Mid-Trimester Test – Trimester 1, 2022

Thi Van Anh DUONG Student ID: 90023112

Diploma of Information Technology, Curtin College

FOP1005: Fundamentals of Programming

Coordinator: Jason Brown

1 May 2022

Student Declaration of Originality

☒	This assignment is my own original work, and no part has been copied from another student's work or source, except where it is clearly attributed.
☒	All facts/ideas/claims are from academic sources are paraphrased and cited/referenced correctly.
☒	I have not previously submitted this work in any form for THIS or for any other unit; or published it elsewhere before.
☒	No part of this work has been written for me by another person.
☒	I recognise that should this declaration be found to be false, disciplinary action could be taken and penalties imposed in accordance with Curtin College policy.

Electronic Submission:

☒	I accept that it is my responsibility to check that the submitted file is the correct file, is readable and has not been corrupted.
☒	I acknowledge that a submitted file that is unable to be read cannot be marked and will be treated as a non-submission.
☒	I hold a copy of this work if the original is damaged, and will retain a copy of the Turnitin receipt as evidence of submission.

QUESTION ONE (Total: 20 marks): Data Types, Python and Linux.

a) **(5 marks)** Write code to modify **Fibonacci.py** (below):

i) **Correct** the errors and run the code. (2)

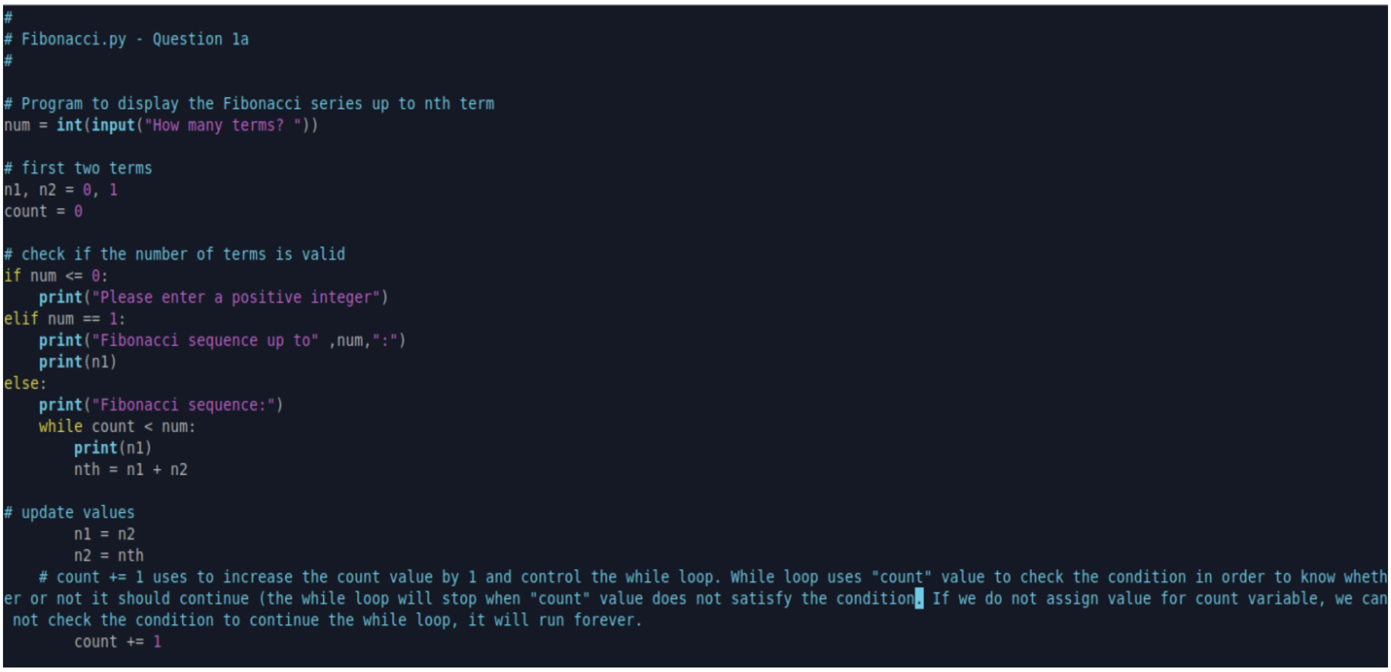
ii) Write the correct **operator** in place of the '?' symbol. (1)

iii) Write the correct **data type** in place of <data_type> in the first line of the code. (1)

iv) **Explain** the purpose of count+=1 in a comment. Put the comment before the line. (1)

(Submit separately as a python file)

Python file: Fibonacci.py



```
#
# Fibonacci.py - Question 1a
#

# Program to display the Fibonacci series up to nth term
num = int(input("How many terms? "))

# first two terms
n1, n2 = 0, 1
count = 0

# check if the number of terms is valid
if num <= 0:
    print("Please enter a positive integer")
elif num == 1:
    print("Fibonacci sequence up to" ,num,":")
    print(n1)
else:
    print("Fibonacci sequence:")
    while count < num:
        print(n1)
        nth = n1 + n2

# update values
        n1 = n2
        n2 = nth
    # count += 1 uses to increase the count value by 1 and control the while loop. While loop uses "count" value to check the condition in order to know whether or not it should continue (the while loop will stop when "count" value does not satisfy the condition. If we do not assign value for count variable, we can not check the condition to continue the while loop, it will run forever.
        count += 1
```

b) **(5 marks)**. Explain the difference between **Syntax** and **Semantic** errors with examples.

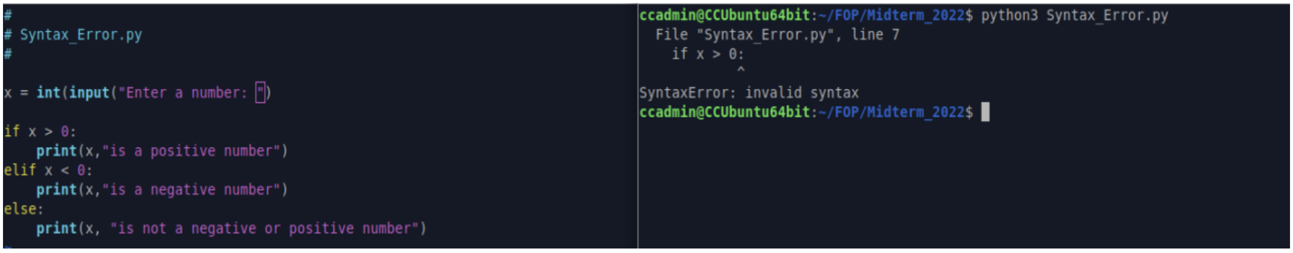
The difference between **Syntax** and **Semantic** errors:

	Syntax	Semantic
Meaning	Is the set of rules of any statement in programing language – caused by grammatically incorrect statements.	Is the meaning associated with any statement in the programming language
Occur	during the parsing of input code	during the execution of the code and after it has been checked as grammatically correct
Errors in Python	• Not ending *if, for, while* statement or a *function* with a colon • Misspelling Python keywords (using matplotlib. pylot instead of matplotlib.pyplot) • Using " and ' at the same time in the print statement • Lack of a parenthesis	• Incorrect variable types • Use nonexistent variables to continue the program • Use the wrong operation to calculate the desired output
Fixing	Easy to recognize and fix (because when we have syntax error the program will not run and display an error message for us to spot the error).	Hard to recognize and fix (program with semantic error can still be run, but it does not give us the operate as intended).

Syntax Error Example:

Python file: (Syntax_Error.py)

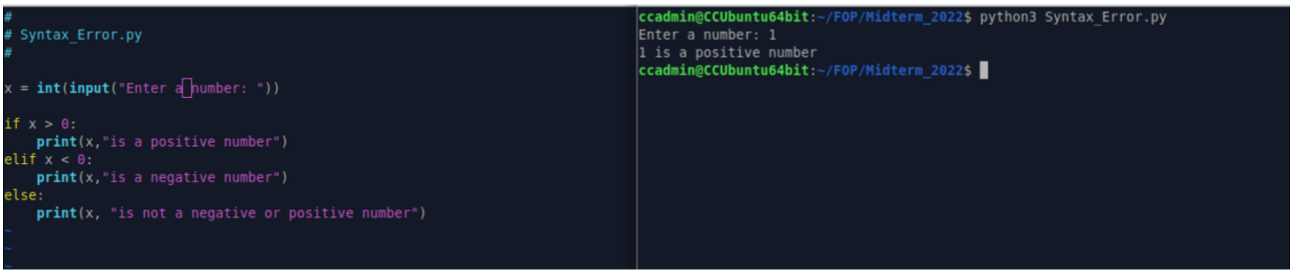
Lack of a parenthesis in line 5



```
#
# Syntax_Error.py
#

x = int(input("Enter a number: ")

if x > 0:
    print(x,"is a positive number")
elif x < 0:
    print(x,"is a negative number")
else:
    print(x, "is not a negative or positive number")
```

```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 Syntax_Error.py
  File "Syntax_Error.py", line 7
    if x > 0:
           ^
SyntaxError: invalid syntax
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$
```

Fixed:



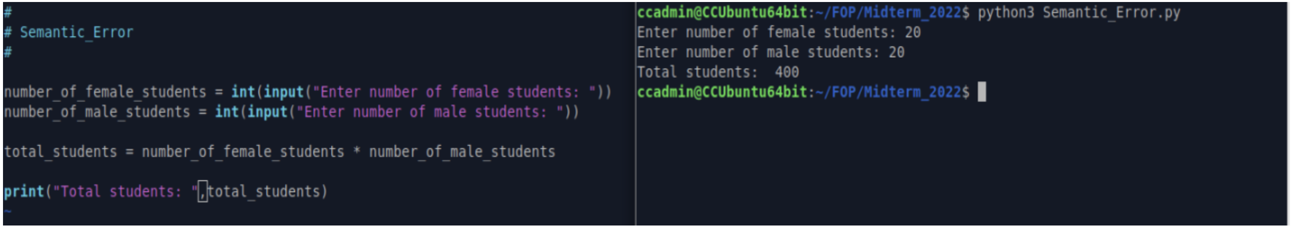
```
#
# Syntax_Error.py
#

x = int(input("Enter a number: "))

if x > 0:
    print(x,"is a positive number")
elif x < 0:
    print(x,"is a negative number")
else:
    print(x, "is not a negative or positive number")
~
~
```

```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 Syntax_Error.py
Enter a number: 1
1 is a positive number
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$
```

