Занятие 10

Работа с файлами

Что напечатает программа?

```
dct = dict.fromkeys([0,-1,1,0,-1,1,0,-1,1],0)
print(len(dct))
print(sorted(dct, key = lambda x:-abs(x)))
#-------------------------
tcid = {1:{1:{1:{1:1111}}}}
print(len(tcid))
print(tcid[1])
print(tcid[1][1][1][1])
```

Задача 9-1

Дан генетический код ДНК (строка, состоящая из букв G, C, T, A)

Постройте РНК, используя принцип замены букв:

- G → C;
- C → G;
- T → A;
- A → T

Напишите функцию, которая на вход получает ДНК, и возвращает РНК, для этого постройте словарь для замены букв.

Например:

Вход:GGCTAA Результат: CCGATT

Задача 9-2

Напишите программу, которая определяет и печатает «похожие» слова.

Слово называется **похожим** на другое слово, если его гласные буквы находятся там же, где находятся гласные буквы другого слова, например: дорога и пароход - похожие слова (гласные буквы на втором, четвертом и шестом местах), станок и прыжок - похожие слова, питон и удав непохожие слова. Считаем, что в русском языке 10 гласных букв (а, у, о, ы, и, э, я, ю, ё, е).

Ввод: x –первое слово, например, питон.

n – количество слов для сравнения, например 6.

Дальше вводятся 6 слов, например: поросенок, титан, итог, лавка, погост, кино.

Вывод - слова, **похожие** на питон:

титан, погост, кино

Задание 9-3

Произвести частотный анализ текста.

Сосчитать с помощью словаря и функции get сколько раз встречается каждый символ в тексте (включая буквы, цифры и служебные символы, включая пробелы), не учитывая регистр.

Отсортировать по убыванию и напечатать первые 10 символов, и их частоты. При равенстве частот отсортировать символы в алфавитном порядке

Например, текст «Мама мыла раму»:

а – 4

м – 4

л – 1

И т.д.

Работа с файлами

TXT

.txt — это формат файлов, который содержит текст, упорядоченный по строкам.

Текстовые файлы отличаются от двоичных файлов, содержащих данные, не предназначенные для интерпретирования в качестве текста (закодированный звук или изображение).

.py – это тоже текстовые файлы)

Что мы можем делать с файлом?

- Открыть
- Прочитать
- Дописать
- Переписать
- Закрыть!!!

Открытие файла

Прежде, чем работать с файлом, его надо открыть.

Для этой задачи есть встроенная функция open:

```
f = open("test.txt",  encoding="utf-8")
```

Результатом работы функция open возвращает специальный объект, который позволяет работать с файлом (файловый дескриптор)

Создайте в PyCharm текстовый файл test.txt, введите туда 4-5 строк:

First string

Second string

Третья строка

Четвертая строка

Синтаксис функции open()

fp = open(**file, mode='r',** buffering=-1, **encoding**=None, errors=None, newline=None, closefd=True, opener=None)

Параметры:

- **file** - абсолютное или относительное значение пути к файлу или файловый дескриптор открываемого файла.
- **mode** - необязательно, строка, которая указывает режим, в котором открывается файл. По умолчанию 'r'.
- **buffering** - необязательно, целое число, используемое для установки политики буферизации.
- **encoding** - необязательно, кодировка, используемая для декодирования или кодирования файла.
- **errors** - необязательно, строка, которая указывает, как должны обрабатываться ошибки кодирования и декодирования. Не используется в бинарном режиме
- **newline** - необязательно, режим перевода строк. Варианты: None, '\n', '\r' и '\r\n'. Следует использовать только для текстовых файлов.
- **closefd** - необязательно, bool, флаг закрытия файлового дескриптора.
- **opener** - необязательно, пользовательский объект, возвращающий открытый дескриптор файла.

Имя файла, какой файл, что делать

У функции **open()** много параметров, нам пока важны 3 аргумента:

Первый, это имя файла.

Путь к файлу может быть относительным или абсолютным.

