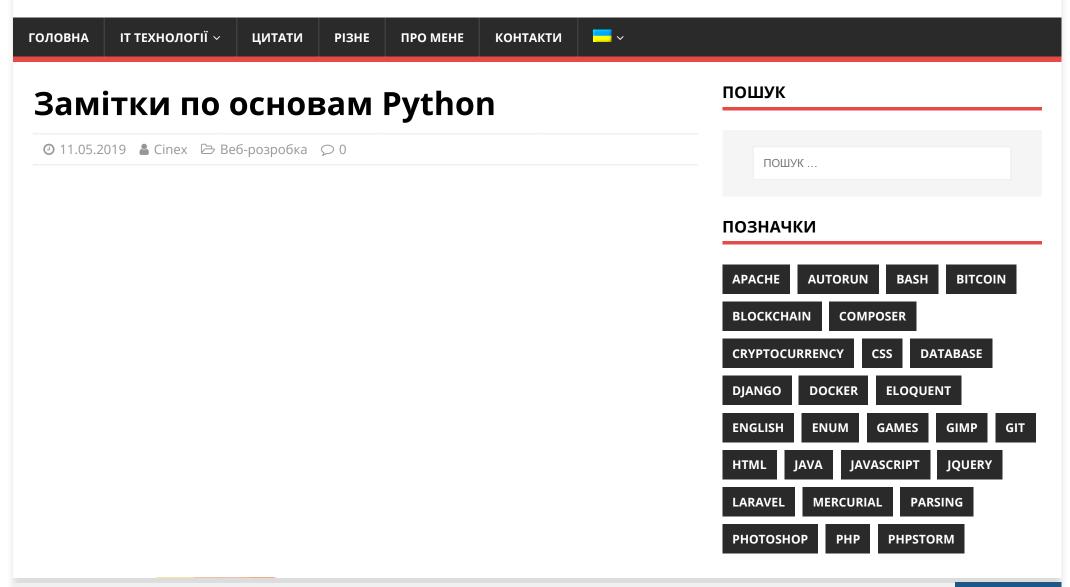
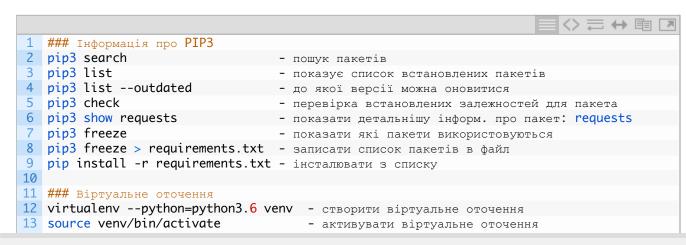
THE CINEX

ВСЕ ДЛЯ ВСІХ





Різне





останні нотатки

20 Laravel Eloquent порад і трюків 15.12.2019

Як зробити скріншот сайту по URL на PHP 04.08.2019

Docker Cheat Sheet

05.07.2019

Гарячі клавіші Ubuntu Linux

01.07.2019

Git happens! 6 типових помилок Git і як їх виправити

22.06.2019

НЕДАВНІ КОМЕНТАРІ

```
14 deactivate
                                           - вийти з віртуального оточення
15
16 ### Перемінні оточення
17 PS1="something else..." - зміна перемінного оточення
18 bash
                            - відкрити суб-процес.
19 export token="1234"
                              - экспорт перемінної в дочірній процес // echo $token
                          - зберегти перемінну в терміналі ВАЅН
20 nano ~/.bashrc

    source ~/.bashrc - оновити оточення
    import os - імпортувати в python перемінне оточення
    os.environ['token'] - відобразити перемінну

24
25 ### різне
26 which firefox - дізнатися шлях до програми
27 killall python3 - вбити запущені процеси пітона
28
29 a = 1
30 b = 2
31 a, b = b, a //обмін значеннями перемінних
32
33 передаємо список аргументів в функцію або словник(kwargs)
34 aras = [1,2,3]
35 kwargs = {'one': 1, 'two': 2}
36 test(*args, **kwargs)
```

Приклад 1.

```
■ <> ≡ ■ Python
1 print(3 ** 9) # 3 привести в степінь 9
2 print(9 // 3) # покаже скільки трійок в числі 9 - 3
3 print(9 % 3) # знайти залишок при діленні
4 # res:
5 # 19683
6 # 3
7 # 0
9 res = input('Введіть щось: ')
10 print(res)
11
12 num_1 = input('Введіть перше число: ')
13 num_2 = input('Введіть друге число: ')
14 print(int(num_1) + int(num_2)) # приведення типів
15
16 for j in 'Hi!':
       print(j * 2, '\n')
```

```
18  # res: HH \n ii\n !!
19
20  i = 0
21  while i < 10:
    print(i, ' ')
    i += 2
24  # res:
25  # 0
26  # 2
27  # 4
28  # 6
29  # 8
30
31  a = [a + b for a in 'list' if a != 's' for b in 'soup' if b != 'u']
32  print(a)</pre>
```