Semantic Error Example:

Python file: (Semantic_Error.py)

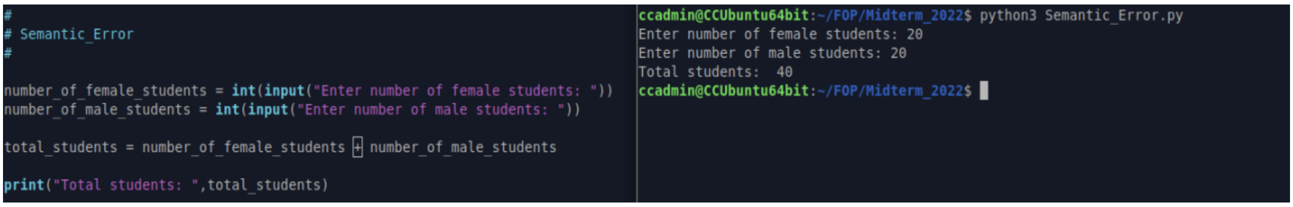
Use the wrong operation to calculate, the program still run but give me the wrong answer:



```
#
# Semantic_Error
#

number_of_female_students = int(input("Enter number of female students: "))
number_of_male_students = int(input("Enter number of male students: "))

total_students = number_of_female_students * number_of_male_students

print("Total students: ",total_students)
```

```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 Semantic_Error.py
Enter number of female students: 20
Enter number of male students: 20
Total students:  400
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$
```

Fixed:



```
#
# Semantic_Error
#

number_of_female_students = int(input("Enter number of female students: "))
number_of_male_students = int(input("Enter number of male students: "))

total_students = number_of_female_students + number_of_male_students

print("Total students: ",total_students)
```

```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 Semantic_Error.py
Enter number of female students: 20
Enter number of male students: 20
Total students:  40
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$
```

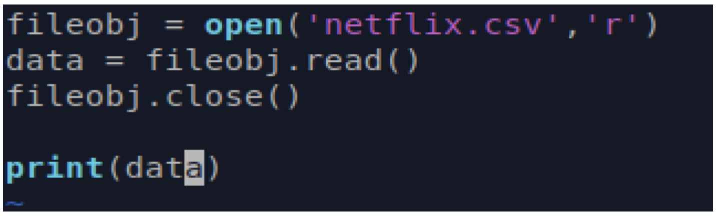
c) **(5 marks)**. What **type** of variable will result from each of the following:

i) variable1 = 5 <= 10 Boolean

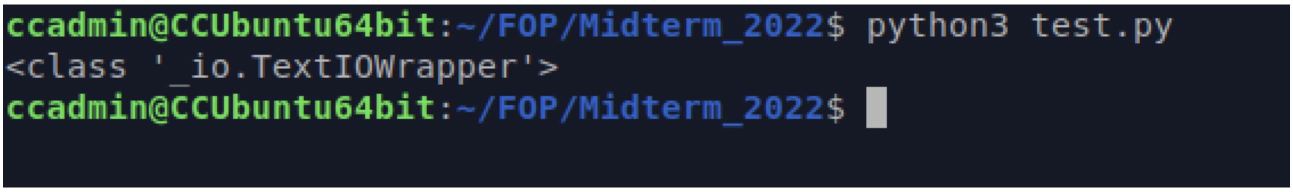
ii) variable2 = len(instring)-1 Integer

iii) variable3 = open(filename) File Object_____________

➔ it depends on what type of file you want to open.



```
fileobj = open('netflix.csv','r')
data = fileobj.read()
fileobj.close()

print(data)
~
```



```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 test.py
<class '_io.TextIOWrapper'>
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$
```

iv) variable4 = 12 % 5 Integer

v) variable5= 10/3 Float

d) **(5 marks)**. Write **Linux commands** for the following:

i) **Copy** a file test1.py from Prac01 directory to Subdir1 directory in the following directory structure. (2)

FOP-
Prac01
Prac02
Prac03-
Subdir1
Subdir2

(Where FOP is at the top level then Prac01, 02 and 03 are at next level and Subdir1 and 2 are next to next level.)

➔ Assuming, I am currently in Subdir1:

cp ../../Prac01/test1.py .

ii) To **find** a hidden file in a directory.(1)

➔ ls -a

iii) To **delete** a directory DIR which has a file test2.py in it. (2)

➔ rm -r DIR

QUESTION TWO (Total: 20 marks): Arrays and Plotting

a) **(5 marks).** What is the **output** of the following code?

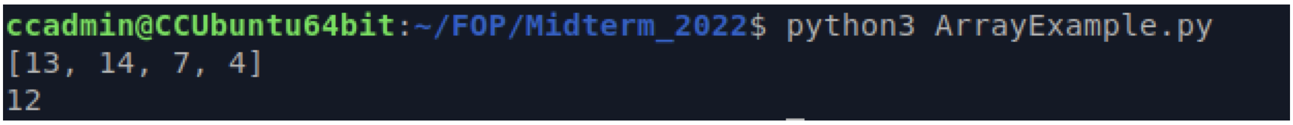
ArrayExample.py

```
A = [[13, 14, 7, 4], [17, 8, 12], [12, 10, 14, 7], [14, 17, 10, 8]]

print (A[0])

print (A[1][2])
```

Output: [13, 14, 7, 4]

12

Python file: ArrayExample.py



```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 ArrayExample.py
[13, 14, 7, 4]
12
```

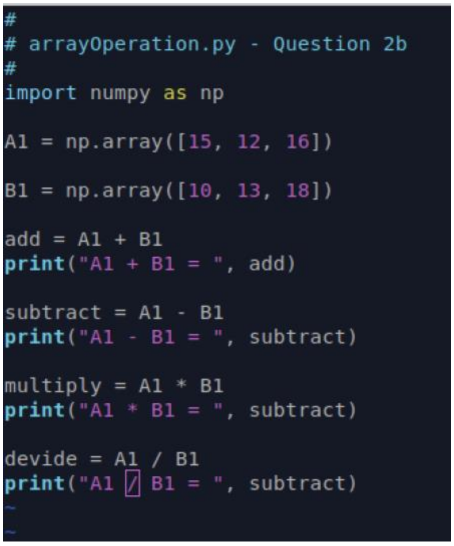
b) (5 marks). Write a program in python to**:**

i. Create 2 arrays A1 and B1 with 3 elements each (A1= [15, 12, 16] and B1= [10, 13, 18]).

ii. Add and subtract elements of A1 and B1.

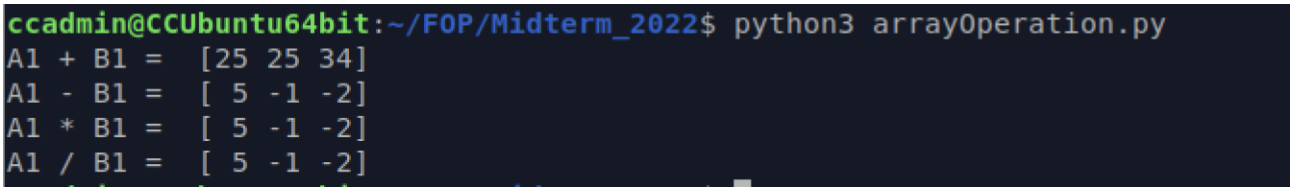
iii. Multiply and divide the elements of A1 and B1. (Submit separately as a python file)

Python file: arrayOperation.py



```
#
# arrayOperation.py - Question 2b
#
import numpy as np

A1 = np.array([15, 12, 16])

B1 = np.array([10, 13, 18])

add = A1 + B1
print("A1 + B1 = ", add)

subtract = A1 - B1
print("A1 - B1 = ", subtract)

multiply = A1 * B1
print("A1 * B1 = ", subtract)

devide = A1 / B1
print("A1 / B1 = ", subtract)
~
~
```

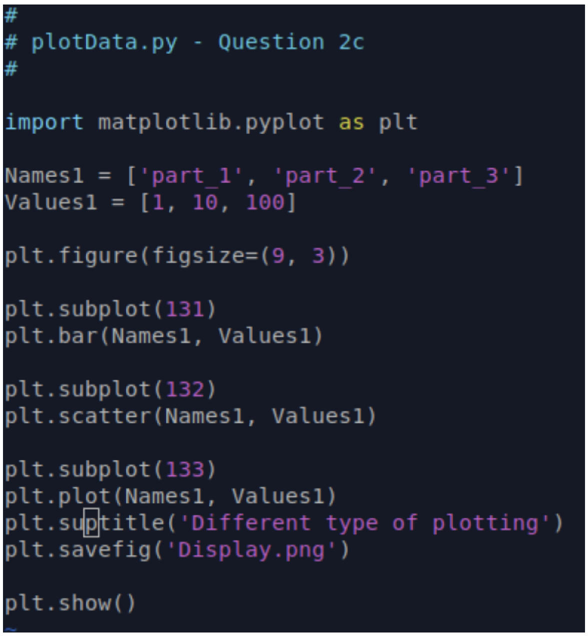
Result:



```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 arrayOperation.py
A1 + B1 =  [25 25 34]
A1 - B1 =  [ 5 -1 -2]
A1 * B1 =  [ 5 -1 -2]
A1 / B1 =  [ 5 -1 -2]
```

