Revision - File IO

1. Text File

read()

The `read()` function reads all data from a file handler.

Example: Read and print out all text in file `./data/quote.txt`.

In [1]:

```
with open('./data/quote.txt') as f:
    t = f.read()
    print(repr(t))
```

```
'Give a man a program, frustrate him for a day.\nTeach a man to program, fru
strate him for a lifetime.\n'
```

write()

The `write()` function writes data into a text file.

Example: Convert text in `./data/quote.txt` to upper case, and save it in file `./data/quote_upper.txt`.

In [2]:

```
with open('./data/quote_upper.txt', 'wt') as f:
    t = t.upper()
    f.write(t)
```

In [3]:

```
!notepad data/quote_upper.txt
```

writelines()

To write a list of strings to a file, method `writelines()` can be used.

- Note: NO new line character `\n` will be added automatically.
- Append `\n` to write each item in a line.

In [4]:

```python
data = ['A{:05}\n'.format(i) for i in range(10)]
print(repr(data))

with open('./data/test.txt', 'w') as f:
    f.writelines(data)
```

```
['A00000\n', 'A00001\n', 'A00002\n', 'A00003\n', 'A00004\n', 'A00005\n', 'A0
0006\n', 'A00007\n', 'A00008\n', 'A00009\n']
```

In [5]:

```
!notepad data/test.txt
```

readlines()

The `readlines()` function returns a list, where each item contains a line.

- Note: NO character will be removed automatically, e.g. there may be a new line character `\n` at the end of each line.
- Use `strip()` method to remove leading and trailing white spaces.

In [6]:

```python
with open('./data/test.txt') as f:
    data = f.readlines()

for i in data:
    i = i.strip()
    print(i, end=' ')
```

```
A00000 A00001 A00002 A00003 A00004 A00005 A00006 A00007 A00008 A00009
```

In [7]:

```python
with open('./data/test.txt') as f:
    for i in f:
        i = i.strip()
        print(i, end=' ')
```

```
A00000 A00001 A00002 A00003 A00004 A00005 A00006 A00007 A00008 A00009
```

2. CSV File

Using `csv.reader`

After opening a CSV file, create a csv.reader object which returns a iterable object to process CSV data.

- Each record is represented as a list.
- All fields are string type.

Note:

- How to skip a header line?
- How to save data into a list?

In [8]:

```python
import csv
with open('./data/olympics-medals-sample.csv') as f:
    reader = csv.reader(f)
    header = next(reader)
    data = [row for row in reader]

print(header)
print(data)
```

```
['NOC', 'Country', 'Total', 'Medal']
[['USA', 'United, States', '2088', 'Gold'], ['URS', 'Soviet Union', '838',
'Gold'], ['GBR', 'United Kingdom', '498', 'Gold'], ['FRA', 'France', '378',
'Gold'], ['GER', 'Germany', '407', 'Gold'], ['AUS', 'Australia', '293', 'Gol
d']]
```

Using csv.writer()

The `csv.writer()` function takes only 1 required parameter, the file object, and returns a csv writer.

- Note: You may need to add `newline=''` parameter when create a file handler to avoid extra empty line between rows.
- Its `writerow()` method writes a list as a row to the csv file.

In [9]:

```python
import csv

with open('./data/olympics-medals-sample-1.csv', 'w', newline='') as f:
    writer = csv.writer(f)
    for row in data:
        writer.writerow(row)
```

In [10]:

```
!notepad data/olympics-medals-sample-1.csv
```

- Its `writerows()` method writes a nested-list as multiple rows to the csv file.

In [11]:

```python
import csv

with open('./data/olympics-medals-sample-2.csv', 'w', newline='') as f:
    writer = csv.writer(f)
    writer.writerows(data)
```

In [12]:

```
!notepad data/olympics-medals-sample-2.csv
```