Второй аргумент - это режим, mode, в котором мы будем открывать файл. Режим обычно состоит из двух букв, первой является тип файла - текстовый или бинарный, в котором мы хотим открыть файл, а второй указывает, что именно мы хотим сделать с файлом.

Третий аргумент – кодировка файла

Первая буква режима:

"b" - открытие в двоичном режиме.
"t" - открытие в текстовом режиме (является значением по умолчанию).

Второй буква режима:

"r" - открытие на чтение (является значением по умолчанию).
"w" - открытие на запись, содержимое файла удаляется, если файла не существует, создается новый.
"x" - эксклюзивное создание (открытие на запись), бросается исключение FileExistsError, если файл уже существует.
"a" - открытие на дозапись, информация добавляется в конец файла.
"+" - открытие на чтение и запись

Примеры

Режим "w" открывает файл только для записи.

Перезаписывает файл, если файл существует.

Если файл не существует, создает новый файл для записи.

f = open("test.txt", mode="w" **encoding=**"utf-8")

Открывает файл в бинарном режиме для записи и чтения.

Перезаписывает существующий файл, если файл существует.

Если файл не существует, создается новый файл для чтения и записи.

f = open("music.mp3", mode="wb+")

По всем режимам см. документацию open()

Закрыть файл

После того как вы сделали всю необходимую работу с файлом - его следует закрыть.

```
f = open("test.txt",  encoding="utf-8")
# какие-то действия
f.close()
```

Чтение файла

Теперь мы хотим прочитать из него информацию.

Для этого есть несколько способов, но большого интереса заслуживают лишь два из них.

Первый - метод **read,** читающий весь файл целиком, если был вызван без аргументов, и n символов, если был вызван с аргументом (целым числом n).

```
f = open("test.txt", "r")

print(f.read(5))
print(f.read(5))
print(f.read(4))
print(f.read())

f.close()
```

Фунция readlines()

Файлы можно читать не только целиком или посимвольно, но и построчно.

Для этого у объекта файла есть метод readlines, который возвращает список из строк файла.

```
f = open("test.txt", "rt")
print(f.readlines())
f.close()
```

Обратите внимания, что каждая строка в списке имеет в конце символ `\n`.

Задание

Прочитайте содержимое файла с помощью функции readlines()

Присвойте ее результат переменной lst

Напечатайте пронумерованный список строк.

Постарайтесь избавиться от лишних пустых строк.

Фунция readline()

Функция **readlines()** загружает все строки целиком и хранит их в оперативной памяти,
что может быть очень накладно, если файл занимает много места на жёстком диске.

А можно читать файл построчно с помощью функции **readline()**

```
f = open("test.txt", "rt")
print(f.readline())
print(f.readline())
f.close()
```

Также обратите внимание, что возвращённые строки имеют в конце символ `\n`.

Задание

Прочитайте содержимое файла с помощью функции readline()

Напечатайте пронумерованный список строк.

Итерирование файла

Ещё один способ прочитать файл построчно - использовать файл как итератор. Такой вариант считается самым оптимизированным

```
f = open("test.txt")
for line in f:
    print(line)

f.close()
```

Запись

Теперь рассмотрим запись в файл.

Для того чтобы можно было записывать информацию в файл, нужно открыть файл в режиме записи.

Для записи в файл используется функция write.

При открытии файла на запись из него полностью удаляется предыдущая информация.

```
fout = open("test.txt", "wt")
fout.write("New string")
fout.write("Another string")
fout.close()
```

Если вы откроете файл в текстовом редакторе, то увидите, что строки "New string" и "Another string" склеились.

Так произошло, потому что между ними нет символа перевода строки.

writelines()

Также в файлах, открытых на запись, есть метод writelines, который позволяет записать несколько строк в файл

```
f = open("text.txt", "wt")
lines = [
    "New string\n",
    "Another string\n",
]
f.writelines(lines)
f.close()
```

Общий алгоритм работы с текстовыми файлами

1. Открыть файл
2. Считать информацию в какую-то конструкция Python, словарь, список, кортеж и т.д.
3. Обработать информацию, напечатать результат, ввести результат в этот или другой файл
4. Закрыть файл

Задание

Прочитать информацию из файла test.txt.

