



Урок 1. Python: Типы файлов и чтение

Часть 1. Материалы ученика

Программирование на Python и C++ · 9 класс

Урок 1 из 34 · 45 минут · Модуль «Файлы»

Уровень: начинающий программист; Python 3.

Программа: <https://schoolotzyv.ru/blog/python-cpp-9-klass-programma>

В файле три строки, одна из них пустая, а последняя не заканчивается переводом строки. Потеряет ли программа последнюю запись и равны ли пустая строка и конец файла?

Результат урока: Прочитать текстовый файл построчно, сохранить пустые строки и пробелы, отличить текст от байтов и корректно обработать пустой файл.

01. Проблемная ситуация

0–5 минут. Учебный файл `input.txt` содержит `oak`, пустую строку, затем `elm` без перевода строки в конце. Запишите ожидаемые номера строк. Сравните его с совершенно пустым файлом.

02. Понятия и разобранный пример

5–14 минут: 4 минуты на понятия, 3 на пример, 2 на объяснение способа партнёру.

Имя, режим и кодировка — разные свойства

Файл хранит байты. В текстовом режиме Python декодирует их в `str` с указанной кодировкой; в двоичном режиме возвращает `bytes` без декодирования. Расширение `.txt` само не гарантирует формат. В этом уроке текстовые файлы сохранены в UTF-8 без BOM. Путь `input.txt` отсчитывается от текущей рабочей папки запуска, а не обязательно от места хранения программы.

`with open(..., encoding="utf-8")` закрывает файл после выхода из блока. Цикл `for raw in f` читает строки, включая последнюю без завершающего перевода строки. `rstrip("\n")` удаляет только переводы строки справа, а `strip()` удалил бы ещё значимые пробелы. В обычном текстовом режиме CRLF при чтении преобразуется в `\n`.

Пустая строка внутри файла читается как `"\n"`, после удаления окончания получается `""`. Пустой файл не даёт ни одной итерации. Ошибки отсутствия файла и кодировки здесь видимы как исключения; не следует объявлять их пустым успешным результатом. Их обработке посвящён урок 3. Данные короткие: до 8 строк по 40 символов.

Отступы набирайте обычными пробелами: 4 на уровень. При копировании с веб-страницы замените неразрывные пробелы обычными; полный код с обычными пробелами есть в PDF.

Создайте в отдельной папке `input.txt` с точным текстом `oak\n\nelm`: запись `\n` здесь означает реальный переход на новую строку, не два печатных символа. Получаются строки 1: `"oak"`, 2: `""`, 3: `"elm"`.

`print(repr(line))` делает пустоту и пробелы видимыми.

```
with open("input.txt", "r", encoding="utf-8") as f:
    rows = [raw.rstrip("\n") for raw in f]
for i, line in enumerate(rows, 1):
    print(i, repr(line))
print("count=", len(rows), sep="")
```

Файл	<code>oak + LF + LF + elm</code>
Строка 1	<code>raw: oak\n → oak</code>
Строка 2	<code>raw: \n → пустая строка</code>
Строка 3	<code>raw: elm → elm; затем конец файла</code>

LF обозначает перевод строки. Пустая запись существует, а конец файла новой записью не является.



03. Практика 1. Проверяем материал

Построчное чтение сохраняет записи

14–25 минут. По очереди сохраните пять вариантов input.txt в UTF-8 и запустите один и тот же исходник. В таблице \n обозначает Enter; кавычки в файл не входят.

Содержимое input.txt	Строки после удаления окончаний
oak\n\nelm	["oak", "", "elm"]; count=3
Пустой файл, 0 байт	[]; count=0
\n	[""]; count=1
a \nb\n	[" a ", "b"]; count=2
oak\nelm\n	["oak", "elm"]; count=2

2 минуты — прогноз; 4 — запуск; 3 — замените `rstrip("\n")` на `strip()` и проверьте вариант с пробелами; 2 — восстановите код. Критерии: число строк, пустота, пробелы, последняя строка без LF. На бумаге составьте тот же журнал без заявления о запуске.

