

[Get started](#)[Open in app](#)

# Mario Gunawan

[Follow](#)

1 Follower

[About](#)

## Tutorial 1 : Read dan Save Data dari CSV tanpa library di python



Mario Gunawan 14 hours ago · 15 min read

Pertama tama, kalau kalian ada yg read limit, (udh 3x read), copy url page ini, buka incognito tab, paste disitu. nanti read limitnya ilang (semoga medium ga notis :))

### Prerequisite

1. Mengerti tentang printing, if else, loop, dan deklarasi variabel di python
2. Ngerti basic datatype di python (string, boolean, integer, float) kalo kurang ngerti, ada artikel singkat: <https://realpython.com/python-data-types/>
3. Ngerti cara run python file dari terminal
4. Masih semangat hidup (dan ngerjain tubes pastinya!)

### Daftar konten

1. Setup proyek
2. Membaca file di python
3. Mengubah string baris menjadi array of data
4. Modifikasi dan menyimpan data
5. (optional) Menerima user input dan mengubah data

## 6. (optional) Merapihkan kode

### Sebelum Memulai Tutorial:

1. Bila tidak memenuhi prerequisite tutorial ini, saya sarankan belajar terlebih dahulu konsep konsep diatas dan latihan mengerjakan praktikum sebelumnya
2. Dalam tutorial ini, akan diberikan kode
3. **Jangan copas kode, pahami kode terlebih dahulu baru ketikan apa yang baru kamu pahami.** Ingat, mungkin bila copas kode anda akan jauh lebih cepat dibanding teman2 anda yang mengikuti ini, tapi nanti bila ada bug atau ada perubahan, anda tidak akan bisa membereskannya karena anda tidak mengerti kode secara konseptual

Jadi, kalau sudah siap langsung kita mulai aja!

## 1. Setup Proyek

Untuk memulai tutorial ini, ikuti langkah langkah dibawah ini:

1. Buka IDE\* favoritmu
2. Buat folder baru, misalnya namanya “python\_koreandrama”
3. Buka folder tersebut di IDE\* favoritmu

\*untuk selanjutnya, istilah IDE akan sering dipakai, IDE artinya text editor yang kamu gunakan sekarang (seperti geany, visual studio code, atom, dsb)

## 2. Membaca File di Python

python memiliki fungsi untuk membaca file **dengan format apapun**. yaitu dengan fungsi open(nama\_file, method). Misalnya, aku memiliki file data.csv dengan isi sebagai berikut:

data tentang korean drama dan ratingnya

(untuk save filenya, klik view raw di kanan bawah lalu copas aja kontennya)

save file tersebut dengan nama “korean\_drama.csv” tanpa tanda kutip di folder yang sudah terbuka di IDE mu.

Kemudian, buat file baru di folder yang sama dengan korean\_drama.csv, bernama read\_csv.py . Pertama tama, kita hanya akan membaca konten dari csv baris per baris dan menyimpannya ke suatu variabel. Caranya adalah:

## 1 Buka file

```
f = open("korean_drama.csv", "r")
print(f.read())
f.close()
```

- f adalah variabel untuk menyimpan data file yang kita baca
- open( **filename** , **mode** ) adalah cara untuk membuka suatu file dengan nama **filename** . Mode adalah apa yang kita ingin lakukan ke file tersebut. Ada beberapa mode yang perlu diingat:
  - r → artinya kita ingin membaca file tersebut, dengan kondisi file tersebut sudah ada. Bila file tidak ada akan error
  - w → artinya kita ingin menulis ke file tersebut
  - w+ → artinya kita selain ingin menulis, ingin membaca isi file tersebut juga, **tidak akan** memberikan error apabila file tidak ada
  - a → (append) artinya kita ingin menambahkan konten ke ujung file
- f.read() adalah cara untuk membaca keseluruhan file
- f.close() adalah cara untuk memberi tahu bahwa file tersebut tidak akan kita gunakan lagi. kenapa perlu f.close()? Karena misal program berjalan lama, file yang kita open akan dianggap tidak bisa digunakan oleh program lain bila tidak di close, maka jangan lupa untuk close() (jaga2 aja).

Bila dijalankan, outputnya adalah:



## 2 Pisahkan file jadi beberapa baris

Jadi, cara diatas bagus apabila kita ingin membaca keseluruhan file, tapi bisa lebih baik lagi apabila kita membaca file baris per baris. Ini dikarenakan kita ingin membaca data dari csv yang satu baris berarti satu set data.

```
f = open("korean_drama.csv", "r")
lines = f.readlines()
f.close()
print(lines)
```

Perbedaannya hanya dari menggunakan method `f.read()`, kita menggunakan `f.readlines()` dan menyimpannya ke variabel. Hasil output terminalnya adalah:

Dapat dilihat kalau hasilnya adalah sebuah array. Fungsi dari `f.readlines()` ini adalah memisahkan baris per baris dari suatu file dan menyimpannya ke array.