Записать в файл test1.txt, только те строки, которые содержат цифры.

Например:

Hello!

This is the 1st letter.

Bye

Результат: This is the 1st letter.

Задание

Откройте текстовый файл.

Каждый второй знак этого файла перенесите в другой файл.

print(..., file = f)

Можно использовать print(), если указать **file = файловый объект**

```
f = open('text.txt', 'w', encoding = 'utf-8')
print(*objects, sep=' ', end='\n', file=f)
```

По умолчанию стандартный вывод на экран, а можно указать file = f

Напишите программу, которая печатает в текстовый файл строчки из числа и его квадрата, т.е.

0 0

1 1

2 4

3 9 и т.д.

```
fi = open('file.txt', 'w', encoding = 'utf-8')
for i in range(5):
    print(i, i * i, file = fi)
```

Дозапись

Если нужно записать в конец файла какую-то информацию, то можно сделать это, открыв файл в режиме дозаписи.

Все методы, доступные в режиме записи также доступны в режиме дозаписи.

```
f = open("text.txt", "at")

f.write("First string\n")
lines = [
    "Second string\n",
    "Third string\n",
]
f.writelines(lines)

f.close()
```

Давайте проверим это

Запись с возможностью чтения

Иногда нужно открыть файл с возможностью и записи, и чтения.

В Python есть два режима:

* Запись с возможностью чтения ("w+")
* Чтение с возможностью записи ("r+")

На первый взгляд кажется, что они ничем не отличаются, но это не так.

При открытии файла на запись (w+) с возможностью чтения из файла полностью удаляется вся информация.

with … as … - менеджер контекста

```
with open('file.txt', 'r', encoding = 'utf-8') as fi:
        print(fi.readlines())
# файл закрывается автоматически
```

```
# откройте файл с помощью with и напечатайте его содержимое
 print(fi.readlines() )
```

Задание

Прочитать строки текста из одного файла,
отсортировать слова внутри строки по возрастанию и
записать обновленные строки в другой файл.

Модуль openpyxl

Модуль openpyxl - это библиотека Python для чтения/записи форматов Office Open XML (файлов Excel 2010) с расширениями xlsx/xlsm/xltx/xltm.

