# Python-4
# Работа с файлами. Правила PEP.

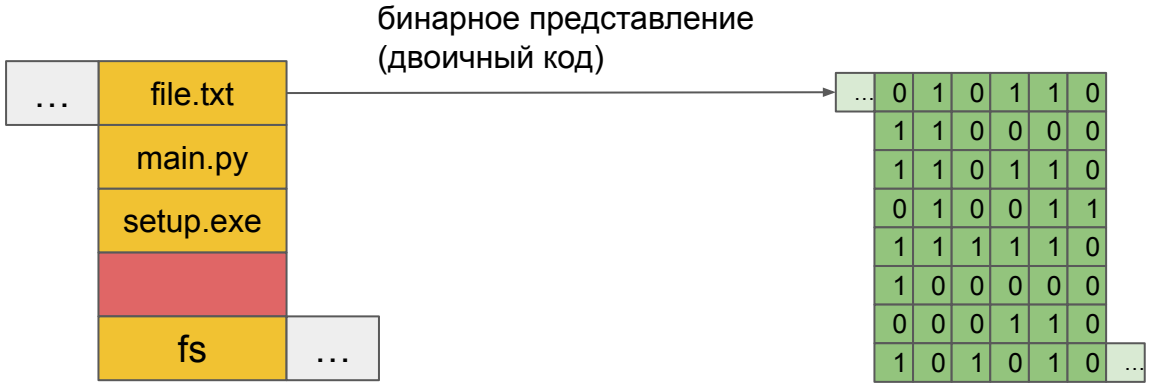
# Что такое файлы? Как представлены файлы в ПК?

**Файлы** - это определенное количество информации (*программа или данные*), имеющее имя и хранящееся в долговременной (*внешней*) памяти.

Файлы на компьютере представлены в виде бинарных данных (*двоичный код*), которые хранятся на жестком диске или других носителях информации. Каждый файл имеет свое уникальное имя и расширение, которое указывает на тип данных, хранящихся в этом файле.

# Как представлены файлы в ПК?

# Как представлены файлы в ПК?



# Работа с файлами.

**Работа с файлами** - это встроенные функции для чтения и записи файлов. Вы можете открыть файл с помощью функции open(), прочитать его содержимое с помощью метода read(), записать данные в файл с помощью метода write(), и закрыть файл с помощью метода close().

| | |
|---|---|
| (записать) - write() | w, a (дозапись) |
| (прочитать) - read() | r |

# Режимы работы с файлами.

| Режим | Обозначение |
|---|---|
| 'r' | открытие на чтение (является значением по умолчанию). |
| 'w' | открытие на запись, содержимое файла удаляется, если файла не существует, создается новый. |
| 'x' | открытие на запись, если файла не существует, иначе исключение. |
| 'a' | открытие на дозапись, информация добавляется в конец файла. |
| 'b' | открытие в двоичном режиме. |
| 't' | открытие в текстовом режиме (является значением по умолчанию). |
| '+' | открытие на чтение и запись |

# Работа с файлами. Пример.

main.py > ...

```python
f = open("myfile.txt", "w")  # открываем файл для записи
f.write("Hello, World!")  # записываем строку в файл
f.close()  # закрываем файл

f = open("myfile.txt", "r")  # открываем файл для чтения
print(f.read())  # читаем и печатаем содержимое файла
f.close()  # закрываем файл

# новый синтаксис
with open("myfile.txt", "r") as f:
    print(f.read())
```

# Практические задачи#1.

1. Создайте файл text.txt (с текстом каким угодно). В Python файле напишите код, который будет выполнять чтение данных с сохранением в некоторую переменную data. Вывести прочитанные данные из файла на экран, используя функцию вывода.
2. (по предыдущей задаче-1). Создайте переменную **new_string** с текстом "дополнительный текст" и дополните файл text.txt, используя режим "a" (добавление).
3. (по предыдущей задачу-2). Попробуйте использовать режим "w" (запись) для файла text.txt. Обратите внимание, что этот режим перезапишет содержимое файла, поэтому будьте осторожны.
4. Создайте файл text.txt с произвольным текстом, и затем напишите программу на Python, которая будет считать количество символов в этом тексте и выводить результат на экран. (*напоминаю функция для подсчета символов в строке len()*)
5. Реализовать **поиск и замену** слова.

- Создайте бесконечный цикл.
- В каждой итерации цикла выполните следующие шаги:
    - Запросите путь к текстовому файлу (переменная **path_to_file**).
    - Если пользователь ввел "0", завершите цикл.
    - Введите слово (переменная **old_word**).
    - Введите новое слово (переменная **new_word**).
    - Откройте файл, прочитайте его содержимое, выполните поиск и замену старого слова на новое слово и сохраните изменения обратно в файл.
    - Выведите сообщение о том, что слово было заменено в файле.
    - Повторите цикл.

# PEP.

**PEP** в Python означает *Python Enhancement Proposal*.

**PEP** - это документ, который описывает новые функции Python, процессы и окружения, которые влияют на большую часть сообщества Python.

*https://peps.python.org/pep-0008/*

# PEP.

**Именование:** Используйте **CamelCase** для имен классов, **lowercase_with_underscores** для функций и методов, и **UPPER_CASE_WITH_UNDERSCORES** для констант.

**Отступы:** Используйте 4 пробела на уровень отступа, а не табуляцию.

**Длина строки:** Строки должны быть не более 79 символов.

**Пробелы:** Используйте пробелы вокруг операторов и после запятой для улучшения читаемости.

**Импорты:** Импорты должны быть на отдельных строках и размещаться в начале файла.

```
if __name__ =='__main__':
```

1. Интерпретатор Python выполняет весь код, который сможет найти в файле.
2. Когда Python запускает "исходный файл" в качестве основной программы, он присваивает переменной __name__ значение __main__.

Фактически - это точка входа. Некоторые языки программирования просят настоятельно писать конструкцию подобного вида, а такие как Python - нет (но лучше писать).

```python
# 3D point
# const variables
X = 5
Y = 3
Z = 1

if __name__ == "__main__":
    print(f"({X}, {Y}, {Z})") # (5, 3, 1)
```

# PEP. Демонстрация ошибок. #1

```
1   x= 1
2   y =2
3   z=3
4
5   x +=2
6   y-=3
7   z*= 1
8   # x: 3, y: -1, z: 3
9   # функция вывода на экран
10  print(f"x: {x}, y: {y}, z: {z}")
11
12  string = "test string test string test string test string test string test string test string test string test string test string test
13  print(string)
```

# PEP. Демонстрация ошибок. #1 (исправленный)

```
1   x = 1
2   y = 2
3   z = 3
4   x += 2
5   y -= 3
6   z *= 1
7   print(f"x: {x}, y: {y}, z: {z}")
8   STRING = "test string " * 10
9   print(STRING)
```

# PEP. Демонстрация ошибок. #2

```
1   i= 0
2   """
3   .
4   """
5   while  i <4   :
6    # какой то код
7
8
9    print("Hello")
.0
.1   # увеличение i
.2   i+= 1
```

# PEP. Демонстрация ошибок. #2 (исправленный)

```
i = 0
while i < 4:
    print("Hello")
    i += 1
```

# Практические задачи#2. PEP.

1. Исправить код в соответствии с PEP. (task1.py + task2.py)

https://disk.yandex.ru/d/DUa4k8zaqqiCyg