Misalnya anda perhatikan, ada `\n` di akhir setiap elemen kecuali elemen terakhir. Bisa diingat lagi, `\n` artinya line baru, sehingga di akhir setiap baris akan ada elemen ini.

Mari kita ubah kode supaya mencetak seperti `f.read()`:

```
f = open("korean_drama.csv", "r")
lines = f.readlines()
f.close()
for line in lines:
    print(line)
```

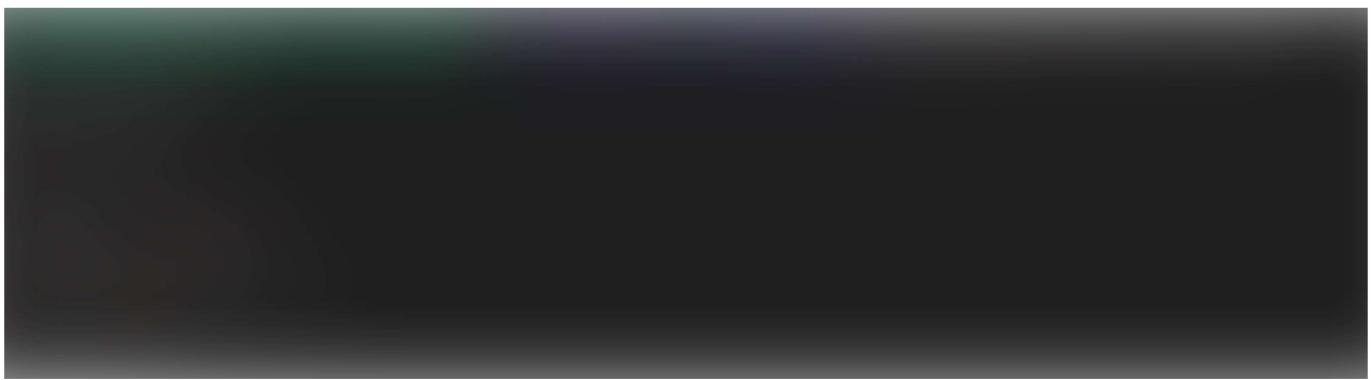
For loop diatas akan mengiterasi setiap elemen di array `lines` dengan nama elemen yang dibaca adalah “`line`”. `print(line)` akan mencetak elemen tersebut. Bila anda kurang

paham sintaks **for line in lines**, kode ini sama saja seperti

```
for i in range(len(lines)):  
    print(lines[i])
```

namun lebih singkat

Bila kode diatas dijalankan maka akan menjadi



hasil kode

yang.... bisa diterima tapi bukan yang kita inginkan. Hal ini terjadi karena fungsi print di python menambahkan line baru di akhir string. Apabila di akhir string ada \n, maka akan ke double line barunya. Ada 2 cara untuk mengatasi hal ini, yaitu menghapus karakter \n dan mengubah fungsi print untuk tidak menambahkan line.

Dalam tutorial ini, akan dibahas cara untuk menghapus karakter \n pada masing masing elemen dalam lines.

```
f = open("korean_drama.csv", "r")  
raw_lines = f.readlines()  
f.close()  
lines = [raw_line.replace("\n", "") for raw_line in raw_lines]  
for line in lines:  
    print(line)
```

2 baris yang di bold adalah perubahan program. Pertama tama, ubah nama lines menjadi raw\_lines. Ini untuk memperjelas kalau data hasil f.readlines() masih akan

diproses. Kemudian, untuk

```
lines = [raw_line.replace("\n", "") for raw_line in raw_lines]
```

saya harap ini bukan pertama kalinya anda melihat ada loop didalam suatu array di python. Namun, misalnya ini pertama kali, coba lihat kode yang lebih simpel di bawah:

```
array_of_zeros = [0 for i in range(5)]
# hasil: [0,0,0,0,0]
```

Yang ini lebih mudah dimengerti bukan? Artinya, dia menambahkan 0 sebanyak 5 kali, karena for i in range(5) akan iterasi sebanyak 5 kali. Coba lihat lagi contoh berikut:

```
array_number = [1,5,3,2]
array_number_plus_2 = [(number + 2) for number in array_number]
# hasil: [3,7,5,4]
```

Dalam operasi **[(number + 2) for number in array\_number]**, kita bisa membaginya jadi 2 part:

- **for number in array\_number**

Kita mengambil masing masing angka dalam array number sebagai variabel number

- **(number + 2)**

Kita menambahkan 2 kepada setiap number yang telah dideklarasikan

Untuk latihan, coba tebak hasil dari deklarasi array menggunakan for loop dibawah ini (misalnya masih belum tau juga, bisa tanya temen atau baca2 lagi)

```
strings = ["haha", "hehe", "hoho"]
m = [string + "!" for string in strings]

numbers = [4,1,2,3]
y = [number * number for number in numbers]
```

```
sesuatu_random = [(i % 5) for i in range(10)]
```

