



ЦЕНТР  
ДОПОЛНИТЕЛЬНОГО  
ОБРАЗОВАНИЯ  
МГТУ им. Н.Э. Баумана

# Программирование на языке Python. Уровень 1.

Работа с файлами.

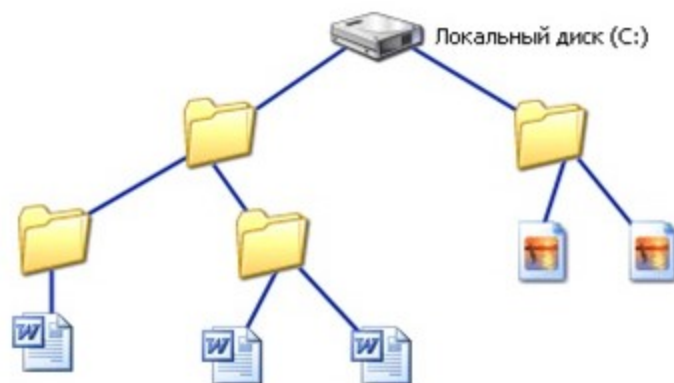
# Понятие файла

- ❑ **Файл** (file) — именованная область памяти на носителе информации.
- ❑ **Имя файла** состоит из двух частей, разделенных точкой: собственно имя файла и **расширение**, определяющее его тип (программа, данные и так далее).
- ❑ Имя файлу дает пользователь, а тип файла обычно задается программой автоматически при его создании.
- ❑ Работа с файлами реализуется средствами операционных систем.
- ❑ В различных операционных системах существуют различные форматы имен файлов.
- ❑ В операционной системе Windows имя файла может иметь длину до 255 символов, причем можно использовать национальные алфавиты, в том числе кириллицу.
- ❑ **Пример:** Документ программы Word.doc

# Понятие файловой системы

- ❑ Порядок хранения файлов на диске определяется используемой файловой системой.
- ❑ **Файловая система** — это система хранения файлов и организации каталогов.
- ❑ Если на диске хранятся сотни и тысячи файлов, то для удобства поиска используется многоуровневая **иерархическая файловая система**, которая имеет древовидную структуру.

**Пример.**



- ❑ Начальный, корневой каталог содержит вложенные каталоги 1-го уровня, в свою очередь, каждый из последних может содержать вложенные каталоги 2-го уровня и так далее.
- ❑ В каталогах всех уровней могут храниться и каталоги, и файлы.

# Операции с файлами

- ❑ **Открытие файла** (обычно в качестве параметров передается имя файла, режим доступа и режим совместного доступа, а в качестве результата выступает файловый **дескриптор**\*), кроме того имеется возможность в случае открытия на запись указание на действия с информацией файла.
- ❑ **Закрытие файла.** В качестве аргумента выступает значение, полученное при открытии файла. При закрытии все файловые буферы сбрасываются.
- ❑ **Запись** — в файл помещаются данные.
- ❑ **Чтение** — данные из файла помещаются в область памяти.
- ❖ При открытии файла (в случае, если это возможно), операционная система возвращает **дескриптор** (число или указатель на структуру), с помощью которого выполняются все остальные файловые операции.  
По их завершении операций файл закрывается, а дескриптор теряет смысл.

# Понятие файловой системы

- ❑ При работе с файлами в Python необходимо соблюдать следующую последовательность действий для каждого файла:
  1. Попытаться получить доступ к содержимому файла. Этот этап называется **открытием файла**.
  2. Проверить **наличие доступа**.
  3. Если доступ получен, реализовать **работу с файлом**.
  4. Когда все задачи с файлом будут решены, **закрыть доступ к файлу**.

# Работа с файлами в Python

## Открытие файла.

- Для открытия файла используется следующая функция:  
`open(имя_файла, режим_работы)`
- **имя\_файла** – в виде строки указывается имя файла или путь к файлу, используя системные соглашения, доступ к которому необходимо получить.

### Примеры:

- ❑ `'my.txt'` – указано только имя файла, включая расширение. Такой файл должен находиться в рабочем каталоге программы. Обычно это каталог, из которого запускается программа.
- ❑ `'data\my.txt'` – указан относительный путь (относительно рабочего каталога) к файлу.
- ❑ `'c:\documents\my.txt'` – указан абсолютный путь к файлу.
- **режим\_работы** – задается в виде строки, в которой может содержаться от одного до трех символов, с помощью которых указывается, что программа намерена с файлом делать.

