо словник порожній, кидає виняток KeyError. Пам'ятайте, що словники невпорядковані.

dict.setdefault (key [, default]) - повертає значення ключа, але якщо його немає, не кидає виняток, а створює ключ з значенням default (за замовчуванням None).

dict.update ([other]) - оновлює словник, додаючи пари (ключ, значення) з other. Існуючі ключі перезаписуються. Повертає None (не новий словник!).

dict.values ​​() - повертає значення у словнику.

**Що ж можна ще робити зі словниками?**

Так то ж саме, що і з іншими об'єктами : вбудовані функції, ключові слова (наприклад, цикли for і while), а також спеціальні методи словників.

**Що таке множина?**

Множина в python - "контейнер", що містить елементи у випадковому порядку які не повторюються.

Створюємо множину:

Наприклад:

>>> a = set('hello')  
>>> a  
{'h', 'o', 'l', 'e'}  
>>> a = {'a', 'b', 'c', 'd'}  
>>> a  
{'b', 'c', 'a', 'd'}

Множину використовують для видалення повторюваних елементів.

Наприклад:

>>> words = ['hello', 'daddy', 'hello', 'mum']  
>>> set(words)  
{'hello', 'daddy', 'mum'}

З множинами можна виконувати безліч операцій: знаходити об'єднання, перетин ...

• len (s) - число елементів у множині (розмір множини).

• x in s - чи належить x множині s.

• set.isdisjoint (other) - істина, якщо set і other не мають спільних елементів.

• set == other - все елементи set належать other, все елементи other належать

set.

• set.issubset (other) або set <= other - все елементи set належать other.

• set.issuperset (other) або set> = other - аналогічно.

• set.union (other, ...) або set | other | ... - об'єднання декількох множин.

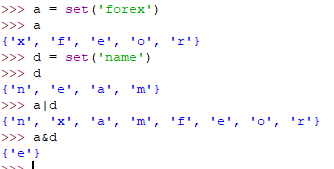
• set.intersection (other, ...) або set & other & ... - перетин.

• set.difference (other, ...) або set - other - ... - безліч з усіх елементів set, які не належать жодному з other.

• set.symmetric\_difference (other); set ^ other - безліч з елементів, що зустрічаються в одному безлічі, але не зустрічаються в обох.

• set.copy () - копія безлічі.

Наприклад:



І операції, безпосередньо змінюють множини:

• set.update (other, ...); set | = other | ... - об'єднання.

• set.intersection\_update (other, ...); set & = other & ... - перетин.

• set.difference\_update (other, ...); set - = other | ... - віднімання.

• set.symmetric\_difference\_update (other); set ^ = other - безліч з елементів, що зустрічаються в одному безлічі, але не зустрічаються в обох.

• set.add (elem) - додає елемент в множину.

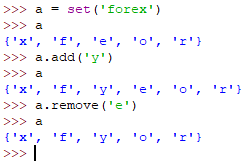
• set.remove (elem) - видаляє елемент з безлічі. KeyError, якщо такого елемента не існує.

• set.discard (elem) - видаляє елемент, якщо він знаходиться в множині.

• set.pop () - видаляє перший елемент з множини. Так як множина не впорядковані, не можна точно сказати, який елемент буде першим.

• set.clear () - очищення множини.

Наприклад:



Єдина відмінність **set** від **frozenset** полягає в тому, що set - змінюваний тип даних, а frozenset - ні.

Наприклад:

>>> a = set('qwerty')  
>>> b = frozenset('qwerty')  
>>> a == b  
True  
>>> True  
True  
>>> type(a - b)  
<class 'set'>  
>>> type(a | b)  
<class 'set'>  
>>> a.add(1)  
>>> b.add(1)  
Traceback (most recent call last):  
File "<stdin>", line 1, in <module>  
AttributeError: 'frozenset' object has no attribute 'add'

Підпрограма - це іменований фрагмент програми, до якого можна звернутися з іншого місця програми. Підпрограми діляться на дві категорії: процедури і функції.