Anda juga bisa menggunakan cara yang intuitif( for loop biasa ), tapi saya lebih suka cara ini soalnya lebih cepet hehe

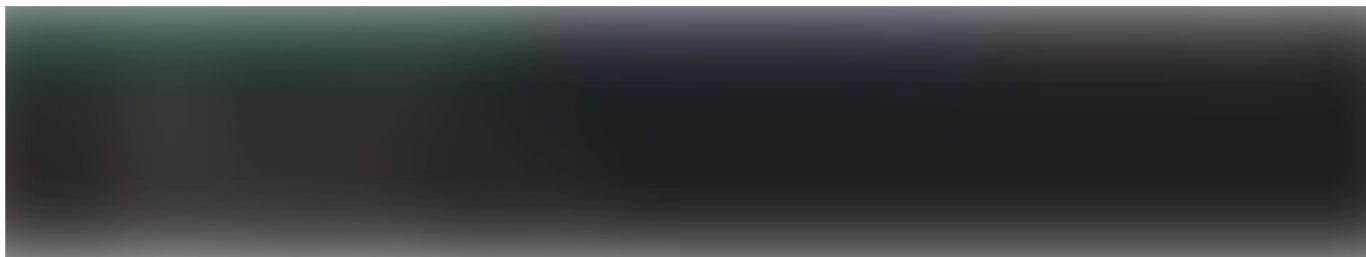
### 3. Mengubah String Baris Menjadi Array of Data

Kode terakhir telah mengubah file menjadi beberapa baris yang disimpan di variabel lines. Sekarang, tugas kita adalah memisahkan setiap baris yang masih berupa string sekarang menjadi array of data.

Untuk mengubah string menjadi array, ada banyak cara. Python telah memberikan fungsi utilitas yang sangat memudahkan dalam pengubahannya dengan fungsi `string.split( delimiter )` delimiter adalah karakter pemisah, dalam kasus ini adalah karakter koma ( , )

```
f = open("korean_drama.csv", "r")
raw_lines = f.readlines()
f.close()
lines = [raw_line.replace("\n", "") for raw_line in raw_lines]
for line in lines:
    array_of_data = line.split(",")
    print(array_of_data)
```

hasilnya adalah:



Namun, array diatas masih sedikit... kotor. Yang kita inginkan adalah array of data yang bermakna. Kita ingin supaya rating dan id keduanya berbentuk integer, bukan string, dan bisa dilihat juga ada spasi sebelum angka rating pada baris dengan id 3.

Kita harus menghandle kasus spasi karena kita tidak bisa expect datanya akan selalu bersih, kemudian baru kita ubah tipe data dari string menjadi integer. Ubah kode dibawah baris **for line in lines** menjadi:

```
raw_array_of_data = line.split(",")
array_of_data = [data.strip() for data in raw_array_of_data]
print(array_of_data)
```

- Karena data masih akan diproses, kita menggunakan istilah “raw” lagi di awal
- pemanggilan .strip() ke suatu string akan menghapus space di awal dan di akhir string tersebut.

Ketika dijalankan, output kode diatas adalah:



Yay! Sudah hampir beres membersihkan data ini. Sekarang tinggal konversi data dari tipe string ke integer. Sebelum itu, mari kita refaktor kode supaya lebih bisa dibaca:

```
f = open("korean_drama.csv", "r")
raw_lines = f.readlines()
f.close()
lines = [raw_line.replace("\n", "") for raw_line in raw_lines]

def convert_line_to_data(line):
    raw_array_of_data = line.split(",")
    array_of_data = [data.strip() for data in raw_array_of_data]
    return array_of_data

for line in lines:
    array_of_data = convert_line_to_data(line)
    print(array_of_data)
```

di kode diatas, kita membuat sebuah fungsi untuk mengonversikan line menjadi array of data. Ini ditujukan agar kode kita lebih *readable*. Bila kedepannya ada fungsi yang cukup panjang, bisa dijadikan fungsi. **Walaupun hanya dipakai satu kali, tetap worth it buat bikin fungsi supaya kodennya enak dibaca.** Ini untuk kebaikan kelompok, juga kebaikan diri sendiri (pas future kamu ngedebug)

Sekarang, diatas fungsi **convert\_line\_to\_data** dan tepat dibawah **lines = [...]**, buat fungsi baru **convert\_array\_data\_to\_real\_values** (*kalau anda punya ide nama yang lebih baik silakan diubah*) yang bertugas mengubah array data menjadi value yang sesungguhnya, misal integer, atau semacamnya. Kodennya adalah:

```
...
lines = [raw_line.replace("\n", "") for raw_line in raw_lines]

def convert_array_data_to_real_values(array_data):
    arr_cpy = array_data[:]
    for i in range(3):
        if(i == 0):
            arr_cpy[i] = int(arr_cpy[i])
        elif(i == 2):
            arr_cpy[i] = float(arr_cpy[i])
    return arr_cpy

def convert_line_to_data(line):
    ...

