gb, факультет Разработчик (фронтенд)

Лекция № 2 по Python



**Основы языка**

**PYTHON**

код с лекций

(преподаватель С. Каменецкий)

( лекция от 17.12.2022)

Содержание:

|  |  |
| --- | --- |
|  | Стр. |
| 1. Файлы | 3 |
| Основное API работы с файлами | 3 |
| ‘a’ / ‘w’ | 5 |
| в чем отличие **.writelines**(…) от **.write**(….) | 6 |
| Сокращенная запись через **with open()** | 7 |
|  |  |
| 2. Функции и модули | 8 |
| Вызов ф-ции из другого файла | 8 |
| Создание **алиаса** (укороченного псевдонима ф-ции) | 8 |
| Значения аргументов ф-ции по умолчанию | 9 |
| Передача не ограниченного кол-ва аргументов | 9 |
|  |  |
| 3. Рекурсия | 10 |
|  |  |
| 4. Кортежи | 11 |
| Создание кортежей и основной API работы с ним | 11 |
| Перебор кортежей через for | 12 |
| Распаковка кортежей в отдельные переменные | 12 |
|  |  |
| 5. Словари | 13 |
| Создание словаря | 13 |
| Перебор словаря | 13 |
|  |  |
| 6. Множества set | 14 |
| базовый API работы с set | 14 |
|  |  |

1. Файлы

# Основное API работы с файлами

**a**  = **дозаписать** в уже имеющийся файл

* Если внутри файла уже есть текст, то будет дозапись (как продолжение )
* если дописывать в НЕсуществующий ф-л, он будет создан и записан новой инф-ей

**r**  = **читаем** данные из файла

чтение из НЕ существующего ф-ла - будет ошибка

**w**  = **создать файл + записать в него** (мод),

* если в файле что-то есть, то W все удалит и запишет с чистого листа

**w+**  открыть файл для записи + чтение из него

**r+** открыть для чтения и дописывать в него

**open(**'путь', 'мод'**) -** связь с файлом

**.writelines(**'имя файла'**)** – откуда берем инфу для записи

**.write (**'текст для записи'**)** – аргумент – это источник инф-ции

вариант 1.1.

('1sport.txt', **'a'**):

# мои данные (пример, список)

sport = ['sambo', 'judo', 'bjj']

# переменная data, связываем ее файлом ('путь', 'мод')

data = **open**('1sport.txt', 'a')

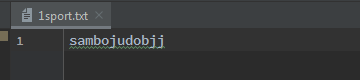
# writelines (для записи берет данные из переменной **sport**)

data**.writelines**(sport)

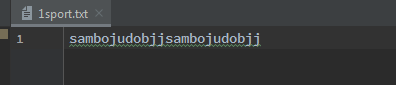
# разрыв связи между 'sport' и '1sport.txt' (файл на диске)

data**.close**()

В консоли после первого исполнения программы:



В консоли после второго исполнения программы (дозапись):



вариант 1.2.

('1sport.txt', **'w'**), файл '1sport.txt' уже имеет в себе запись:

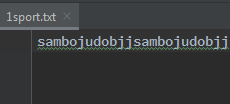
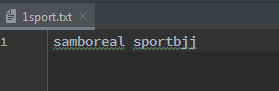
sport = ['sambo', 'judo', 'bjj']

# переменная data, связываем ее файлом ('путь', 'мод')

data = **open**('1sport.txt', **'w'**)

data**.writelines**(sport)

data**.close**()

файл '1sport.txt' – **ДО** файл '1sport.txt' – **ПОСЛЕ**

Т.е. СТАРЫЕ данные автоматически удалились,

НОВЫЕ записались поверх их

вариант 1.3.

в чем отличие **.writelines**(…) от **.write**(….) :

sport = ['sambo ', 'real\_sport ', 'bjj'] # мои данные для записи в файл

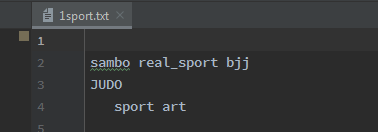
data = **open**('1sport.txt', 'a') # ('откуда взять инф-ию', 'мод')

data**.writelines**(sport) # writelines (запись в строку, без разделителей)

data**.write**('\nJUDO\n') # запись в файл идет прямо из аргумента

data**.write**(' sport art\n') # запись в файл идет прямо из аргумента

data**.close()**  # разрыв связи между 'sport' и '1sport.txt'



Консоль 1

Убрана **.writelines()** , идет дозапись (‘а’) из **.write()**

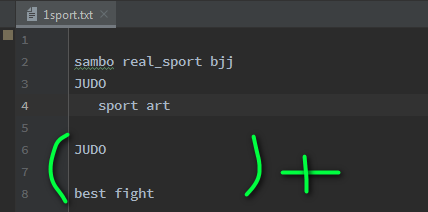
sport = ['sambo ', 'real\_sport ', 'bjj'] # мои данные для записи в файл

data = open('1sport.txt', 'a') # ('откуда взять инф-ию', 'мод')

data.write('\nJUDO\n') # запись в файл идет прямо из аргумента

data.write('\nbest fight\n') # дозапись в файл идет прямо из аргумента

data.close() # разрыв связи между 'sport' и '1sport.txt'



Вариант 1.4.