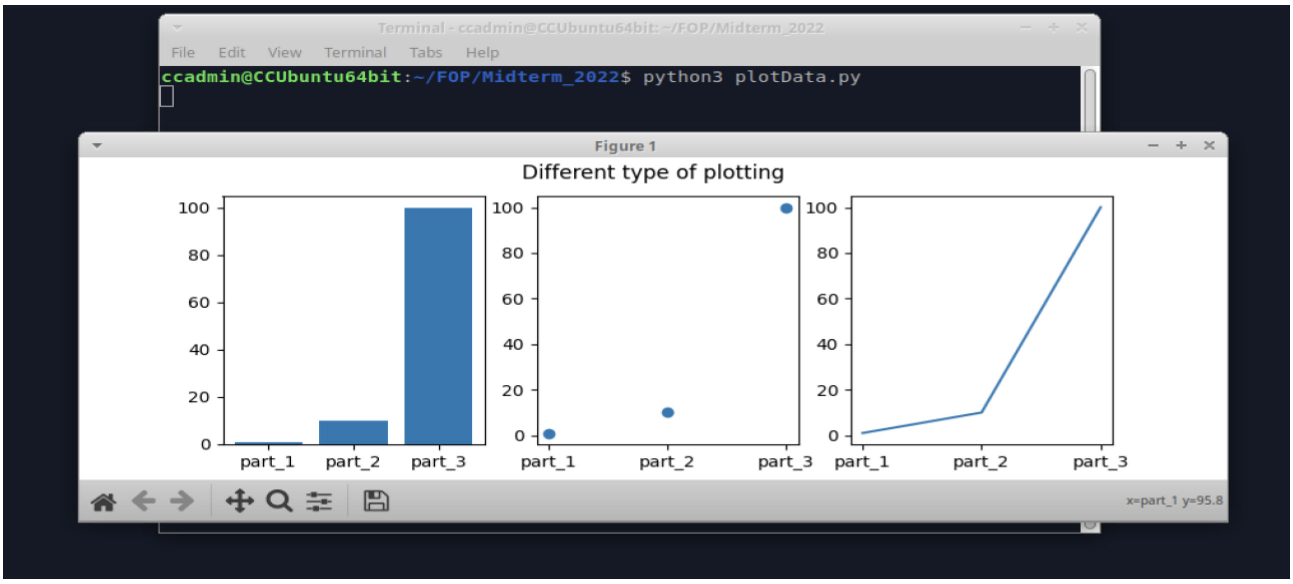
c) **(5 marks)**. Modify the code and plot the output according to the given instruction

Python file: plotData.py



```
#
# plotData.py - Question 2c
#

import matplotlib.pyplot as plt

Names1 = ['part_1', 'part_2', 'part_3']
Values1 = [1, 10, 100]

plt.figure(figsize=(9, 3))

plt.subplot(131)
plt.bar(Names1, Values1)

plt.subplot(132)
plt.scatter(Names1, Values1)

plt.subplot(133)
plt.plot(Names1, Values1)
plt.suptitle('Different type of plotting')
plt.savefig('Display.png')

plt.show()
```

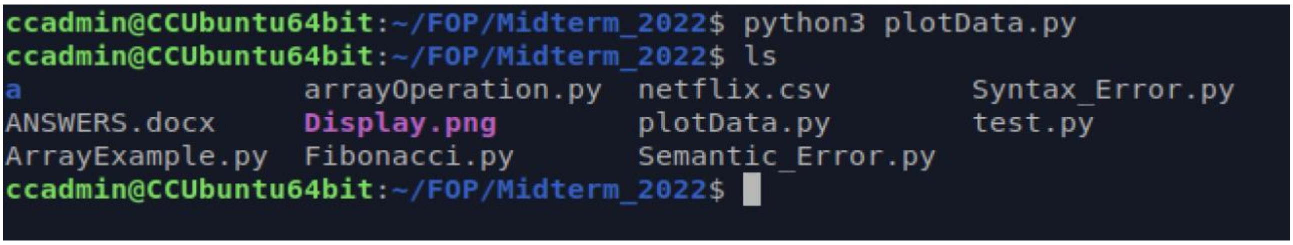
Result:



```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 plotData.py
```

Terminal - ccadmin@CCUbuntu64bit: ~/FOP/Midterm_2022
File Edit View Terminal Tabs Help
Figure 1
Different type of plotting
100
80
60
40
20
0
part_1
part_2
part_3
x=part_1 y=95.8

And the figure is saved as "Display.png":



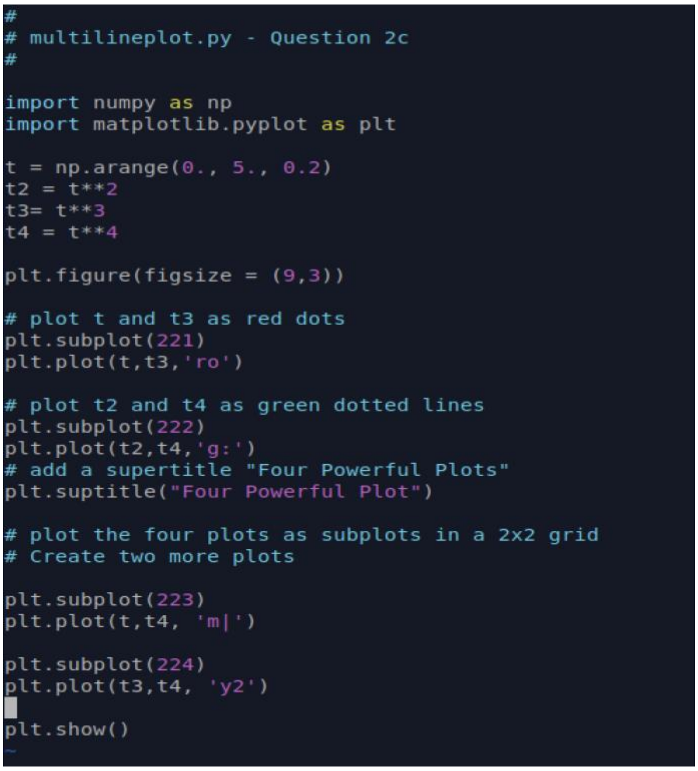
```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 plotData.py
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ ls
a                 arrayOperation.py  netflix.csv        Syntax_Error.py
ANSWERS.docx      Display.png        plotData.py        test.py
ArrayExample.py   Fibonacci.py       Semantic_Error.py
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ 
```

d) **(5 marks)**. The following code creates four arrays: t, t2, t3 and t4. Modify the code to:

- plot t and t3 as red dots (1)
- plot t2 and t4 as green dotted lines (1)
- add a supertitle "Four Powerful Plots" (1)
- plot the four plots as subplots in a 2x2 grid (2)

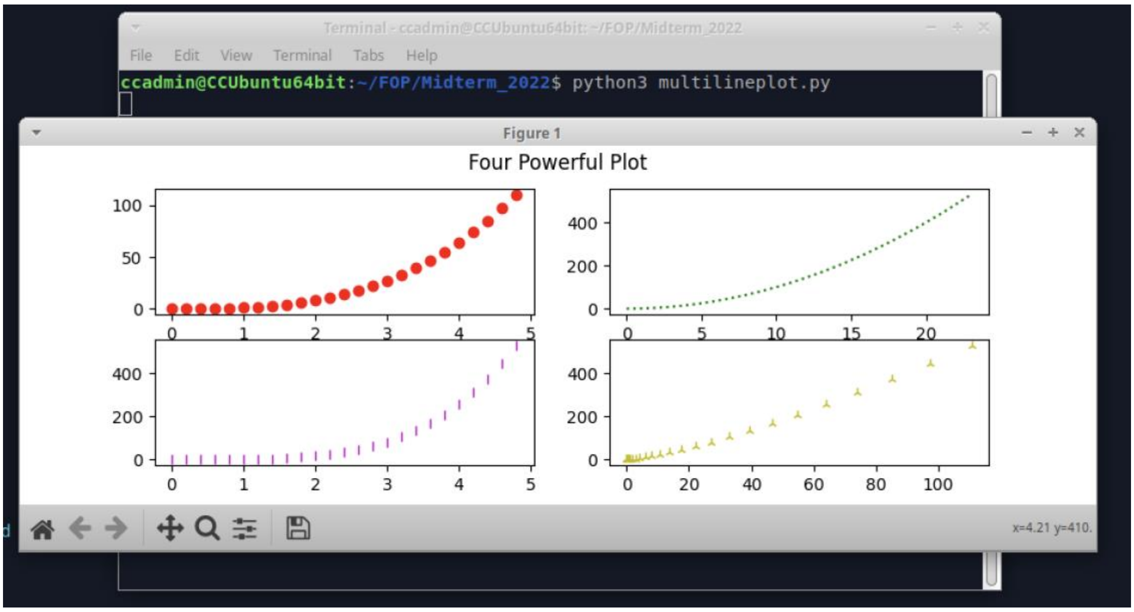
(Submit separately as a python file)

Python file: multilineplot.py



```
#
# multilineplot.py - Question 2c
#

import numpy as np
import matplotlib.pyplot as plt

t = np.arange(0., 5., 0.2)
t2 = t**2
t3= t**3
t4 = t**4

plt.figure(figsize = (9,3))

# plot t and t3 as red dots
plt.subplot(221)
plt.plot(t,t3,'ro')

# plot t2 and t4 as green dotted lines
plt.subplot(222)
plt.plot(t2,t4,'g:')
# add a supertitle "Four Powerful Plots"
plt.suptitle("Four Powerful Plot")

# plot the four plots as subplots in a 2x2 grid
# Create two more plots

plt.subplot(223)
plt.plot(t,t4, 'm|')

plt.subplot(224)
plt.plot(t3,t4, 'y2')

plt.show()
~
```

Result:

