ファイル操作

write()メソッド

- 普通の文字列のみの場合1行書き込む
- エスケープシーケンスの `\n` で、改行を書き込める
- 戻り値があり、書き込んだ文字数を返す
- "w"なので、2回やっても、4行ではなく3行にならない（↓のコードの場合の話です）
    - 既存の内容は削除し上書きする

```
# f(ファイルオブジェクト).write()

f = open("hello_1.txt", "w")  # "w" ➡ "wt"
how_many = f.write("Hello, world!\n")  # 改行する場合は、\nが必要である
f.write("Hello, world!")
f.close()  # close()を忘れないこと！
"""
`f.close()`を実行しないと、完全に書き込みが終了せず、
メモリの解放、ファイルへのアクセスなどの問題が起きる。
with文では、close()メソッドは書かなく、推奨されるやり方になっている
"""

print(how_many)  # 何文字であるかを返す、\nも1文字である、14文字

"""
(hello_1.txtの出力)
Hello, world!
Hello, world!

(コンソール画面への出力)
14
"""
```

- "w"を"a"にすると、追加で書き込まれることを確認する
    - mode="at"の場合、既存の内容に追加する

```
# ファイルオブジェクト.write()
# mode="at"で、このコードを2回実行する

f = open("hello_1.txt", "a")  # "a" ➡ "at"
f.write("Hello, world!\n")  # 改行する場合は、\nが必要である
f.write("Hello, world!")
f.close()  # close()を忘れないこと！

"""
(hello_1.txtの出力: 2回実行した結果)
```

```
Hello, world!
Hello, world!Hello, world!（2回目のwrite()は、改行文字を書き込んでいない）
Hello, world!
"""
```

writelines()メソッド + with文 ← with文は、また別の話です

- writelines()メソッド：文字列リストを書き込む
- with文：推奨される方法、close()メソッドは使わず書ける
  - コロン(:)と、その後のインデントを忘れないようにしよう

```
# f(ファイルオブジェクト).writelines()
# with文で書いて、`f.close()` を省略する

# as f → as(～として) f → ファイルオブジェクトとして
# with open(～) as f: fとして～をオープンする
# ★超重要★：日本語が文字化けして書き込まれる時は、
#   with文に、`encoding="utf-8"`を追加しておこう！
with open("hello_2.txt", "w", encoding="utf-8") as f:
    f.writelines([
        "A: このTシャツ、色が素敵だね。試着してみたら？\n",
        "B: うーん、でも少し高いかな。セールになるまで待った方がいいかも。\n",
        "A: でも人気の商品だから、サイズがなくなるかもよ。\n",
        "B: それもそうだね。じゃあ、試着だけでもしてみようかな。"
    ])

""" (hello_2.txt)
A: このTシャツ、色が素敵だね。試着してみたら？
B: うーん、でも少し高いかな。セールになるまで待った方がいいかも。
A: でも人気の商品だから、サイズがなくなるかもよ。
B: それもそうだね。じゃあ、試着だけでもしてみようかな。
"""
```

read()メソッド

- EOF(end of file)まで、一つの文字列として、読み込んで表示する
- 最大の文字数を、int型で指定出来る ↓： ten_letters = f.read(10)

```
# f.read()

# hello_3.txt
"""
abcdefghijklmnopqrstuvwxyz0123456789.
```

```
9876543210zyxwvutsrqponmlkjihgfedcba.

the end
"""

with open("hello_3.txt", "r") as f:  # "r" ➡ mode="rt"
    one_string = f.read()
    print(one_string)

# 文字列としては1つの文字列である
# 改行文字も含まれているので、複数行になる
"""
abcdefghijklmnopqrstuvwxyz0123456789.

9876543210zyxwvutsrqponmlkjihgfedcba.

the end
"""

f = open("hello_3.txt", "r")
ten_letters = f.read(10) # 10：位置引数（キーワード引数ではないということ）
print(f"最初の10文字：{ten_letters}")
f.close()  # with文ではないので、必要である

"""
最初の10文字：abcdefghij
"""
```

readline()メソッド

- 1行を読み込む

```
""" hello_4.txt
Hello!
How are you?

Fine, thanks.
And, you?
"""

f = open("hello_4.txt")
"""f = open("hello_4.txt", mode="rt")"""  # 省略無しで書いた場合

while True:
    line = f.readline()

    if line == "":  # `if not line:` でもOK（""はFalsy）
        # ファイル終端で""を返すのは、readline()のファイル処理の仕様である
        print("\n ---\nNow, line == ''")
```

```
        break
    print(line, end="")

f.close()

""" print()関数の出力
Hello!
How are you?

Fine, thanks.
And, you?
 ---
Now, line == ''
"""
```

readlines()メソッド

- read()メソッドが、単一の文字列を返すのに対して、文字列のリストを返す
- デフォルトでは、すべての行を返す
  - 行単位で、文字列リストの要素になる
- ~~最大行数を指定できる~~（最大バイト数であったが、行数を決めたいなら、with文＋スライス が便利です）

```
# hello_5.txt

# 今日YouTubeで、関先生の音読の話を聞きました。
#
# イントロの段階では私の考えと同じでした。
# その後、私はリプロダクションが良いと思っていたのですが。
# 長文に関しては、まずは音読の方がいいのかなと考えを変えました。

f = open("hello_5.txt", "r", encoding="utf-8")  # r: mode="rt"
lines = f.readlines()
print(lines)

print()
print("●リストlinesを、for文ですべての行を取得する")
print()

for line in lines:
    print(line, end="")  # print()関数は改行付きで出力する

f.close()

# 実行結果
# 最初のリストの取得は一部省略して表示します:

# ['今日、～。\n', '\n', 'イントロの～。\n', 'その後、～。\n', '長文に～。']
```

```
# ●リストlinesを、for文ですべての行を取得する
#
# 今日YouTubeで、関先生の音読の話を聞きました。
#
# イントロの段階では私の考えと同じでした。
# その後、私はリプロダクションが良いと思っていたのですが。
# 長文に関しては、まずは音読の方がいいのかなと考えを変えました。
```

ファイルオブジェクトから、直接1行ずつ取り出す

- 実は、ファイルオブジェクトから、for文で直接取り出せる
  - テキストファイルオブジェクトは、イテラブルなオブジェクトなのである
- さらに、高速＋効率的 な、方法である！　高速+効率的 → 適切な（この方が分かりやすい）

```
""" (lorem.txt: `Lorerm Ipusm`は、ダミー文字列が使える有名なWebサイト)
Lorem Ipsum is simply dummy text of the printing and typesetting industry.


Lorem Ipsum has been the industry's standard dummy text.
"""

f = open('lorem.txt', mode="rt")  # "rt"：read,text（デフォルト）

for line in f:
    print(line, end='')

f.close()  # クローズ

"""
Lorem Ipsum is simply dummy text of the printing and typesetting industry.


Lorem Ipsum has been the industry's standard dummy text.
"""
```

END

P.S.
現代のPythonプログラミングでは、with文の使用は「推奨」というよりも「標準的な慣行」、あるいは「必須」と言っても過言ではありません。
（with文を使わないコードは、close()の意味を知ることが出来るなど、少しでも内部のコードを知れるので、学習段階では有効である）