**Процедури. Cинтаксис процедури**

**def ім'я процедури (Відомості про опції):**

**система команд**

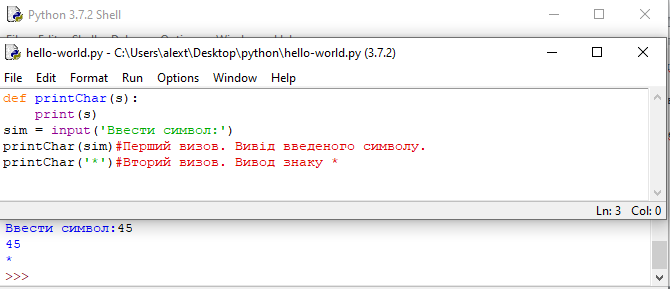
Для визначення процедури використовується ключове слово def, потім вказується ім'я процедури і в дужках її формальні параметри, якщо вони присутні. Після ставиться двокрапка і з наступного рядка з відступом в 4 пробілу вказуються команди. Процедура - допоміжний алгоритм, який виконує деякі дії.

Процедура повинна бути визначена до моменту її виклику. Визначення процедури починається зі службового слова def.

Виклик процедури здійснюється за її імені, за яким слідують круглі дужки, наприклад, Err ().

В одній програмі може бути скільки завгодно багато викликів однієї і тієї ж процедури. Використання процедур скорочує код і підвищує читабельність.

Написати процедуру, яка друкує раз вказаний символ (введений з клавіатури), кожен з нового рядка. Тут важливі відступи.

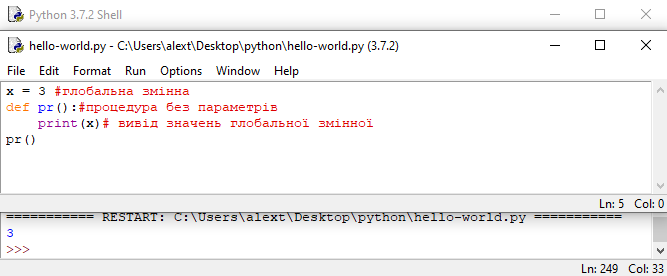


**Глобальна змінна** - якщо їй присвоєно значення в основній програмі (поза процедурою).

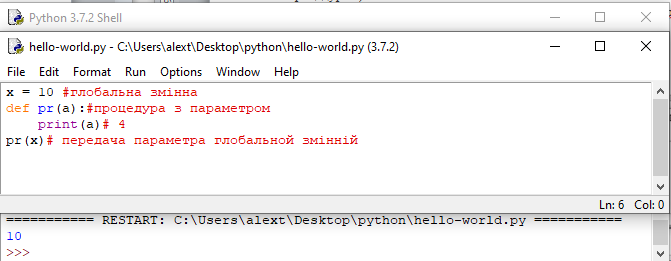
**Локальна змінна** (внутрішня) відома тільки на рівні процедури, звернутися до неї з основної програми і з інших процедур не можна.

**Параметри процедури** - локальні змінні.

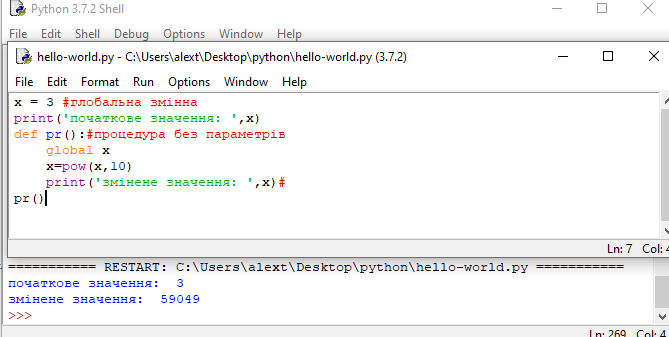
Наприклад:



Наприклад:

  
Існує можливість змінити значення глобальної змінної (не створюючи локальну). У процедурі за допомогою слова **global:**

Наприклад:



**Функції та їх аргументи**

Функція в python - об'єкт, який приймає аргументи і повертає значення. Зазвичай функція визначається за допомогою інструкції def.

Визначимо найпростішу функцію:

def add (x, y):

return x + y

Інструкція return каже, що потрібно повернути значення. У нашому випадку функція повертає суму x та y.

