

Работа с текстовыми файлами

Открытие файла для чтения или для записи информации

1. Для того, чтобы открыть файл для чтения информации, надо завести файловую переменную. Дальнейшее обращение к файлу будет вестись через эту переменную
fin = open("input.txt")
При использовании функции `open` в таком виде надо понимать:
 - текстовый файл и ваш программный файл должны лежать в одной папке;
 - файл по умолчанию открывается для чтения.
2. После чтения информации из файла, его надо закрыть **fin.close()**. Если вы его не закроете, операционная система будет считать его открытым. Открытый файл нельзя удалить, переименовать и т.д.
3. Для того, чтобы открыть файл для записи информации, также используют команду `open`, только с параметром "w": **fout = open("output.txt", "w")**. После работы с файлом его также надо закрыть **fout.close()**. Если файла не было, то он создастся в той же папке, в которой сохранён программный файл.
4. Кроме того, файл может быть открыт с другими целями, для этого используют другие параметры. Параметр "a" открывает файл для добавления информации в конец файла:
fout = open("output.txt", "a")

Чтение одной строки из файла

1. Чтение строки вместе с переносом строки "\n": **s = fin.readline()**.
2. Если нужен целочисленный тип, то перевод строки уберёт функция `int`:
s = int(fin.readline())
3. Если же нужно оставить строку, но есть необходимость убрать перенос строки, то можно использовать метод `strip()`:
s = fin.readline().strip() – метод `strip()` удаляет пробелы и перенос строк с двух сторон
s = fin.readline().rstrip() – метод `rstrip()` удаляет пробелы и перенос строк справа
s = fin.readline().lstrip() – метод `lstrip()` удаляет пробелы и перенос строк слева

Чтение строкового или целочисленного массива из файла

1. Можно просто перебирать строку за строкой и записывать их в файл:

```
1 fin = open("15.txt")
2 A = []
3 for st in fin:
4     A.append(int(st))
5
6 print(A)
7 fin.close()
```

Search Python Shell

Commands execute without debug. Use arrow keys for history.

3.3.2 (v3.3.2:d047928ae3f6, May Python Type "help", "copyright" [evaluate 15.py] [34, 455, 367, 56]

2. Тоже самое, что в предыдущем пункте, только короче, с помощью генератора:

```
1 fin = open("15.txt")
2 A = [int(st) for st in fin]
3 print(A)
```

Search Python Shell

Commands execute without debug. Use arrow keys for history.

3.3.2 (v3.3.2:d047928ae3f6, May Python Type "help", "copyright", [evaluate 15.py] [34, 455, 367, 56]

3. Всю информацию из файла можно считать как список строк, в конце каждой строки “\n”:

s = fin.readlines()

```
1 fin = open("15.txt")
2 A = fin.readlines()
3 print(A)
```

Search Python Shell

Commands execute without debug. Use arrow keys for history.

3.3.2 (v3.3.2:d047928ae3f6, May 16 2013, 0
Python Type "help", "copyright", "credits"
[evaluate 15.py]
['34\n', '455\n', 'qwerty\n', 'qqqq\n']

Если нужен массив целых чисел, то применяем **map**:

```
1 fin = open("15.txt")
2 A = list(map(int, fin.readlines()))
3 print(A)
```

Search Python Shell

Commands execute without debug. Use arrow keys for history.

3.3.2 (v3.3.2:d047928ae3f6, May 16 2013
Python Type "help", "copyright", "credi
[evaluate 15.py]
[34, 455, 367, 56]

4. Всю строку из файла можно считать в одну строку с помощью функции **s = fin.read()**, при этом внутри строки будут содержаться переносы строк. Сама строка будет выглядеть как содержимое файла:

```
1 fin = open("15.txt")
2 A = fin.read()
3 print(A)
```

Search Python Shell

Commands execute without debug. Use arrow keys for history.

34
455
qwerty
qqqq

5. Для последнего случая можно применить **split()**, чтобы получить массив строк:

```
1 fin = open("15.txt")
2 A = fin.read().split()
3 print(A)
```

Search Python Shell

Commands execute without debug. Use arrow keys for history.

3.3.2 (v3.3.2:d047928ae3f6, May 16 2013
Python Type "help", "copyright", "cre
[evaluate 15.py]
['34', '455', 'qwerty', 'qqqq']

Если нужен массив целых чисел, то применяем **map**:

```
1 fin = open("15.txt")
2 A = list(map(int, fin.read().split()))
3 print(A)
```

Search Python Shell

Commands execute without debug. Use arrow keys for history.

3.3.2 (v3.3.2:d047928ae3f6, May 16 2013,
Python Type "help", "copyright", "credit
[evaluate 15.py]
[34, 455, 367, 56]

Запись информации в файл

Для записи информации в файл, его надо открыть с параметром “w”. Если файл заранее не был создан, то он создастся в результате работы этой команды. Если в файле было что-то записано, то информация будет перезаписана. Для записи используется метод **write**, смотрите листинг программы:

```
1 fin = open("15.txt")
2 fout = open("16.txt", "w")
3
4 A = int(fin.readline()) # чтение первого числа из файла
5 B = int(fin.readline()) # чтение второго числа из файла
6
7 fout.write(str(A+B)) # запись результата сложения в файл
8                       # в файл можно записать только тип str!
9 fin.close()
10 fout.close()
```