```

Mari kita *dissect* kode diatas satu per satu:

- **arr\_cpy = array\_data[:]**

Tujuan kode ini adalah membuat copy dari **array\_data**. Kita membuat copy supaya kita tidak langsung memodifikasi **array\_data**. Ini dilakukan supaya kita masih memiliki array data original yang tidak dimodifikasi, bila nantinya kita ingin melakukan sesuatu terhadap **array\_data** itu.

- **for i in range(3)**

3 adalah banyaknya kolom yaitu id , nama, dan rating.

- **if(i == 0):**

**arr\_cpy[i] = int(arr\_cpy[i])**

Indeks dari id adalah 0 (id adalah elemen pertama array data, dan python memulai loop dari index 0)

Maka bila `i == index_id`, ubah variabel menjadi tipe integer

- `elif(i == 2):`

```
arr_cpy[i] = float(arr_cpy[i])
```

indeks dari rating adalah 2 (elemen ketiga array data)

Maka bila `i == index_rating`, ubah variabel menjadi tipe float, karena rating bisa koma komaan

- `return arr_cpy`

```
return array yang telah dimodifikasi
```

Kemudian, dalam block `for line in lines:` ubah isinya menjadi:

```
for line in lines:  
    array_of_data = convert_line_to_data(line)  
    real_values = convert_array_data_to_real_values(array_of_data)  
    print(real_values)
```

Ketika dijalankan, anda akan mendapat error sebagai berikut:



Karena, kita memasukan baris pertama yang hanya berisi label id,name,rating.

Sehingga ketika python mencoba untuk mengonversikan `int('id')`, akan terjadi error

Oleh karena itu, kita harus menghapus baris pertama dari variabel `lines`. Untungnya, python menyediakan fungsi `nama_array.pop(index)` untuk menghapus dan menyimpan suatu elemen dalam array.

Maka tepat sebelum block `for line in lines`, ketikan:

```
...
return array_of_data

raw_header = lines.pop(0)
header = convert_line_to_data(raw_header)
print(header)

for line in lines:
...
...
```

- lines.pop(0) akan menghapus elemen indeks ke 0 dari lines ( berarti string baris pertama) **sekaligus** mereturn elemen yang baru saja dihapus, maka kita simpan elemen yang baru saja dihapus menjadi raw\_header
- kemudian, lakukan konversi string menjadi format array. Gunakan fungsi yang telah didefinisikan, yaitu convert\_line\_to\_data terhadap raw\_header.
- coba print(header) untuk melihat

Jangan ubah kode bagian lain apapun, dan coba jalankan program. Hasil yang keluar adalah sebagai berikut:



Berikut adalah kode lengkap sejauh ini:

```
f = open("korean_drama.csv", "r")
raw_lines = f.readlines()
f.close()
lines = [raw_line.replace("\n", "") for raw_line in raw_lines]

def convert_array_data_to_real_values(array_data):
    arr_cpy = array_data[:]
    for i in range(3):
        if(i == 0):
            arr_cpy[i] = int(arr_cpy[i])
```

```

        elif(i == 2):
            arr_cpy[i] = float(arr_cpy[i])
        return arr_cpy

def convert_line_to_data(line):
    raw_array_of_data = line.split(",")
    array_of_data = [data.strip() for data in raw_array_of_data]
    return array_of_data

raw_header = lines.pop(0)
header = convert_line_to_data(raw_header)
print(header)

for line in lines:
    array_of_data = convert_line_to_data(line)
    real_values = convert_array_data_to_real_values(array_of_data)
    print(real_values)

```

## 4. Modifikasi dan Menyimpan Data

Sebelum melakukan modifikasi data itu sendiri, kita harus menyimpan array `real_values` kita ke dalam sebuah array sehingga formatnya:

```
[[1, 'goblin', 8.5]
[2,'itaewon class', 9], ...]
```

Modifikasi block `for line in lines` menjadi:

```

datas = []
for line in lines:
    array_of_data = convert_line_to_data(line)
    real_values = convert_array_data_to_real_values(array_of_data)
    datas.append(real_values)
print(datas)

```

Jadi kita buat array bernama `datas` yang nantinya akan menyimpan semua data. Kemudian, tambahkan `real_values` ke array `datas` tersebut. Kemudian print. Hasilnya adalah:



Perfekt! Selanjutnya, buat fungsi yang bisa memodifikasi variabel **datas** tersebut. Dalam kasus ini , fungsi akan **langsung memodifikasi datas**. Kita tidak akan melakukan copy dari seluru matriks datas. Alasannya adalah:

- Operasi copy dari array yang besar seperti datas, apalagi saat operasi pengubahan data dilakukan akan sangat costly
- Kita tidak perlu membuat lebih dari satu variabel yang menyimpan datas, karena kita yakin kita tidak akan menggunakan datas yang outdated.