Приклад 2.

```
1 list_one = [23, 45, 6.45, 's', ['h', 'i', '!']]
2 list_one.append('Hi!')
3 list_one.remove(45) # видаляє по ключу
4 list_one.insert(0, 81) # місце в списку - значення
5 #list_one.index('HI!') # знайде індекс елементу
6 print(list_one[-1]) # вивести з кінця
7 print(list_one[2:-2:1]) # з якого елементу стартуємо, пропускаємо останні два елементи
8 print(list_one[::2]) # вивести кожен другий елемент
9 print(list_one)
10
11
12
13 # кортеж - це теж ті самі списки, тільки в списках ми можемо змінювати конкретні елемен
14 # а в кортежах таке робити не можна.
15 # На кортежі тратиться меньше пам'яті, а ніж на списки
16
17 kortej = (23, 's', 5.34)
18 kortej[0] = 13 # переписати значення 23 не вийде!
19 kortej_2 = 23, 45, 67 # це також буде кортежом
20 print(kortej)
21
22 tuple1 = (12, 45, 14.5, 'asd') # кортежі міняти не можна
23 no_tuple = [12, 45, 14.5, 'asd']
24
25 print(tuple1.__sizeof__())
```

```
26 print(no_tuple.__sizeof__())
27 # res: 56 / 72 кортеж займає менше пам'яті
28
29
30 # Словник - це щось схоже, як масив (асоціативний масив) в РНР: ключ - значення
31 d = {'test': 1}
32 dd = dict(short='dict', longer='dictionary') # {'short': 'dict', 'longer': 'dictionary'
33 ddd = dict.fromkeys(['a', 'b']) # створюємо ключі без значень
34 dede = dict.fromkeys(['a', 'b'], 1) # всім ключам буде призначено значення 1
35 dada = \{a: a ** 2 \text{ for a in range}(7)\} \# \{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36\}
36 print(dada)
37
38 ax = \{23, 56, 32, 156, 90\}
39 xa = \{32, 54, 47, 76, 13\}
40 print(ax.isdisjoint(xa)) # перевіряє чи пересікається хоть одне значення
41 ax.update(xa) # об'єднює множини
42 print(ax)
43 # ax.intersection_update(ха) # пересікання
44 # ax.difference_update(ха) # пересікання значень
45 print(ax)
46 ax.remove(444) # не видалить і видасть помилку
47 ax.discart(444) # якщо знайде - видалить, інакше помилки не буде
48
49
50 # безлічі (множества) - це ті ж самі списки. Різниця в тому, що тут немає елементів, як:
51 df = set("hello") # Безлічі в python - "контейнер", що містить не повторюються у випаді
52 dff = {'32', 23}
53 af = \{i ** 2 \text{ for } i \text{ in range}(10)\} \# \{0, 1, 64, 4, 36, 9, 16, 49, 81, 25\}
54 print(af)
55 x = 23
56 print(x in dff) # перевіряє чи є таке число
57
58 bf = frozenset("Qerty") # це те саме, що списки і кортежі - заморожене множество
59 \# bf.add = (119)
60 print(bf)
```

Приклад 3.

```
1 def some_func():
2 pass # нічого не вертає
3
4 # якщо к-сть параметрі не визначена
5 def dcc(*args): # передаємо у вигляді списку
```

```
return args
8 def ddc(**kwargs): # передаємо у вигляді кортежа
9
       return kwargs
10
11 add = lambda x, y: x + y
12 print(add(2, 5)) # res: 7
13
14 x = int(input('Введіть перше число: '))
15 y = int(input('Введіть друге число: '))
16
17 try:
18
      print(x / y)
19 except ZeroDivisionError:
20
      print('Не можна ділити на НУЛЬ!')
21
22
23 # для виконання критичних операцій
24 def ddaa(x, y):
25
      return x + y
26
27 with ddaa(10,0) as gr:
28
      gr.__init__()
29
30
31 def decorator(funct):
32
       def wrapper():
33
           print("Before")
34
           funct()
35
           print("After")
36
       return wrapper
37
38 @decorator
39 def show():
40
       print('I\'m just a func')
41
42 # dec = decorator(show)
43
44 show()
```

•

PYTHON

TIPS AND TRICKS

ШПАРГАЛКИ



Невеликий посібник по mercurial

ЗАЛИШТЕ ПЕРШИЙ КОМЕНТАР

Залишити коментар

Вашу адресу електронної пошти не буде опубліковано	
Коментувати	
	//
Ім'я <mark>*</mark>	
Email *	

□ Збережіть моє ім'я, електронну пошту у цьому веб-переглядачі наступного разу, коли я коментуватиму.

ОПУБЛІКУВАТИ КОМЕНТАР