# Работа с файлами в Python

- `"r"` – открыть существующий файл только для чтения.
- `"w"` – открыть файл для перезаписи. Если файл не существует, он создается, в противном случае содержимое файла удаляется, и размер файла сразу после открытия будет равен 0.
- `"a"` – открыть файл для добавления. Если файл не существует, он создается, в противном случае содержимое файла не удаляется, а все, что программа будет записывать в файл, будет добавляться в конец файла.
- `"r+"` – открыть существующий файл для чтения и записи.
- `"w+"` – открыть файл для перезаписи и чтения.
- `"a+"` – открыть файл для добавления в конец и чтения с любого места.

# Режимы работы с файлами. Модификаторы.

- После указания режима может следовать **модификатор**:
- **b** — файл будет открыт в бинарном режиме. Файловые методы принимают и возвращают объекты типа `bytes`.
- **t** — файл будет открыт в текстовом режиме (значение по умолчанию в Windows). Файловые методы принимают и возвращают объекты типа `str`.

В этом режиме будет автоматически выполняться обработка символа конца строки — так, в Windows при чтении будет подставлен символ `\n`.

- **Пример. Создание файла *file.txt* и запись двух строк.**

```
s1='Первая строка для теста записи в файл\n'
```

```
s2='Вторая строка для тех же целей\n'
```

```
f=open(u'file.txt', 'w')
```

```
f.write(s1)
```

```
f.write(s2)
```

```
print('Строки записаны в файл!')
```

```
...
```

# Работа с файлами в Python

## Чтение информации из файла.

➤ Чтобы прочитать информацию из файла, открытого **для чтения**, можно использовать два способа:

➤ 1. Прочитать весь текст из файла и записать его в переменную s.

```
f=open('file.txt', 'r')  
s=f.read()  
print(s)
```

➤ 2. Последовательно читать из файла отдельные строки с помощью цикла for.

```
f=open('file.txt', 'r')  
for x in f:  
    print(x)
```

## Заккрытие файла.

➤ После того как работа с файлом закончена нужно закрыть его.

```
f.close()
```

# Работа с файлами в Python

## Открытие файла в режиме добавления и чтения.

- Откроем файл в режиме добавления данных и чтения, запишем еще одну строку:

```
s3='Третья строка\n'
f=open(u'file.txt', 'a+')
f.write(s3)
# Для чтения файла перемещение указателя на начало
import io
f.seek(0, io.SEEK_SET)
# Чтение файла
s=f.read()
print(s)
f.close()
```

# Указатель чтения-записи в файле

- При работе с файлами, место в файле, с которого будет осуществляться чтение или запись информации, определяется положением **указателя чтения /записи**.
- При **открытии файла** в программе, указатель чтения/записи устанавливается в начало файла (кроме режима a+, если файл существует).
- Когда программа **записывает или читает** информацию из файла, указатель чтения/записи перемещается к концу файла на объем прочитанной/записанной информации.
- Когда указатель дойдет до конца файла дальнейшее чтение информации будет невозможно.
- Запись в файл не ограничивается размером файла.

# Работа с файлами в Python

## Изменение позиции указателя чтения-записи.

- › **seek (<Смещение> [, <позиция>])** — устанавливает указатель в позицию, имеющую заданное **<смещение>** относительно параметра **<позиция>**.

В качестве параметра **<позиция>** могут быть указаны следующие атрибуты из модуля **io** или соответствующие им значения:

- › **io.SEEK\_SET** или **0** — начало файла (значение по умолчанию);
- › **io.SEEK\_CUR** или **1** — текущая позиция указателя.

Положительное значение смещения вызывает перемещение к концу файла, отрицательное — к его началу;

- › **io.SEEK\_END** или **2** — конец файла.

Вывод значений этих атрибутов:

```
>>> import io
>>> io.SEEK_SET, io.SEEK_CUR, io.SEEK_END
(0, 1, 2)
```



**ЦЕНТР  
ДОПОЛНИТЕЛЬНОГО  
ОБРАЗОВАНИЯ**  
МГТУ им. Н.Э. Баумана



[do.bmstu.ru](https://do.bmstu.ru)