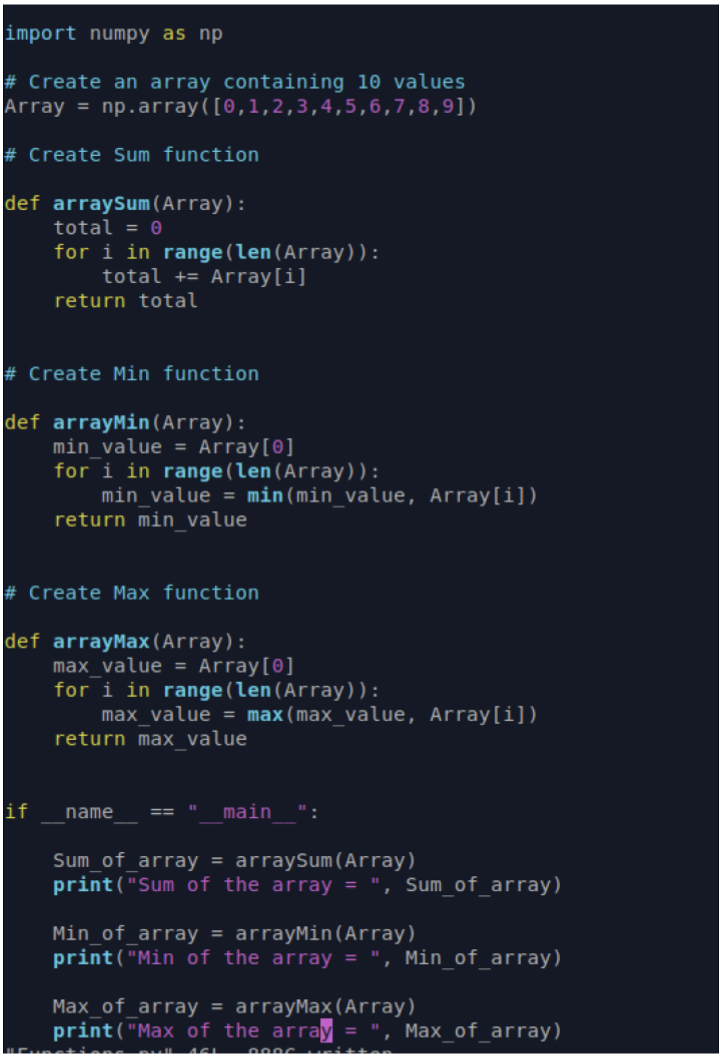


```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 multilineplot.py
```

QUESTION THREE (Total: 20 marks) Functions/List Comprehensions

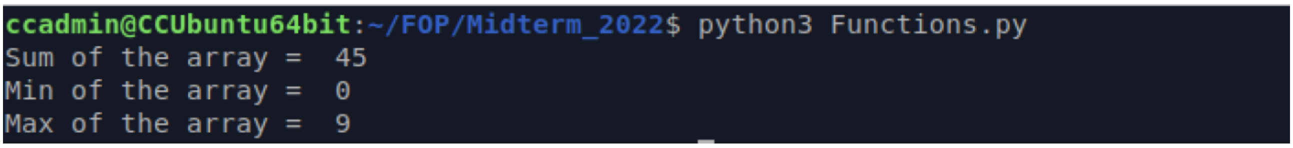
a) **(8 marks)**. Write three functions for calculating the Sum, Min and Max of an Array containing 10 values (using the def keyword). Do not use the inbuilt functions. Write the code for calling these functions in a separate file or the same file for the program to run properly. (Hint: You should use control structures) (Submit separately as a python file)

Python file: Functions.py



```
import numpy as np

# Create an array containing 10 values
Array = np.array([0,1,2,3,4,5,6,7,8,9])

# Create Sum function

def arraySum(Array):
    total = 0
    for i in range(len(Array)):
        total += Array[i]
    return total


# Create Min function

def arrayMin(Array):
    min_value = Array[0]
    for i in range(len(Array)):
        min_value = min(min_value, Array[i])
    return min_value


# Create Max function

def arrayMax(Array):
    max_value = Array[0]
    for i in range(len(Array)):
        max_value = max(max_value, Array[i])
    return max_value


if __name__ == "__main__":

    Sum_of_array = arraySum(Array)
    print("Sum of the array = ", Sum_of_array)

    Min_of_array = arrayMin(Array)
    print("Min of the array = ", Min_of_array)

    Max_of_array = arrayMax(Array)
    print("Max of the array = ", Max_of_array)
"Functions.py" 46L, 888C written
```

Result:

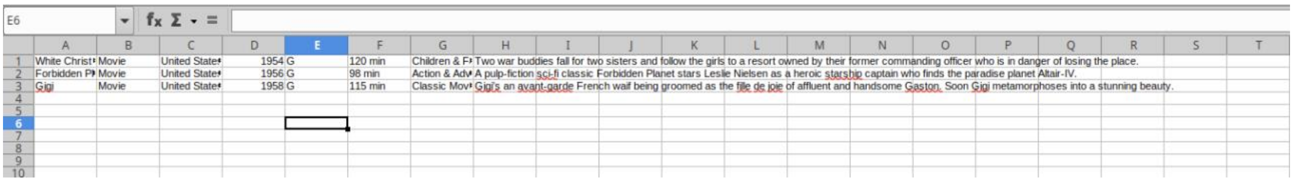


```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 Functions.py
Sum of the array =  45
Min of the array =  0
Max of the array =  9
```

b) **(6 marks)** Explain with an example the situation where using read() is appropriate and also an example where readlines() is appropriate.

Answer:

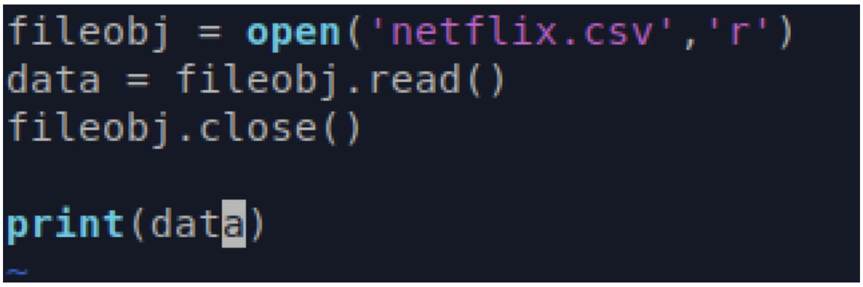
Taking a part of **netflix.csv** in the Assignment1 as an example:



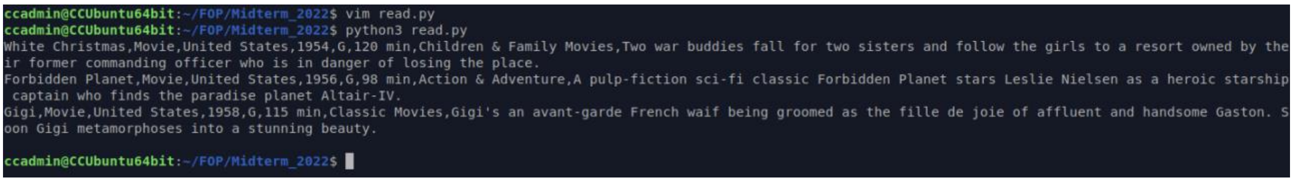
E6

	A	B	C	D	E	F	G	H
1	White Christ	Movie	United State	1954	G	120 min	Children & F	Two war buddies fall for two sisters and follow the girls to a resort owned by their former commanding officer who is in danger of losing the place.
2	Forbidden Pl	Movie	United State	1956	G	98 min	Action & Adv	A pulp-fiction sci-fi classic Forbidden Planet stars Leslie Nielsen as a heroic starship captain who finds the paradise planet Altair-IV.
3	Gigi	Movie	United State	1958	G	115 min	Classic Mov	Gigi's an avant-garde French waif being groomed as the fille de joie of affluent and handsome Gaston. Soon Gigi metamorphoses into a stunning beauty.
4								
5								
6								
7								
8								
9								
10								

read(): return entire content of the file in a really long string when we call it.



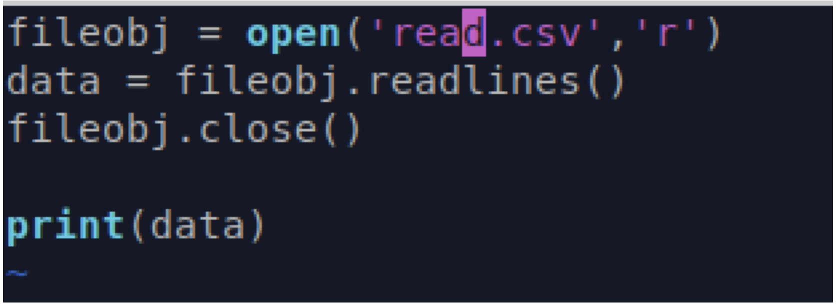
```
fileobj = open('netflix.csv','r')
data = fileobj.read()
fileobj.close()

print(data)
~
```



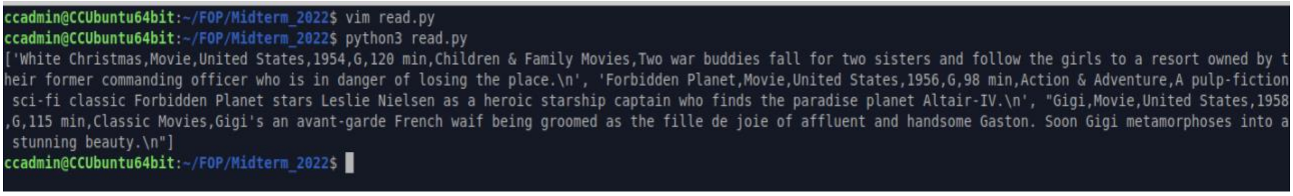
```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ vim read.py
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 read.py
White Christmas,Movie,United States,1954,G,120 min,Children & Family Movies,Two war buddies fall for two sisters and follow the girls to a resort owned by the
ir former commanding officer who is in danger of losing the place.
Forbidden Planet,Movie,United States,1956,G,98 min,Action & Adventure,A pulp-fiction sci-fi classic Forbidden Planet stars Leslie Nielsen as a heroic starship
 captain who finds the paradise planet Altair-IV.
Gigi,Movie,United States,1958,G,115 min,Classic Movies,Gigi's an avant-garde French waif being groomed as the fille de joie of affluent and handsome Gaston. S
oon Gigi metamorphoses into a stunning beauty.

ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ 
```