Oleh karena itu, di ujung file, buat fungsi baru **modify\_datas( idx, col, value )**

...

```
print(datas)

def modify_datas(idx, col, value):
    datas[idx][col] = value

# ubah rating goblin jadi 5.5, jangan triggered ya :D
modify_datas(0, 2, 5.5)
print(datas)
```

Karena datas adalah sebuah matriks, maka pertama tama kita harus tahu index elemen ke berapa yang ingin diakses, kemudian kita harus tahu kolom mana yang ingin diubah, kemudian baru value baru yang ingin dimasukan ke kolom tersebut.

Hasil kode diatas adalah:



Bisa dilihat ada perbedaan di 2 print (datas) yang terakhir, dimana yang awalnya rating goblin 8.5, menjadi 5.5.

Dalam fungsi `modify_index` ini, ada improvement yang bisa anda lakukan (jadi peer aja ya):

- bila idx == 0 (id) , maka kode tidak dilanjutkan karena id tidak boleh diubah. Cara supaya kode dalam fungsi tidak dilanjutkan adalah dengan keyword return (selain return value, bisa return empty untuk force stop fungsi), misalnya fungsi dibawah ini tidak akan print num x is blablabla apabila x == 5:

```
def trythis(x):
    if(x == 5):
        return
    print("num x is " + str(x))
```

- bila idx == 2 , pastikan tipe value == float atau int , bila tidak keduanya, print error ke console dan jangan lanjutkan kode. Cara mengetahui tipe value adalah dengan type(value)

## Menyimpan File

Part wajib terakhir yaitu menyimpan file, hore!!!!!!

Masih inget kan sintaks open(**filename, mode**) tadi? kita cukup mengubah mode menjadi write("w") dan menambahkan lines ke file tersebut ... tapi sayangnya tidak semudah itu, karena kita harus mengonversi lagi dari datas menjadi bentuk string yang dipisahkan koma di csv.

Maka bikin method baru di akhir file, namakan method tersebut convert\_datas\_to\_string

```
...
modify_datas(0, 2, 5.5)
print(datas)

# converting datas
def convert_datas_to_string():
    string_data = ",".join(header) + "\n"
    for arr_data in datas:
        arr_data_all_string = [str(var) for var in arr_data]
        string_data += ",".join(arr_data_all_string)
        string_data += "\n"
    return string_data

print(convert_datas_to_string())
```

Mari bahas lagi satu per satu line fungsi di atas:

- `string_data = ",".join(header) + "\n"`

fungsi **simbol.join(array)** digunakan untuk menggabungkan suatu array yang semua elemennya berbentuk string (header semua elemennya sudah string) menjadi sebuah string yang dipisahkan oleh suatu simbol, misal :

```
test = ".join(["wahai", "manusia"])
# hasilnya "wahai manusia"

tesuto = " warudo ".join(["za", "is invincible"])
# hasilnya za warudo is invincible
```

ditambahkan \n diakhir agar dia pindah ke line baru

- kemudian untuk setiap arr\_data dalam datas :

**arr\_data\_all\_string = [str(var) for var in arr\_data]**

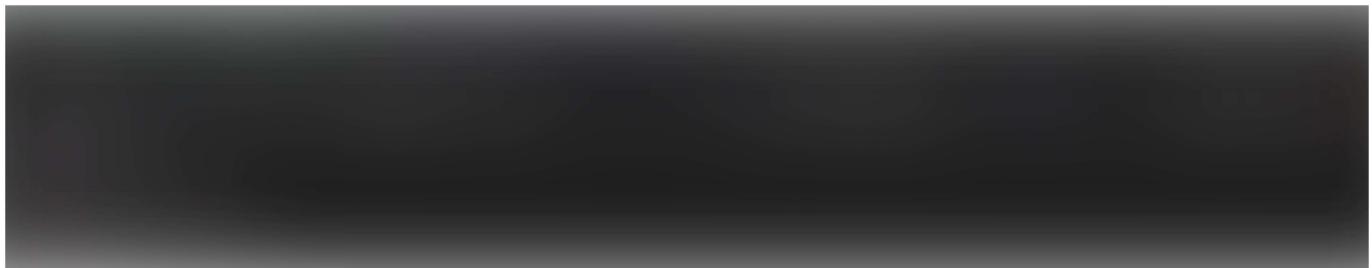
Ini bertujuan untuk mengonversikan kembali variabel yang tadinya tipenya integer atau float menjadi string kembali