Не входит в стандартную библиотеку, необходимо его инсталлировать

IDLE: pip install openpyxl

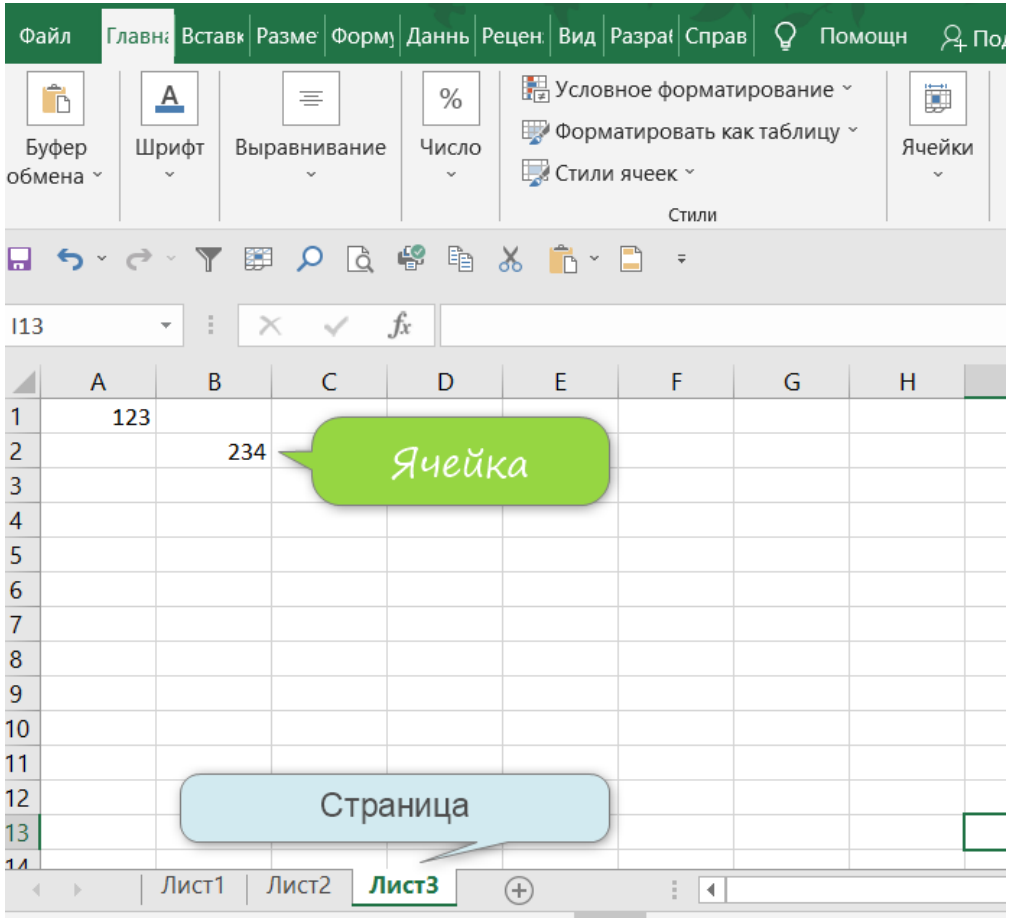
PyCharm: File/Settings/Project…/Python Interpreter/ + / набрать openpyxl / Install Package

Попробуйте в консоли import openpyxl

Excel файл - книга

Функции openpyxl:

- Книга
- Страницы
- Ячейки



Создание книги Excel (workbook)

```
import openpyxl
from openpyxl import Workbook
wb = Workbook()    # создаем экземпляр класса Workbook
wb.save('test.xlsx') # сразу его записываем пустой
```

Создайте файл

Рабочие страницы

```
import openpyxl
wb = openpyxl.load_workbook(‘test.xlsx’) # Открываем книгу
print(wb.sheetnames)                     # Список листов

ws = wb.active                           # Кто активный рабочий лист?
print(ws.title)                          # Каков его title?

wb.create_sheet('New’)                   # Создаем новый лист
ws3 = wb["New"]                          # Другой лист
print(ws3)                               # Посмотрим
```

Рабочие страницы

```
print(wb.sheetnames)                # список листов
wb.active = wb[‘New’]               # назначим другой активный лист
wb.remove(ws)                       # удаление рабочего листа

print(wb.sheetnames)
wb.save('test.xlsx’)                # сохраняем книгу
```

Доступ к ячейкам - cell

```
sheet['A1'].value                # Значение, которое хранится в ячейке (Retrieve the value of a certain cell)

sheet['A1'].value = 123          # Поместить число в ячейку

c = sheet['B2’]                  # выбрать ячейку  (Select element 'B2' of your sheet)

c.row                            # номер строки  (Retrieve the row number of your element)

c.column                         # номер колонки (Retrieve the column letter of your element)

c.coordinate                     # координаты ячейки
```

#Запишите в ячейку A1 число 100, в ячейку B2 – 200, сложите их значения и результат запишите в C3

Работа с ячейками - cell

```
# напечатать значение ячейки по номеру строки и колонки
sheet.cell(row=1, column=2).value

# присвоить значение ячейке ряд = 5, колонка = 3
sheet.cell(row = 5, column = 3) = 12345



#  печатаем колонку номер 2
for i in range(1, 4):
    print(i, sheet.cell(row=i, column=2).value)

#Выполните
```

max_row max_col

#максимальная строка, где есть информация

sheet.max_row

максимальная колонка, где есть информация

sheet.max_col

Как напечатать все ячейки листа

```
for i in range(ws.max_row):
    for j in range(ws.max_column):
        print(i + 1, j + 1, ws.cell(row = i + 1, column = j + 1).value)
```

Выполните этот код

Как прочитать данные одной колонки?