Сокращенная запись через **with open()** :

**with open**('1sport-with.txt', 'w') **as** data:

data**.write**('sambo\n') # запись в файл идет прямо из аргумента

data**.write**(' best sport\n')

**as data** – принять как переменную data

**write** – информацию взять из аргумента

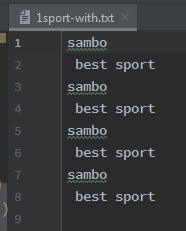
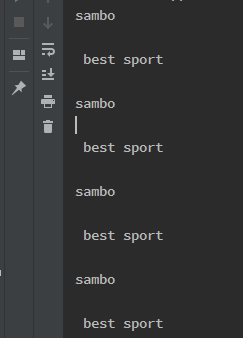
разрыв связи между файлами – автоматический

**Чтение (‘r’)**  **из файла с выводом на экран**

Есть созданный ранее файл '1sport-with.txt'

Его надо прочитать **(‘r’)** / вывести на экран консоли

path = '1sport-with.txt'  
data = **open**(path, **'r'**)  
  
**for** i **in** data:  
 **print**( i )  
  
data**.close**( )

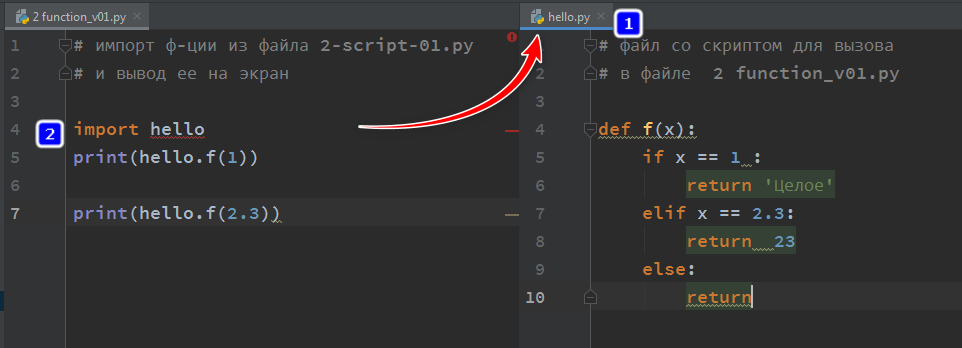
файл '1sport-with.txt' Печать в консоль

timecode: 11:36

2. Функции и модули

Пример 2.1.

Вызов ф-ции из другого файла



В программе PyCharm

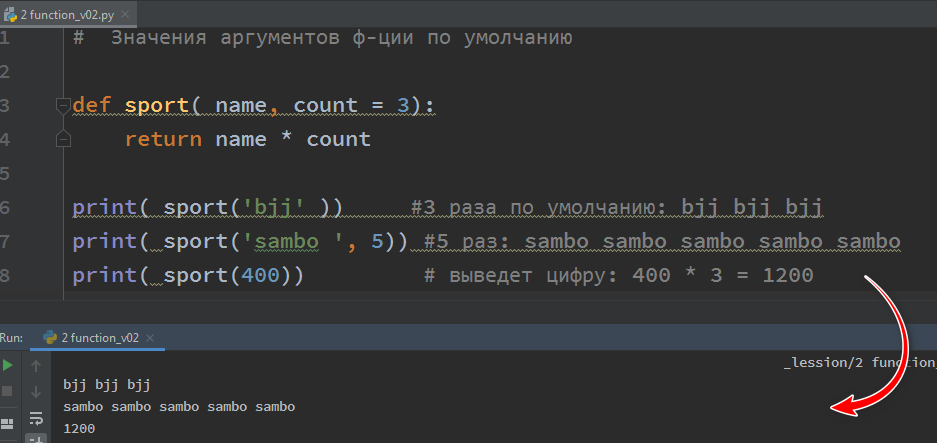
Пример 2.2.