- **string\_data += ",".join(arr\_data\_all\_string) dan string\_data += "\n"**

Mirip seperti header diatas, kita join arr\_data\_all\_string yang isinya adalah array of string dengan koma dan menambahkannya ke string\_data, jangan lupa menambahkan \n untuk new line

- **return string\_data**

Hasil dari run kode diatas adalah:



sekarang sudah dalam format string dan siap disimpan di file csv lagi

Untuk menyimpan file, gunakan fungsi open dengan mode “w”

```
...  
  
print(convert_datas_to_string())  
datas_as_string = convert_datas_to_string()  
  
f = open("korean_drama.csv", "w")  
f.write(datas_as_string)  
f.close()
```

f.write digunakan untuk menuliskan ke file yang dideskripsikan, lalu beres deh! Coba jalankan kode, dan lihat file CSV anda, seharusnya file CSV anda isinya begini:



Ada enter di akhir file, tapi gapapa biarin aja. Hapus semua print() mu (supaya lebih bersih saat di run nantinya), sehingga kode akan menjadi seperti ini:

```
f = open("korean_drama.csv","r")  
raw_lines = f.readlines()  
f.close()  
lines = [raw_line.replace("\n", "") for raw_line in raw_lines]  
  
def convert_array_data_to_real_values(array_data):  
    arr_cpy = array_data[:]  
    for i in range(3):  
        if(i == 0):  
            arr_cpy[i] = int(arr_cpy[i])  
        elif(i == 2):  
            arr_cpy[i] = float(arr_cpy[i])  
    return arr_cpy
```

```
def convert_line_to_data(line):
    raw_array_of_data = line.split(",")
    array_of_data = [data.strip() for data in raw_array_of_data]
    return array_of_data

raw_header = lines.pop(0)
header = convert_line_to_data(raw_header)
datas = []

for line in lines:
    array_of_data = convert_line_to_data(line)
    real_values = convert_array_data_to_real_values(array_of_data)
    datas.append(real_values)

def modify_datas(idx, col, value):
    datas[idx][col] = value

# ubah rating goblin jadi 5.5
modify_datas(0, 2, 5.5)

def convert_datas_to_string():
    string_data = ",".join(header) + "\n"
    for arr_data in datas:
        arr_data_all_string = [str(var) for var in arr_data]
        string_data += ",".join(arr_data_all_string)
        string_data += "\n"
    return string_data

datas_as_string = convert_datas_to_string()

f = open("korean_drama.csv", "w")
f.write(datas_as_string)
f.close()
```

Silahkan modifikasi pemanggilan `modify_datas`, misalnya ubah rating jadi 8 atau 7 lalu lihat kembali csv setelah run code

## **SELAMAT, KAMU SUDAH MENYELESAIKAN BASIC READ/WRITE FILE CSV DI PYTHON TANPA LIBRARY**

Untuk merayakannya, ini gambar doraemon lucu:



Berikut ini adalah materi optional:

## (Optional) Menerima user input dan mengubah data

Dalam materi ini, kita akan membuat drama baru dan menambahkan drama ke datas sesuai dengan input dari user

Untuk merefresh memori, cara menerima user input adalah dengan fungsi `input()`.

Untuk materi ini, pertama tama hapus dulu 2 line ini dibawah fungsi `modify_datas`, karena kita akan modifikasi data berdasarkan user input:

```
# ubah rating goblin jadi 5.5
modify_datas(0, 2, 5.7)
```

Lalu hapus juga kode yang menyimpan file, karena kita akan menyimpan file setelah user input:

```
datas_as_string = convert_datas_to_string()
f = open("korean_drama.csv", "w")
f.write(datas_as_string)
f.close()
```

kemudian di penghujung file, tambahkan kode penerimaan user input:

```
...
f.close()

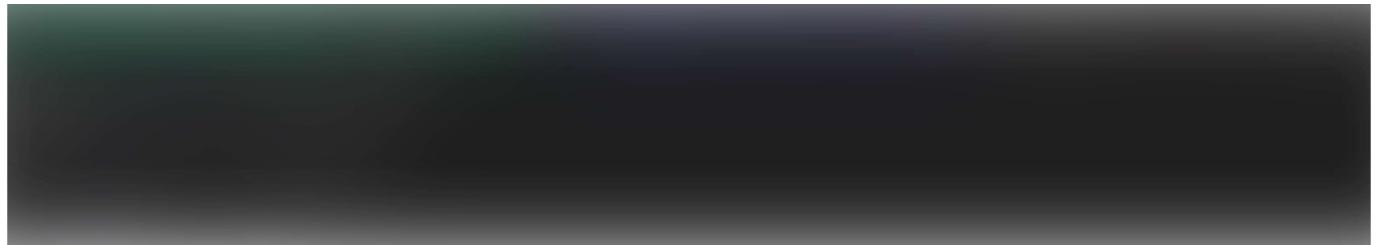
new_drama_idx = datas[-1][0] + 1
new_drama_name = input("What is the drama name? \n>>")
new_drama_rating = float(input("What is your rating? \n>>"))
new_drama = [new_drama_idx, new_drama_name, new_drama_rating]

print(new_drama)
```