➔ This method will be useful when you read a small file in order to manipulate the whole content of the file.

readlines(): return all lines in the file in the format of a list (each element is a line in the file) when we call it.



```
fileobj = open('read.csv','r')
data = fileobj.readlines()
fileobj.close()

print(data)
~
```



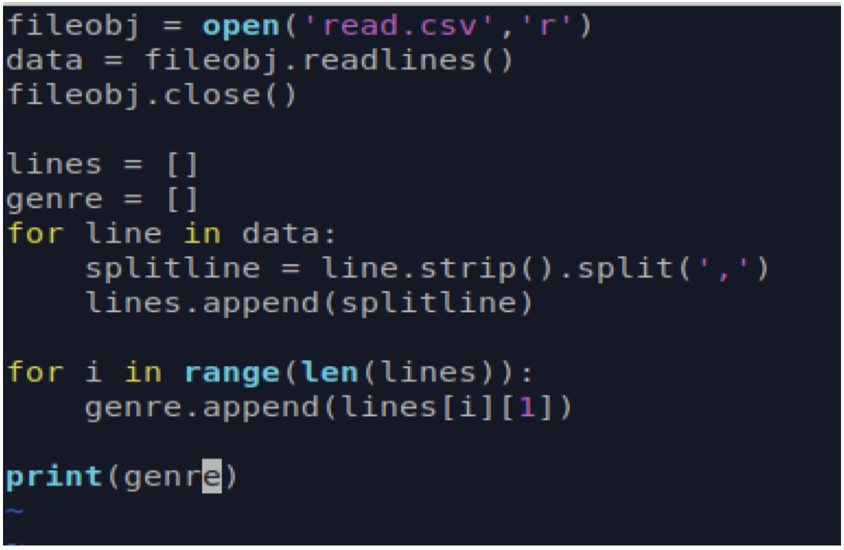
```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ vim read.py
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 read.py
['White Christmas,Movie,United States,1954,G,120 min,Children & Family Movies,Two war buddies fall for two sisters and follow the girls to a resort owned by t
heir former commanding officer who is in danger of losing the place.\n', 'Forbidden Planet,Movie,United States,1956,G,98 min,Action & Adventure,A pulp-fiction
 sci-fi classic Forbidden Planet stars Leslie Nielsen as a heroic starship captain who finds the paradise planet Altair-IV.\n', "Gigi,Movie,United States,1958
,G,115 min,Classic Movies,Gigi's an avant-garde French waif being groomed as the fille de joie of affluent and handsome Gaston. Soon Gigi metamorphoses into a
 stunning beauty.\n"]
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ 
```

➔ Most preferred use, because its return values are in the list format, so it is very useful when we work with a long file which contains a lot of data. List is powerful in this case!

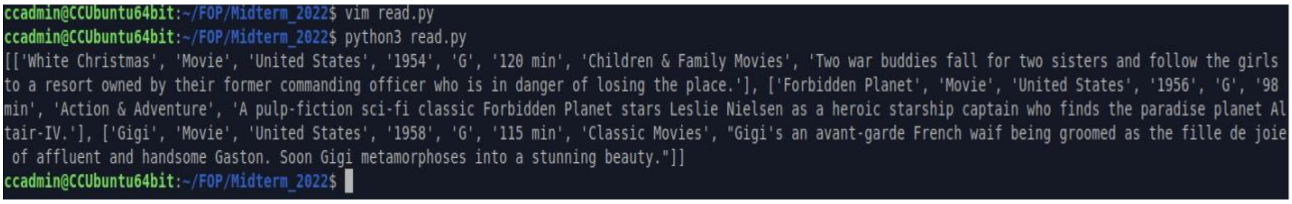
For example: netflix.csv in the Assignment1

It takes ages to manipulate with the data (about 8790 movies) in the file when using **read()** method (because it will print the whole 8790 movies in a very long string and it is such a messy to work with)**,** while we can access the data by using **readlines()** method easily – at this time, we are working with a list and we can access each data (type, genre, description, year,….) for further calculations easily.

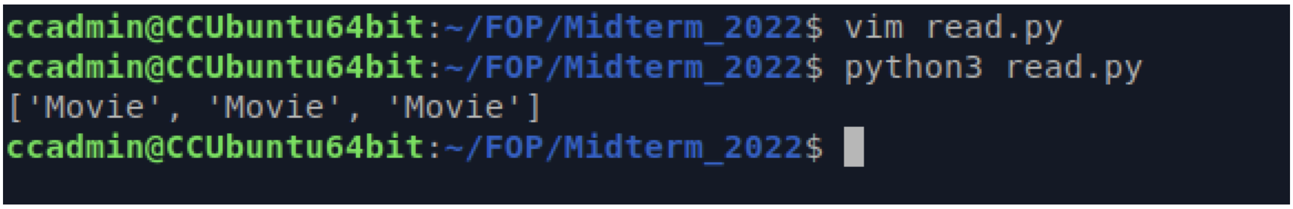
Accessing data (genre):



```
fileobj = open('read.csv','r')
data = fileobj.readlines()
fileobj.close()

lines = []
genre = []
for line in data:
    splitline = line.strip().split(',')
    lines.append(splitline)

for i in range(len(lines)):
    genre.append(lines[i][1])

print(genre)
~
```



```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ vim read.py
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 read.py
[['White Christmas', 'Movie', 'United States', '1954', 'G', '120 min', 'Children & Family Movies', 'Two war buddies fall for two sisters and follow the girls
to a resort owned by their former commanding officer who is in danger of losing the place.'], ['Forbidden Planet', 'Movie', 'United States', '1956', 'G', '98
min', 'Action & Adventure', 'A pulp-fiction sci-fi classic Forbidden Planet stars Leslie Nielsen as a heroic starship captain who finds the paradise planet Al
tair-IV.'], ['Gigi', 'Movie', 'United States', '1958', 'G', '115 min', 'Classic Movies', "Gigi's an avant-garde French waif being groomed as the fille de joie
 of affluent and handsome Gaston. Soon Gigi metamorphoses into a stunning beauty."]]
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$
```

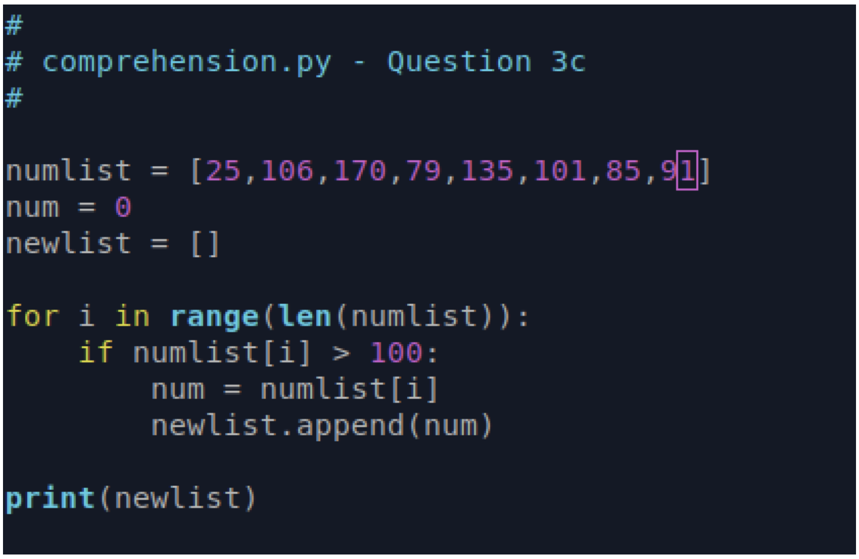


```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ vim read.py
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 read.py
['Movie', 'Movie', 'Movie']
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$
```

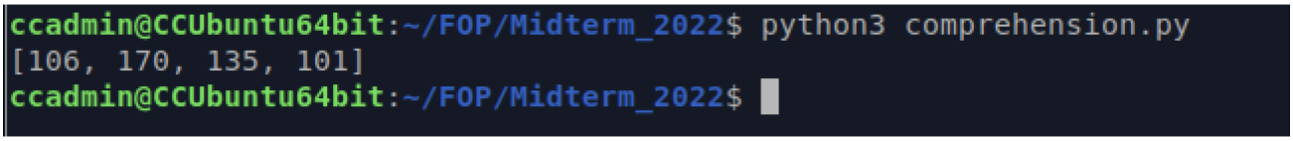
c) **(6 marks)** Write **list comprehensionsv** to generate the following:

i) If **numlist** contains a list of numbers, create a new list that contains only those numbers that are **greater than 100.** *(e.g. [42, 130, 7, 100, 101] becomes [130, 101])*