Как записать данные в колонку?

Основные функции

##		Команда	Что делает
1		import openpyxl	Импортирует модуль openpyxl
2	B	from openpyxl import Workbook	Загрузка класса Workbook
3	B	wb = Workbook()	Создаем рабочую книгу
4	B	wb.save('test.xlsx')	Сохраняем файл
5	B	wb = openpyxl.load_workbook("test.xlsx")	Загружаем файл
6	S	ws = wb.active	Определяем активный рабочий лист
7	S	wb.active = ws	Переопределяем активный рабочий лис
8	S	ws.title	Имя листа
9	B	wb.sheetnames	Список листов книги
10	S	wb.create_sheet("Newsheet")	Создание нового листа
11	S	wb.remove(ws)	Удаление листа
12	C	ws.['A1'].value	Значение ячейки (cell)
13	C	c = ws['B2']	Присвоить с ячейку (не значение!!!)
14	C	c.row	Номер строки
15	C	c.column	Номер колонки
16	C	c.coordinate	Координаты ячейки ('A1')
17	C	ws.cell(row = 1, column = 2).value	Значение ячейки по номеру строки и колонки (нумерация с 1)
18	S	ws.max_row	Максимальная строка с данными
19	S	ws.max_column	Максимальная колонка с данными

Общий алгоритм работы с эксельными файлами

1. Открываем книгу (wb) (или уже существующий, или делаем новый файл)
2. Выбираем лист (ws) (или создаем новый, или делаем активным по имени)
3. **Считываем данные из ячеек и помещаем их в список или в словарь или куда-то еще**
4. Делаем нужные действия (вычисляем, сортируем, что-то еще), т.е. храним данные в Excel, а обрабатываем их в Python
5. Записываем их в какую-то страницу ws
6. Сохраняем wb

Задание

Создайте Эксель файл

Откройте его с помощью openpyxl

Напечатайте список его страниц, для каждой страницы напечатайте максимальное количество строк, столбцов и занятых ячеек

Задание

Дан Эксельный файл.

В нем список людей, у каждого количество отработанных дней и дневную ставку.

Сосчитайте и напечатайте заработанные деньги каждым и итоговую сумму.

Задание

Дан эксельный файл, содержащий список людей и их премию.

Загрузите этот список, отсортируйте его по алфавиту.

Создайте еще одну страницу в файле и запишите туда отсортированный список с премиями.

Задача 10-1

Есть текстовый файл 'test1.txt'. Надо прочитать содержимое этого файла и создать файл 'test2.txt'. В нем должны быть строчки из первого файла, но в обратном порядке. В каждой строке должен быть поменян порядок слов на противоположный.

Например, исходный файл:

Мой дядя самых честных правил

Когда не в шутку занемог

Результат:

занемог шутку в не Когда

правил честных самых дядя Мой

Задача 10-2

Дан эксельный файл. На странице находится список сотрудников и их ЗП.

Надо создать еще одну страницу в этом файле и поместить туда отсортированный список фамилий и их ЗП, а в последней строчке поместить слово ИТОГО: и сумму всех ЗП. Сортировка по убыванию ЗП сотрудников.

Например, исходная страница файла:

Сидоров 100

Петров 200

Иванов 300

Результат:

Иванов 300

Петров 200

Сидоров 100

ИТОГО: 600

Задача 10-3

Дан исходный эксельный файл со списком людей и их ЗП.
Следует создать еще одну страницу со статистическими данными исходного списка.
Например, исходная страница:

Сидоров	**100**
Петров	**200**
Иванов	**300**
Федоров	**1000**

Результат:

Максимальное значение	**1000**
Минимальное значение	**100**
Сумма	**1600**
Среднее арифметическое	**400**
Медиана	**250**