Mari kita breakdown codenya:

- `new_drama_idx = datas[-1][0] + 1`  
datas[-1] akan mendapatkan elemen terakhir dari datas, yang berarti elemen dengan id tertinggi (kita asumsikan elemen makin lama makin tinggi idnya), lalu kita tambahkan dengan 1
- `new_drama_name = input("What is the drama name? \n>>")`  
Mendapatkan input user, biar fancy tambahan new line sama >>
- `new_drama_rating ... sama kyk diatas`

ketika dijalankan, hasil yang keluar:



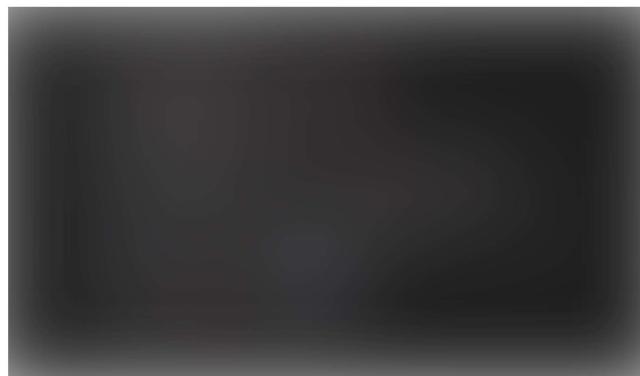
hello dan 5 adalah input anda sendiri. Jangan lupa rating harus berupa integer/ float

Lalu tambahkan new\_drama ke datas menggunakan append lalu simpan data ke file

...

```
new_drama = [new_drama_idx, new_drama_name, new_drama_rating]  
  
datas.append(new_drama)  
  
datas_as_string = convert_datas_to_string()  
f = open("korean_drama.csv", "w")  
f.write(datas_as_string)  
f.close()
```

Coba jalankan kode, dan lihat file csv anda lagi, kalo mengisi hello dan 5, hasilnya:



## (Tambahan) Input Loop

Kamu mungkin berpikir: “wah kalo gitu tiap mau nambah item harus ngejalanin ulang ya file pythonnya?”, tidak juga. Kita bisa menggunakan loop untuk mendapatkan input.

Dalam materi ini, kita akan meminta user untuk mengisikan film sampai user memasukan “stop” sebagai nama film

Biasanya ada 2 loop yang digunakan

- For loop : biasanya untuk iterasi dalam list dan melakukan task yang sama sebanyak X kali (kita tahu kapan selesai)
- While loop : biasanya untuk task yang kita tidak tahu kapan selesai

Oleh karena kita ingin user memasukan detail film sampai “stop” diketikkan sebagai nama film, maka kita menggunakan while loop. modifikasi kode dibawah ini:

```
new_drama_idx = datas[-1][0] + 1
new_drama_name = input("What is the drama name? \n>>")
new_drama_rating = float(input("What is your rating? \n>>"))
new_drama = [new_drama_idx, new_drama_name, new_drama_rating]
```

Menjadi:

```
...
new_drama_name = ""
new_dramas = []
while(new_drama_name != "stop"):
    new_drama_idx = datas[-1][0] + 1
    new_drama_name = input("What is the drama name? \n>>")
    if(new_drama_name == "stop"):
        continue
    new_drama_rating = float(input("What is your rating? \n>>"))
    new_drama = [new_drama_idx, new_drama_name, new_drama_rating]
    new_dramas.append(new_drama)

# hapus line datas.append(new_drama)

datas += new_dramas
...
```

Penjelasan:

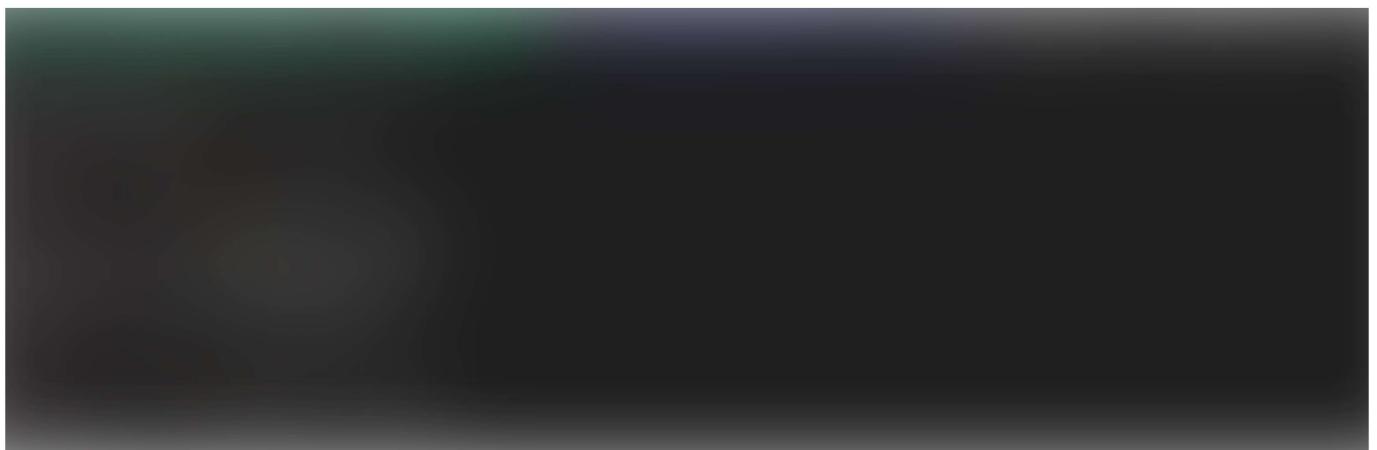
- new\_drama\_name diinisiasi dulu menjadi empty string ( “ ” ) di awal supaya tidak terjadi error saat pengecekan kondisi while loop pertama