Python file: comprehension.py



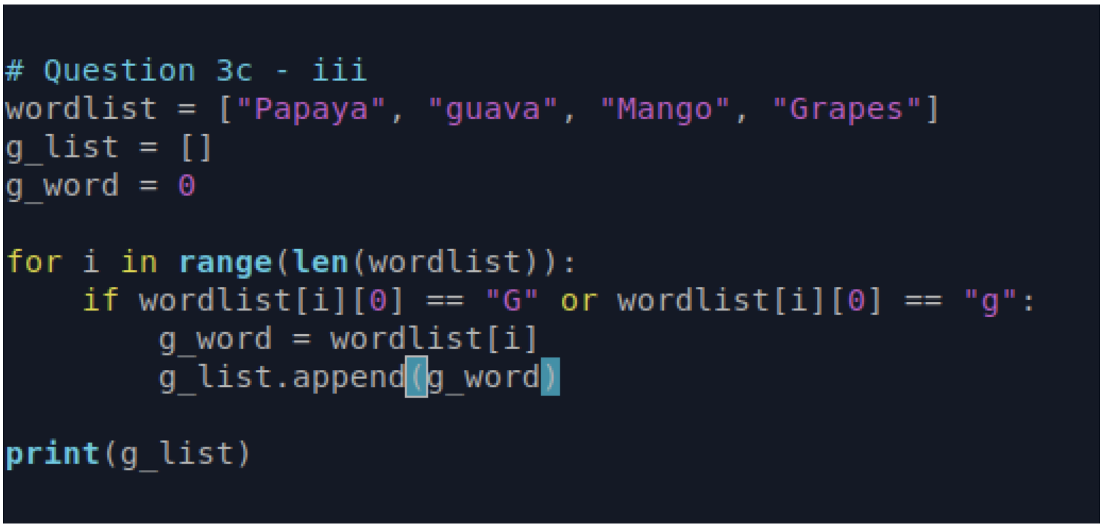
```
#
# comprehension.py - Question 3c
#

numlist = [25,106,170,79,135,101,85,91]
num = 0
newlist = []

for i in range(len(numlist)):
    if numlist[i] > 100:
        num = numlist[i]
        newlist.append(num)

print(newlist)
```



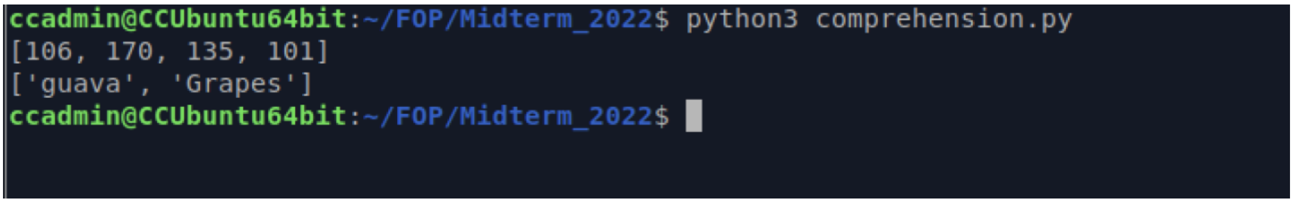
```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 comprehension.py
[106, 170, 135, 101]
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$
```

ii) If **wordlist** contains a list of strings, create a new list that has the **words that start with “g” or “G”**.
(e.g. [“Papaya”, “guava”, “Mango”, “Grapes”] becomes [“guava”,”Grapes”])

Python file: comprehension.py



```
# Question 3c - iii
wordlist = ["Papaya", "guava", "Mango", "Grapes"]
g_list = []
g_word = 0

for i in range(len(wordlist)):
    if wordlist[i][0] == "G" or wordlist[i][0] == "g":
        g_word = wordlist[i]
        g_list.append(g_word)

print(g_list)
```

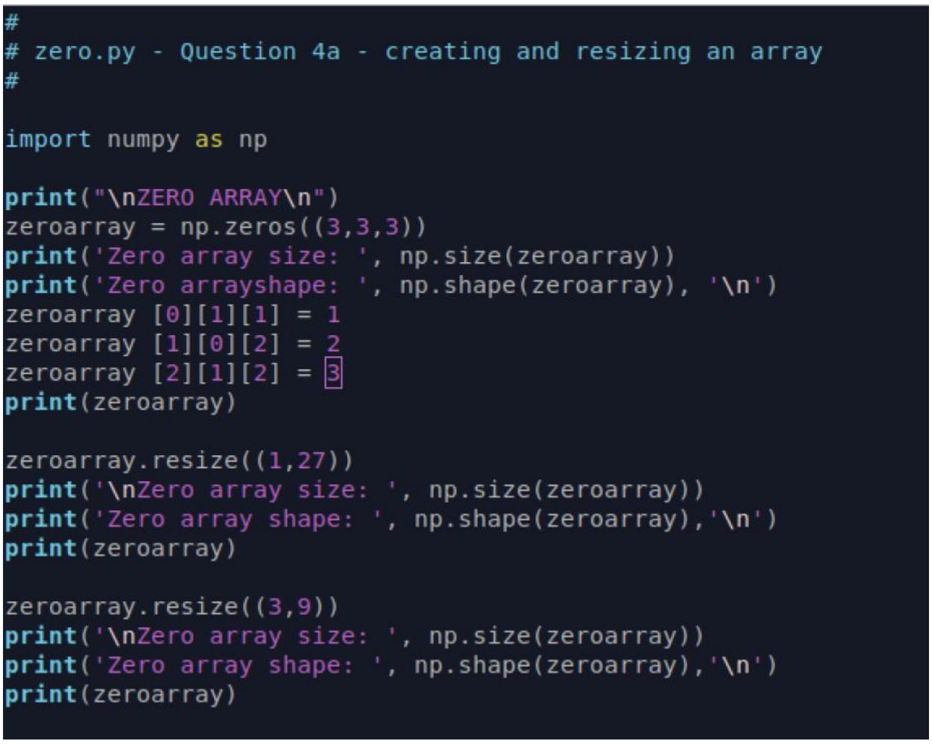


```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 comprehension.py
[106, 170, 135, 101]
['guava', 'Grapes']
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ 
```

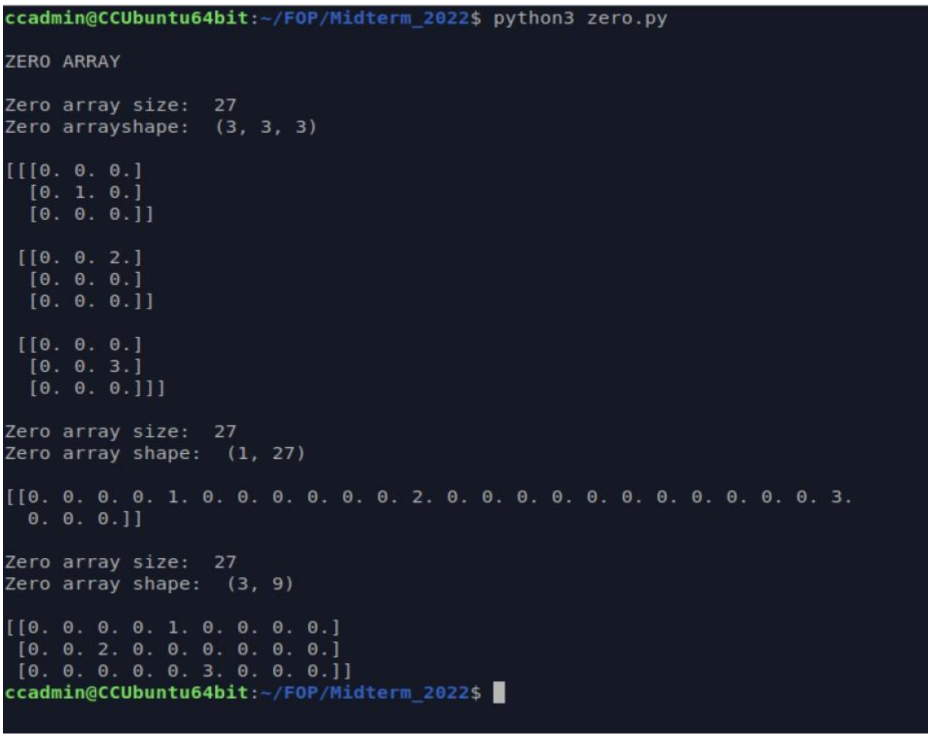
QUESTION FOUR (Total: 20 marks) Multi-Dimensional Arrays, Plotting and Unix

a) **(5 marks)** Modify the code below to resize the array into (1, 27) and (3, 9). Then set the element [0,1,1] = 1, [1,0,2] = 2 and [2,1,2] = 3. Run the code and display the results. (Submit separately as a python file)

Python file: zero.py



```
#
# zero.py - Question 4a - creating and resizing an array
#

import numpy as np

print("\nZERO ARRAY\n")
zeroarray = np.zeros((3,3,3))
print('Zero array size: ', np.size(zeroarray))
print('Zero arrayshape: ', np.shape(zeroarray), '\n')
zeroarray [0][1][1] = 1
zeroarray [1][0][2] = 2
zeroarray [2][1][2] = 3
print(zeroarray)

zeroarray.resize((1,27))
print('\nZero array size: ', np.size(zeroarray))
print('Zero array shape: ', np.shape(zeroarray),'\n')
print(zeroarray)

zeroarray.resize((3,9))
print('\nZero array size: ', np.size(zeroarray))
print('Zero array shape: ', np.shape(zeroarray),'\n')
print(zeroarray)
```