04. Практика 2. Перенос способа

Те же данные в двоичном режиме

25–33 минуты. Для исходного файла oak\n\nelm прочитайте байты. За 2 минуты определите длину и коды, за 4 запустите двоичное чтение, за 2 сравните с файлом из одной буквы Я в UTF-8 без BOM и без LF. Критерии: bytes вместо str, длина в байтах, отличие от числа символов.

05. Обсуждение и границы вывода

33–40 минут. За 2 минуты сопоставьте текст и байты, за 3 объясните, почему переименование image.png в image.txt не превращает картинку в текст, за 2 сформулируйте, какие сведения нужны для повторения чтения.

06. Индивидуальная проверка

40–45 минут. Для файла x\n\n без кавычек назовите число строк, их содержимое, число байтов при LF и результат после `strip()` для отдельной строки " x ". Четыре ответа — четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

`strip()` превратит " а " в "а", то есть изменит содержимое. Завершающий LF в `oak\nelm\n` не создаёт третью строку. Один LF — одна пустая строка. У пустого файла нет записей. Все пять ожидаемых результатов заданы таблицей.

Практика 2: разбор и критерии

Для исходного файла без завершающего LF: 8 байт [111,97,107,10,10,101,108,109]. Для Я: два байта [208,175], но один символ в декодированной строке. В двоичном режиме CRLF не преобразуется; здесь исходная серия задана с LF. Полный отдельный исходник:

```
with open("input.txt", "rb") as f:
    data = f.read()
print(type(data).__name__)
print(len(data))
print(list(data))
```

Обсуждение: возможный вывод

Нужны точные байты или текст с известной кодировкой, окончания строк, путь и режим открытия. Имя и расширение не меняют данные. Число байтов может отличаться от числа символов. Пустой файл и ошибка открытия — разные состояния.

Индивидуальная проверка: ответ и критерии

2 строки: "х" и ""; 3 байта [120,10,10]; `strip()` даст "х" и удалит два значимых пробела. По баллу за пункт.

Учителю: подготовка и план на 45 минут

Подготовьте компьютер, редактор, папку с правом записи, распечатку ученической части PDF, бумагу и ручку. Python 3.11+ и IDLE или редактор; запуск `python lesson.py` (либо `python3 lesson.py`). Внешние библиотеки не нужны. Установку и проверку запуска учитель выполняет заранее. Все программы, данные и ожидаемые ответы даны; регистрация детей и интернет на занятии не нужны.

Уровень — начинающий программист, 9 класс. Предпосылки: строки, циклы `for`, `enumerate`, списки. В паре автор прогнозирует и вводит код, проверяющий ведёт журнал, затем роли меняются. Без компьютера выполните трассировку полного кода на бумаге и сопоставьте с готовыми результатами; учитель демонстрирует подготовленный запуск. Бумажный вариант не считается фактическим запуском ученика. Перед практикой учитель выдаёт готовый файл разобранного примера, чтобы дети тратили время на изменение и проверку, а не длинное перепечатывание. Матрица программы предназначена учителю.

Минуты	Работа	Результат
0–5	Проблема	Вопрос и первое решение
5–14	Понятия и пример	Способ проверки
14–25	Практика 1	Готовый продукт и проверка
25–33	Практика 2	Перенос способа
33–40	Обсуждение	Аргумент и ограничение
40–45	Индивидуальная проверка	Самостоятельный ответ



07. Соответствие исходной программе

Пункт программы	Действие ученика	Проверяемый результат
Файлы	Прочитать текстовый файл построчно, сохранить пустые строки и пробелы, отличить текст от байтов и корректно обработать пустой файл.	Исходник, журнал тестов и объяснение результата
Типы файлов; чтение, запись и обработка данных	Те же данные в двоичном режиме	Самостоятельная доработка, ключ и индивидуальная проверка

Источники и права на изображения

Исходная программа курса

<https://schoolotzyv.ru/blog/python-cpp-9-klass-programma>

Python: чтение и запись файлов

<https://docs.python.org/3/tutorial/inputoutput.html#reading-and-writing-files>

Тексты, учебные данные, реплики, программы и схемы подготовлены для уроков портала «Школы России». Числовые серии — учебные модели, не результаты проведённых классом опытов. Схемы авторские, сторонние фотографии не используются. Методические ссылки приведены для учителя; ученик получает все материалы здесь. Проверка первичных источников: 18.10.2026.