Основы Python

Строки и итераторы

- работа со строками и списками
- работа с итерируемыми типами посредством for
- запуск скрипта и параметры

- Чаще правильнее проходиться циклом for по итерируемому объекту, чем находить от него длинну и строить for по индексам

```
s = 'aeiouyAEIOUYzxcvbnmSDFGHJa'
l = 'aeiouy'
count = 0
for i in range(len(s)):
    if s[i].lower() in l:
        count += 1
for i in s:
    if i.lower() in l:
        count += 1
print count
```

- Использовать логический оператор in для проверки вхождения переменной в последовательность

```
char = ‘a’
# вхождение символа или подстроки в строку
print char in ‘asdfgh’
print char in ‘qwerty’
# вхождение переменной в list или tuple
print char in [‘a’, ‘b’]
print char in (‘q’, ‘w’)
# проверка наличия ключа в dict
print char in {‘a’: 1, ‘b’: 2}
print char in {‘c’: 3, ‘d’: 4}
```

- Играйтесь с типами - перегоняйте данные из одного в другой

```
# преобразование строки к списку
s = ‘python’
l = list(s) # [‘p’, ‘y’, ’t’, ‘o’, ‘n’]
# список в строку
new_str = ‘’.join(l) # ‘python’

# dict в список кортежей, получение списка ключей и значений
d = {‘a’: 1, ‘b’: 2}
print d.items(), d.keys(), d.values()
> [(‘a’, 1), (‘b’, 2)] ['a', 'b'] [1, 2]
# обратное преобразование списка кортежей к dict
d = dict([(‘a’, 1), (‘b’, 2)]) # {‘a’: 1, ‘b’: 2}
# создание dict из двух списков в одном из которых ключи во
втором значения
list_of_tuple = zip([‘a’, ‘b'], [1, 2]) # [(‘a’, 1), (‘b’, 2)]
# потом результат можно преобразовать в dict
d = dict(list_of_tuple) # {‘a’: 1, ‘b’: 2}
```

- Проход фором по элементам разных типов.

```
s = ‘python’
for i in s:
    print i

l = [‘p’, ‘y’, ’t’, ‘o’, ‘n’]
for i in l:
    print l

d = {‘a’: 1, ‘b’: 2}
for i in d.keys():
    print i, d[i]
for i in d.values():
    print i
for i, k in d.items():
    print i, k
```

- Передача параметров в скрипт.

$python script.py hello world

```
import sys
print sys.argv # [‘script’, ‘some_arg’, ‘other_arg’]
# правила: проверять наличие аргументов перед тем
как их использовать
if len(sys.argv)>=3:
    print sys.argv[1] + ‘ ’ sys.argv[2]

# правила: проверять возможность преобразования
агрумента если это подразумевается
a, b = sys.argv[1:2]
if a.isdigit() and b.isdigit():
    print a + b
```

- https://docs.python.org/2/tutorial/
- В туториале достаточно примеров использования встроенных типов данных и функций
- Пройдите (пробегите) почитайте еще раз первые 5 пунктов (функции можно пропустить)
- Если сложно воспринимать инглиш, то думаю гугл вам поможет найти адекватный перевод