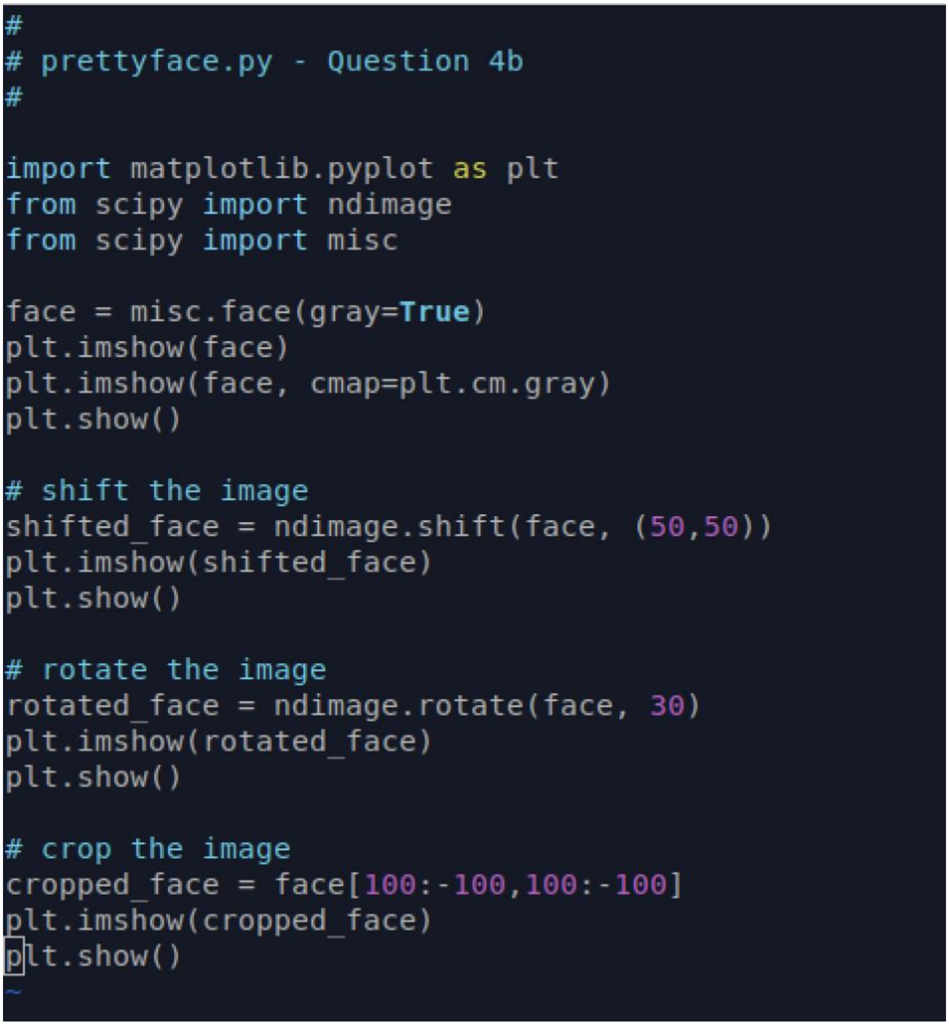


```
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ python3 zero.py

ZERO ARRAY

Zero array size:  27
Zero arrayshape:  (3, 3, 3)

[[[0. 0. 0.]
  [0. 1. 0.]
  [0. 0. 0.]]

 [[0. 0. 2.]
  [0. 0. 0.]
  [0. 0. 0.]]

 [[0. 0. 0.]
  [0. 0. 3.]
  [0. 0. 0.]]]

Zero array size:  27
Zero array shape:  (1, 27)

[[0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 3.
  0. 0. 0.]]

Zero array size:  27
Zero array shape:  (3, 9)

[[0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 2. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 3. 0. 0. 0.]]
ccadmin@CCUbuntu64bit:~/FOP/Midterm_2022$ 
```

b) **(5 marks)** Modify to the following code to shift, rotate and crop the image. You can choose the coordinates of shifting yourself.

(Submit separately as a python file)

Python file: prettyface.py



```
#
# prettyface.py - Question 4b
#

import matplotlib.pyplot as plt
from scipy import ndimage
from scipy import misc

face = misc.face(gray=True)
plt.imshow(face)
plt.imshow(face, cmap=plt.cm.gray)
plt.show()

# shift the image
shifted_face = ndimage.shift(face, (50,50))
plt.imshow(shifted_face)
plt.show()

# rotate the image
rotated_face = ndimage.rotate(face, 30)
plt.imshow(rotated_face)
plt.show()

# crop the image
cropped_face = face[100:-100,100:-100]
plt.imshow(cropped_face)
plt.show()
```

c) **(10 marks)** Write Linux commands for the following:

i. To Change the name of the existing file. (2)

Change the file named: a.py to b.py:

➔ mv a.py b.py

ii. To show the current working directory. (2)

➔ pwd

iii. To go one level up in the directory structure. (2)

➔ cd ..

iv. To change directory. (2)

```
FOP-
  Prac01
  Prac02
  Prac03-
     Subdir1
     Subdir2
```

If I am in FOP directory, I want to move to Prac01:

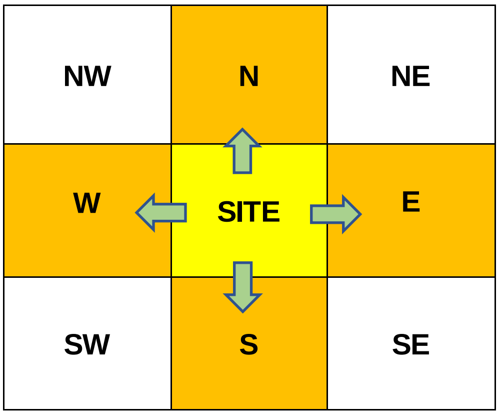
➔ cd Prac01

QUESTION FIVE (Total: 20 marks) Files and Grids

a) **(5 marks)**. Explain Moore and von Neumann neighborhood with an example.

In a 2-D grid, the neighborhood of a site of:

Von Neumann neighborhood: has four neighbors: North(N), West(W), South(S), East(E).



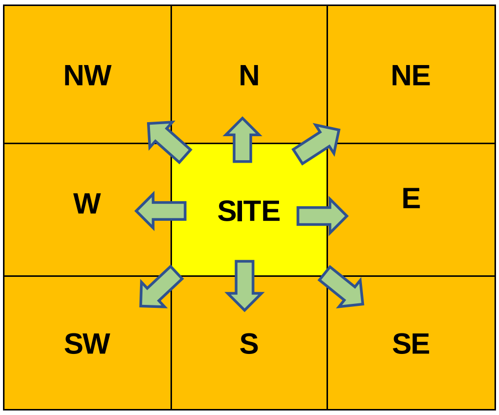
And this is how **Von Neumann neighborhood** works:

		(r,c)		

		(r, c+1)		
	(r-1, c)	(r, c)	(r+1, c)	
		(r, c-1)		

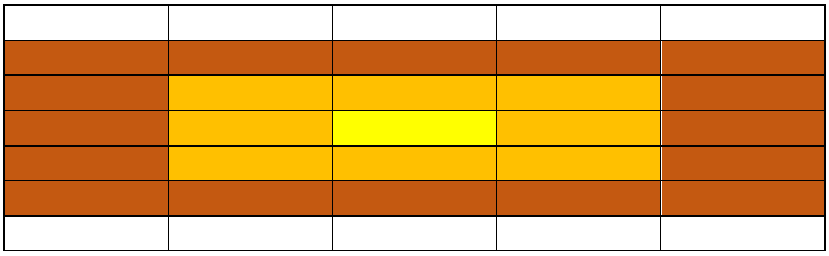
		🟫		
	🟫	🟧	🟫	
🟫	🟧	🟨	🟧	🟫
	🟫	🟧	🟫	
		🟫		

Moore neighborhood: has eight neighbors North(N), West(W), South(S), East(E), North-West(NW), South-West(SW), South-East(SE) and North-East(NE).

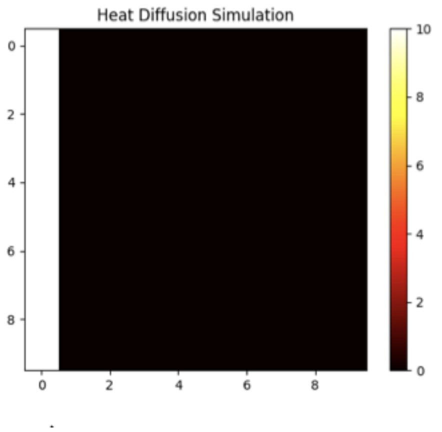


		(r,c)		

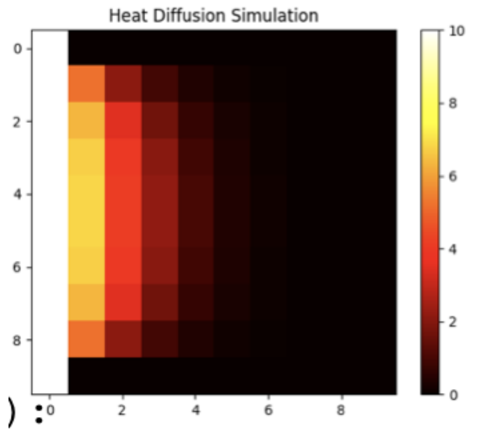
	(r-1, c+1)	(r, c+1)	(r+1, c+1)	
	(r-1, c)	(r, c)	(r+1, c)	
	(r-1, c-1)	(r, c-1)	(r+1, c-1)	



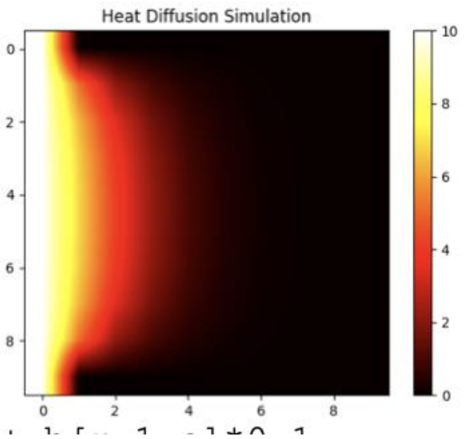
b) **(5 marks).** Refer to the heat.py in Lecture 5 (lecture slides). Explain how first graph changes to the second and then third graph.