Создание **алиаса** (укороченного псевдонима ф-ции)

# алиас (псевдоним ф-ции)  
**import** hello **as** h  
  
**print**(**h**.**f**(2.3)) # вызов ф-ции через псевдним (имя hell = h)

# Значения аргументов ф-ции по умолчанию

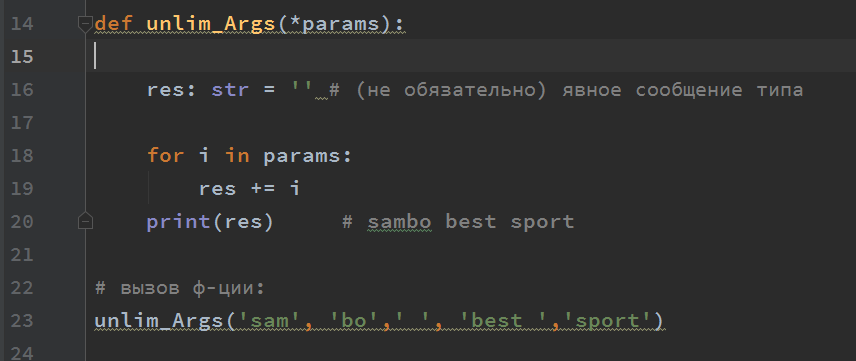
Пример 2.3.



В программе PyCharm

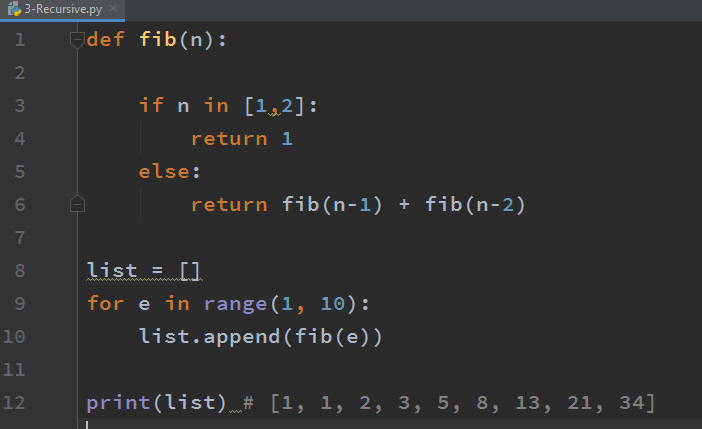
# Передача не ограниченного кол-ва аргументов

Пример 2.4.



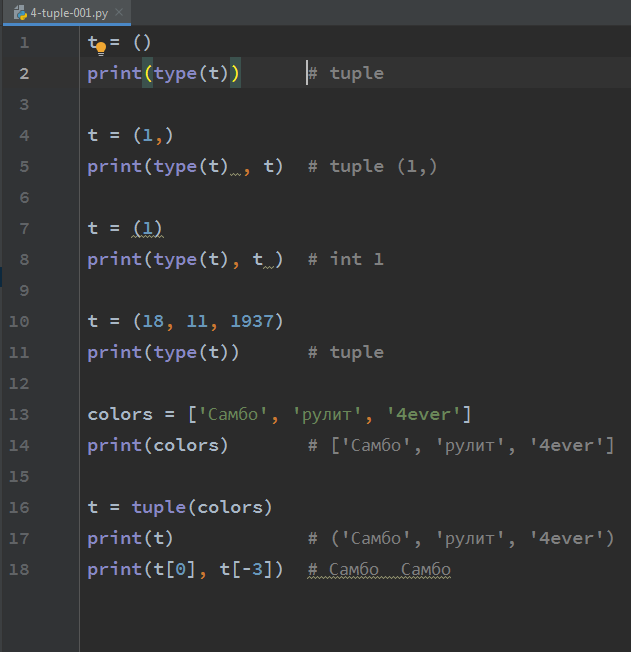
3. Рекурсия

Пример (бегло ) 3.1: Фибоначчи.



4. Кортежи

Кортеж – неизменяемый «список»

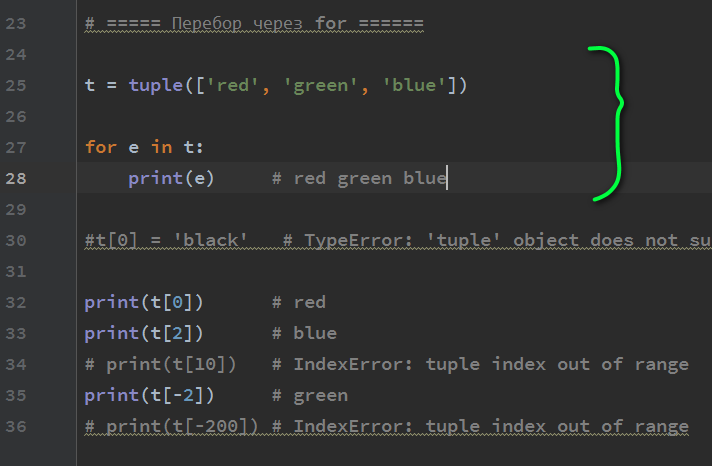


Внимание:

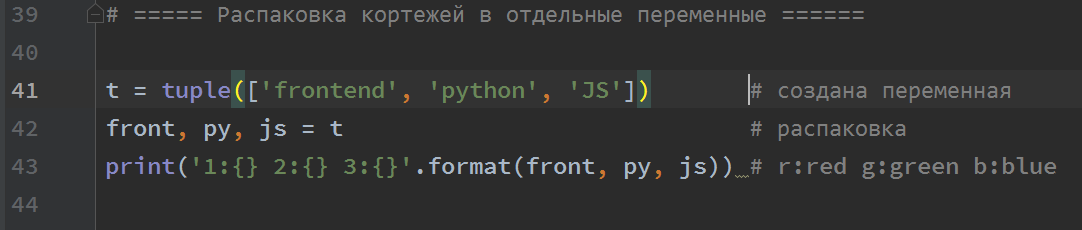
a = (1, ) это кортеж **tuple**

a = (1 ) это будет число **int**

# Перебор кортежей через for



# Распаковка кортежей в отдельные переменные



5. Словари

Пример 5.1. Создание словаря

dictionary = { } # создать пустой словарь

dictionary = \  
 { # / делает перенос строки

'up': '↑', # ключ: "значение"

'left': '←', # ключ: "значение"

'down': '↓', # ключ: "значение"

'right': '→' # ключ: "значение"

}

print(dictionary) # {'up':'↑', 'left':'←', 'down':'↓', 'right':'→'}

print(dictionary['left']) # ←

# типы ключей могут отличаться

Пример 5.2. Перебор словаря

dic = { }  
dic = \  
 {  
 'one': 'SAMBO',  
 'two': 'BJJ',  
 'three': 'judo'  
 }

# просмотр ключей  
for k in dic.keys():  
 print(k ) #one two three  
  
# просмотр значений ключей (вариант 1)  
for p in dic.values():  
 print(p) #SAMBO BJJ judo  
  
# просмотр значений ключей (вариант 2)  
for v in dic:  
 print( dic[v] ) #SAMBO BJJ judo

6. Множества set

Неупорядоченная совокупность элементов

a = {1, 2, 3, 5, 8}

b = {'2', 'string', 8, 13, True}

print( type(a) ) # set

print( type(b) ) # set

Пример 6.1. базовый API работы с set

**set( )** создать множество (аргумент = Список или кортеж)

**.add** добавить значение в множество

**.remove** удалить из множества

**.discard** не будет генерировать исключения при отсутствии элемента:

**.clear** полная очистка множества

colors = {'red', 'green', 'blue'}  
print(colors) # Множество: {'red', 'green', 'blue'}

colors**.add**('gray') # ДОБАВИТЬ 'gray'

print(colors) # {'red', 'green', 'blue','gray'}

colors**.add**('red') # попытка создать ДУБЛЬ 'red'  
print(colors) # {'red', 'green', 'blue'}

colors**.remove**('red') # УДАЛИТЬ 'red'  
print(colors) # {'green', 'blue','gray'}

# colors**.remove**('red') # KeyError: 'red' (нечего удалять)

colors**.discard**('red') # ok  
print(colors) # {'green', 'blue','gray'}

colors**.clear**() # { } ОЧИСТКА множества

print(colors) # set()

Пример 6.2. базовый API работы с set

**copy**  копирование в другую переменную

**union( )** объединяет два множества и возвращает новое множество

**intersection()** производит операцию пересечения множеств и возвращает новое множество

**difference** разность множеств возвращает те элементы, которые есть в первом множестве, но отсутствуют во втором

a = {1, 2, 3, 4, 5}  
b = {4, 5, 6, 7, 8}  
  
c = a**.copy()** # КОПИЯ  
print(c) # c = {1, 2, 3, 4, 5}  
  
u = a**.union**(b) # ОБЪЕДИНЕНИЕ  
print(u) # u = {1, 2, 3, 4, 5, 6, 7, 8}  
  
i = a**.intersection**(b) # ТОЛЬКО СОВПАДЕНИЯ  
print(i) # i = {4, 5}  
  
dl = a**.difference**(b) # ЕСТЬ ТОЛЬКО В МНОЖЕСТВЕ a  
print(dl) # dl = {1, 2, 3}  
  
dr = b**.difference**(a) # ЕСТЬ ТОЛЬКО В МНОЖЕСТВЕ b  
print(dr) # dr = {8, 6, 7}  
  
q = a \  
 .union(b) \  
 .difference(a.intersection(b))  
# {1, 21, 3, 13}  
  
s = **frozenset**(a) # заморозка