- simpan semua new\_drama yang dibuat user ke array new\_dramas
- **while(new\_drama\_name != "stop"):**  
sampai drama name diisikan “stop” , lakukan:
- bila new\_drama\_name adalah “stop”, maka kita tidak melanjutkan kode dibawahnya. Keyword **continue** digunakan untuk tidak melanjutkan kode yang ada dibawah loop dan langsung melanjutkan ke iterasi selanjutnya, sehingga akan dilakukan pengecekan kondisi new\_drama\_name yang sekarang “stop” lalu keluar dari loop.
- **datas.append(new\_drama)** diganti menjadi **datas += new\_dramas**  
digunakan operator += karena new\_dramas berbentuk array of new\_drama. Tidak bisa digunakan .append, karena hasilnya akan menjadi:

```
#kode dibawah tidak akurat
datas.append(new_dramas)
# hasil : [[...], [...], [...], [ [...], [...] ]], ada matrix dalam
matrix

#kode dibawah akurat
datas += new_dramas
# hasil : [...], [...], [...], [...], [...] , tidak ada double
matrix
```

maka ketika dijalankan dan diketikan nama dan skor sebagai berikut:



File CSV nya menjadi:



## (Optional) Merapihkan Kode

Yep , biar kode bisa dibaca teman dan asisten kami sangat apresiasi bila kalian sesekali merapihkan kode. Jadi untuk bagian ini, kalian tidak perlu mengikuti dan code along. cukup baca dan pahami , supaya dalam tubes kalian bisa menerapkan kerapihan ini.

Pertama tama, buat file constants.py yang isinya adalah sebagai berikut:

```
ID = 0  
NAME = 1  
RATING = 2
```

Lalu import ketiga variabel yang dibuat di constants.py ke file read\_csv.py di bagian paling atas file read\_csv.py

```
from constants import ID, NAME, RATING
```

```
...
```

Kemudian ubah SEMUA yang berkaitan dengan indeks data (0 sebagai id, 1 sebagai name, 2 sebagai rating) menjadi ID, NAME, dan RATING, misalnya dalam fungsi **convert\_array\_data\_to\_real\_values**, menjadi:

```
def convert_array_data_to_real_values(array_data):  
    arr_cpy = array_data[:]
```

```
for i in range(3):
    if(i == ID):
        arr_cpy[i] = int(arr_cpy[i])
    elif(i == RATING):
        arr_cpy[i] = float(arr_cpy[i])
return arr_cpy
```

Terlihat bedanya? Awalnya misalnya orang awam melihat dia harus menelusuri dulu dan mungkin menanyakan programmer kenapa bagian ini ada 0 dan 2 , tapi sekarang menjadi lebih clear hanya dengan mengubah 0 menjadi ID dan 2 menjadi RATING

Ada banyak ide untuk merapihkan kode ini (sekali lagi, kalau mager implemen, cukup baca aja biar di tubes nanti konsep code yang rapih bisa diinget” ) , diantaranya:

- Masukin semuanya jadi fungsi, tidak ada lagi block **for line in lines**, buat satu fungsi yang menerima parameter **line** dan panggil fungsi tersebut
- nama file “korean\_drama.csv” masukan sebagai satu constant misalnya `DATABASE_DRAMA_NAME = “korean_drama.csv”` lalu import lagi ke file utama dan gunakan instead of hard code korean\_drama.csv. Tujuannya, misalnya kalian pengen ke file yang berbeda cukup ganti di constants
- dst... (untuk sekarang yg diatas dulu aja)

Sekali lagi, Thankyou udah ngebaca tutorial ini dari awal sampe BENER BENER akhir, semoga tutorial ini bermanfaat dalam pengajaran tubes :).

Get the Medium app