Наприклад:

>>> add(1, 10)  
11  
>>> add('abc', 'def')  
'abcdef'

Функція може бути будь-якої складності і повертати будь-які об'єкти (списки, кортежі, і навіть функції!):

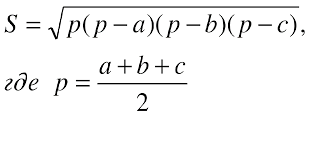
**Лабораторне завдання**

1. Створити будь яку множину .
2. З v = ['ccc', 'ddd', 'yyy', 'iii', ‘ccc’, ‘dd’] видалити елементи які повторюються.
3. Зробить копія множини v = ['ccc', 'ddd', 'yyy', 'iii', ‘ccc’, ‘dd’]
4. Розрахувати кількість елементів у множині v = ['ccc', 'ddd', 'yyy', 'iii', ‘ccc’, ‘dd’].
5. Створіть дві множини . Об’єднайте їх. Знайдіть їх перетин.
6. Додайте будь який елемент у множину.
7. Виділіть будь який елемент із множини.
8. Видаліть перший елемент із множини.
9. Очистіть усі множини.
10. Створіть процедуру змінення значення глобальної змінної, не створюючи локальної, за допомогою слова **global**.

x= 5- глобальна змінна. Треба возвести її у четверту ступень. Ввести змінене значення глобальної змінної.

1. Створіть функцію, яка розраховує суму цифр числа, яке вводиться з екрану.
2. Визначте, чи є три трикутника рівновеликими. Довжини сторін вводити з клавіатури. Для підрахунку площі трикутника використовувати формулу Герона. Обчислення площі оформити у вигляді функції з трьома параметрами.

Формула Герона:

[](https://sites.google.com/site/moiboarkin/laboratornye-raboty/5-kurs/laboratornye-raboty-po-python/l-r-7/geron.gif?attredirects=0)

1. Введіть одновимірний масив A довжиною m. Поміняйте в ньому місцями перший і останній елементи. Довжину масиву і його елементи введіть з клавіатури. У програмі опишіть процедуру для заміни елементів масиву. Виведіть вихідні та отримані масиви.
2. Дано 3 різних масиву цілих чисел (розмір кожного не перевищує 15). У кожному масиві знайдіть суму елементів і середньо арифметичне значення.
3. Знайдіть кількість елементів сгенерованого масиву, які відмінні від найбільшого елемента не більше ніж на 10%.

Програмування з використанням функцій

* + - 1. Дані дійсні числа а0, ..., a6. Отримати для x = 1,3,4 значення р (х + 1) - р (х), де р (у) = a6y6 + a5y5 + ... + а 0.
      2. Дано. Дійсні числа s, t, а 0, ... А12. Отримати р (1) -р (t) + p2 (s-t) -pз (t), де р (х) = а12х12 + а11х11 + ... + а 0.
      3. ОбчислитиZ = (X1 + Y1) / (Х2 - Y2), де Х1 і Х2 - коріння рівняннях х2 + 2x + 1 = 0;Y1 і Y2 - коріння рівнянняy х2 -4у +4 = 0. (Все коріння дійсні).
      4. Знайти найменші елементи і номера рядків і стовпців, в яких вони розташовані, для матриць А (4,5) і В (5,3).
      5. Обчислити z = (s1 + s2) / (k1 + k2), де S1 і K1 - сума і кількість елементів масиву X (N); S2 і k2 - сума і кількість елементів масиву Y (M).
      6. Вивести на друк елементи целочисленной матриці N (3,6), кратні трьом.
      7. Обчислити Z = (es1 + es2) / (k1 \* k2), де S1 і К1 - сума і кількість позитивних елементів масиву X (10); S2, і К2 - сума і кількість негативних елементів масиву Y (15).
      8. Перетворити масиви Х (10) і Y (15), розташувавши в них поспіль тільки позитивні елементи.Замість інших елементів записати нулі.
      9. Обчислити Z = (Xmax-Ymin) / 2, де Хmах - максимальний елемент масиву Х (10): Ymin - мінімальний елемент масиву Y (12): Хmax і Ymin обчислювати в одній підпрограмі.
      10. Обчислити суми позитивних елементів кожного рядка для матриць А (5,6) і В (4,3), використовуючи підпрограму-функцію.
      11. Обчислити і запам'ятати кількість негативних елементів кожного стовпця для матриць А (5,5), В (4,5).
      12. Дано дійсні числа а1, ..., an, b1, ..., bm. У послідовності а1, ..., an і послідовності b1, ..., bm всі члени, які йдуть за членом з найбільшим значенням (за першим по порядку, якщо їх декілька), замінити на 0.5.
      13. Дано цілі числа а1, ..., an, b1, ..., bm, К. Якщо в послідовності а1, ..., an є член зі значенням К, тоді усі наступні замінити значенням К. Якщо немає жодного члена зі значенням К, то перший по порядку найбільший член цієї послідовності замінити на значення К та наступні також. За таким же правилом перетворити b1, ..., bm, стосовно до значення 10.