First Graph

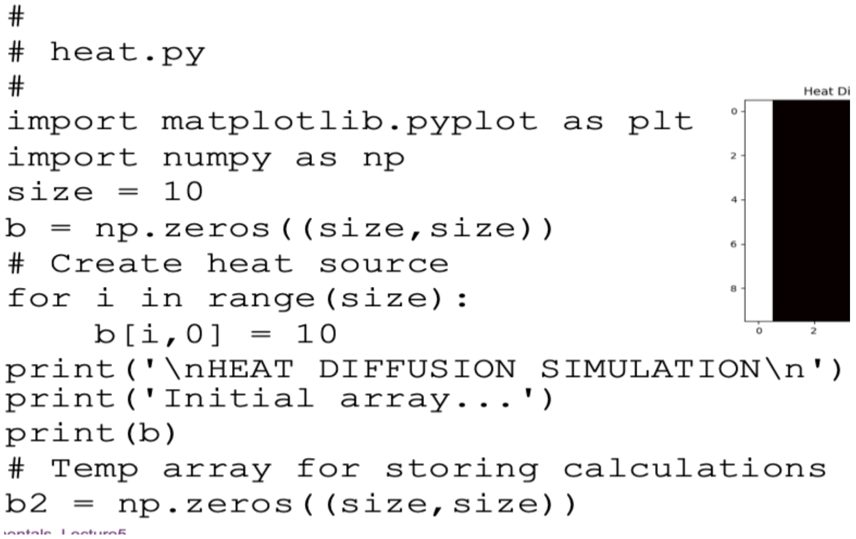


Second Graph



Third Graph

In the first graph: the heat source has just created and set up by array b with the size of 10 x 10. This code below also set the first column equal 10 (white), whereas the rest are 0 (b = np.zeros((size,size))) so the graph is purely black (heat source = 0).



```
#
# heat.py
#
import matplotlib.pyplot as plt
import numpy as np
size = 10
b = np.zeros((size,size))
# Create heat source
for i in range(size):
    b[i,0] = 10
print('\nHEAT DIFFUSION SIMULATION\n')
print('Initial array...')
print(b)
# Temp array for storing calculations
b2 = np.zeros((size,size))
```

In the second graph:

+ The heat source value starts being calculated by using:

timestep for loop, row for loop and column for loop to reach the elements in the array:

```
for timestep in range(5):
    for r in range(1, size-1):
        for c in range (1, size-1 ):
            b2[r,c] = (b[r-1,c-1]*0.1 + b[r-1,c]*0.1
                      + b[r-1,c+1]*0.1 + b[r,c-1]*0.1
                      + b[r,c]*0.2 + b[r,c+1]*0.1
                      + b[r+1,c-1]*0.1 + b[r+1,c]*0.1
                      + b[r+1,c+1]*0.1)
```

and a for loop uses for updating heat source data:

```
for i in range(size):
    b2[i,0] = 10
b = b2.copy()
```

+ The heat source figure was then plotted and display.

```
plt.title('Heat Diffusion Simulation')
```

```
plt.imshow(b2, cmap=plt.cm.hot)
```

➔ display data as an image according to the color map named: plt.cm.hot

```
plt.colorbar()
```

➔ Visualization of the heat source from scalar values to colors.

Therefore, each element has different value are displayed in the graph.

In the third graph: it is almost the same as the code for the second graph except for the interpolation method being added.

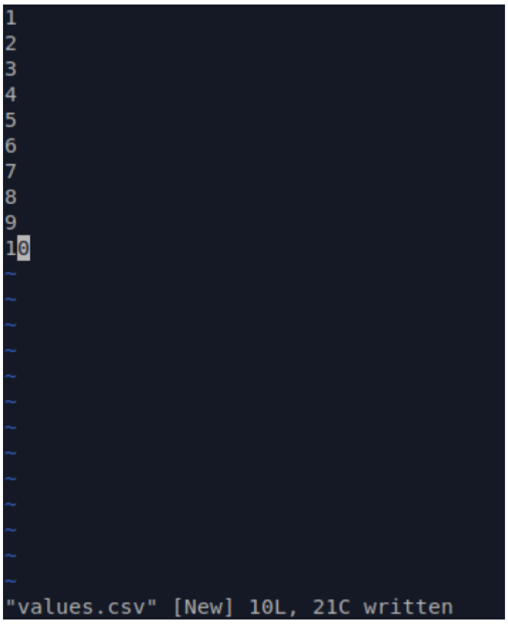
```
plt.imshow(b2, cmap=plt.cm.hot, interpolation='bilinear')
```

➔ is used to make a blurry image.

c) **(5 marks)** Write a program to enter 5 values from a file (.txt or .csv), double those values and then output them to a file (.txt or.csv).
(Hint: 1,2,3,4,5 becomes 2,4,3,8,10)

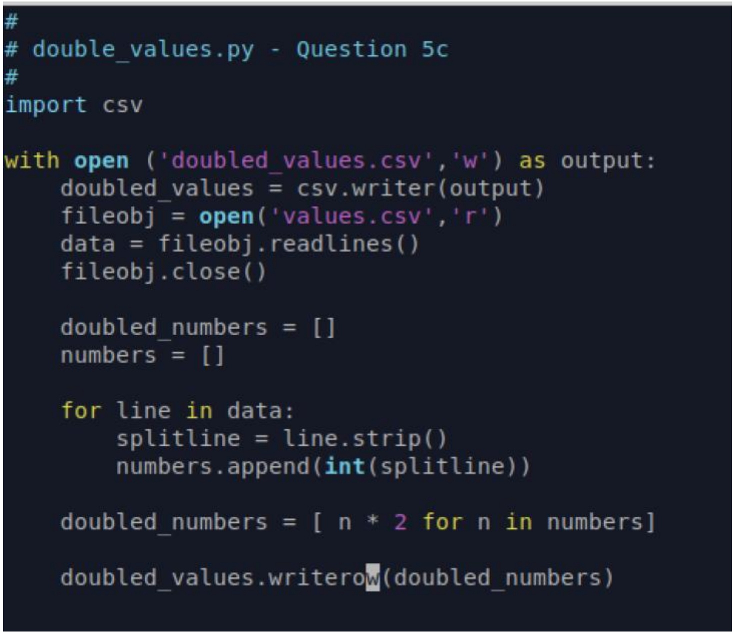
(Submit as a separate python file)

Create a value.csv file:



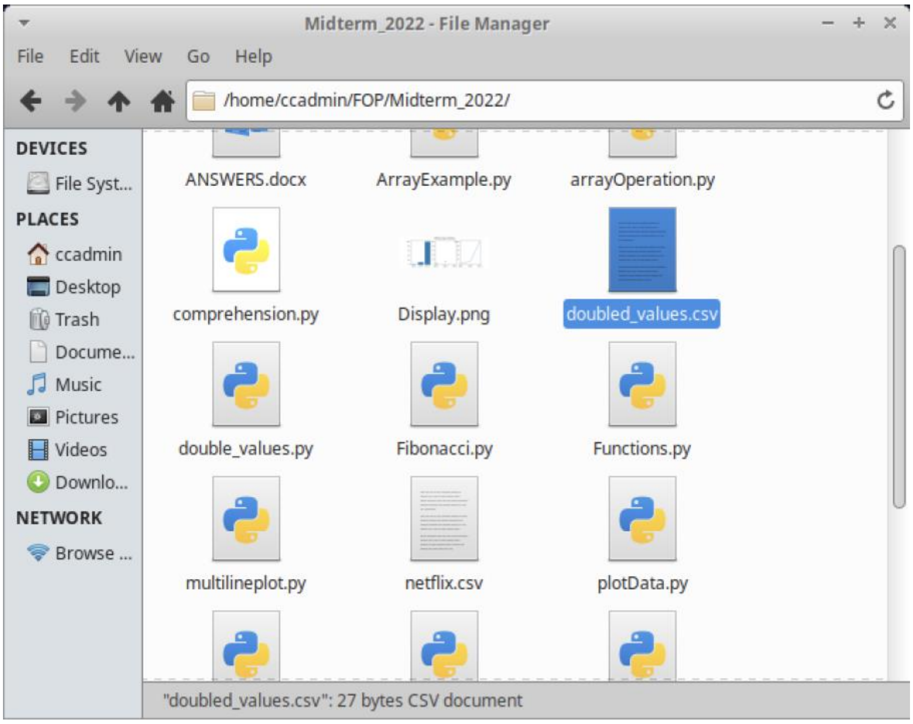
```
1
2
3
4
5
6
7
8
9
10
~
~
~
~
~
~
~
~
~
~
~
~
~
"values.csv" [New] 10L, 21C written
```

Python file: double_values.py

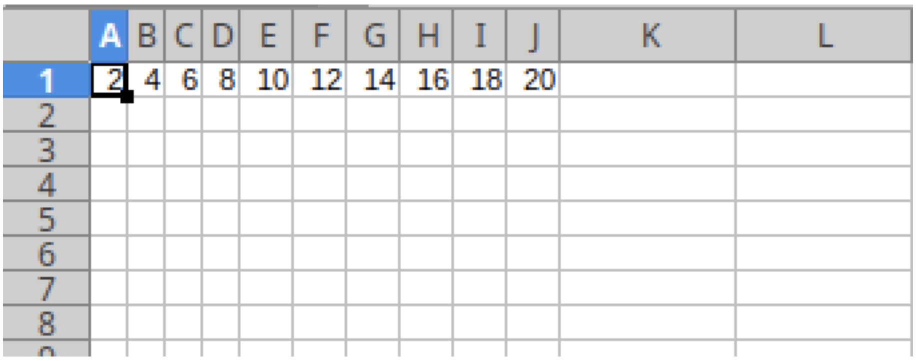


```python
#
# double_values.py - Question 5c
#
import csv

with open ('doubled_values.csv','w') as output:
    doubled_values = csv.writer(output)
    fileobj = open('values.csv','r')
    data = fileobj.readlines()
    fileobj.close()

    doubled_numbers = []
    numbers = []

    for line in data:
        splitline = line.strip()
        numbers.append(int(splitline))

    doubled_numbers = [ n * 2 for n in numbers]

    doubled_values.writerow(doubled_numbers)
```

Result:



"doubled_values.csv": 27 bytes CSV document



	A	B	C	D	E	F	G	H	I	J	K	L
1	2	4	6	8	10	12	14	16	18	20		
2												
3												
4												
5												
6												
7												
8												

d) **(3 marks)** Explain the meaning of file1.close () in detail.

file1.close(): is a method using to close the file (it can be .csv or .txt, etc …) after we open it. If we don't close file, it will:

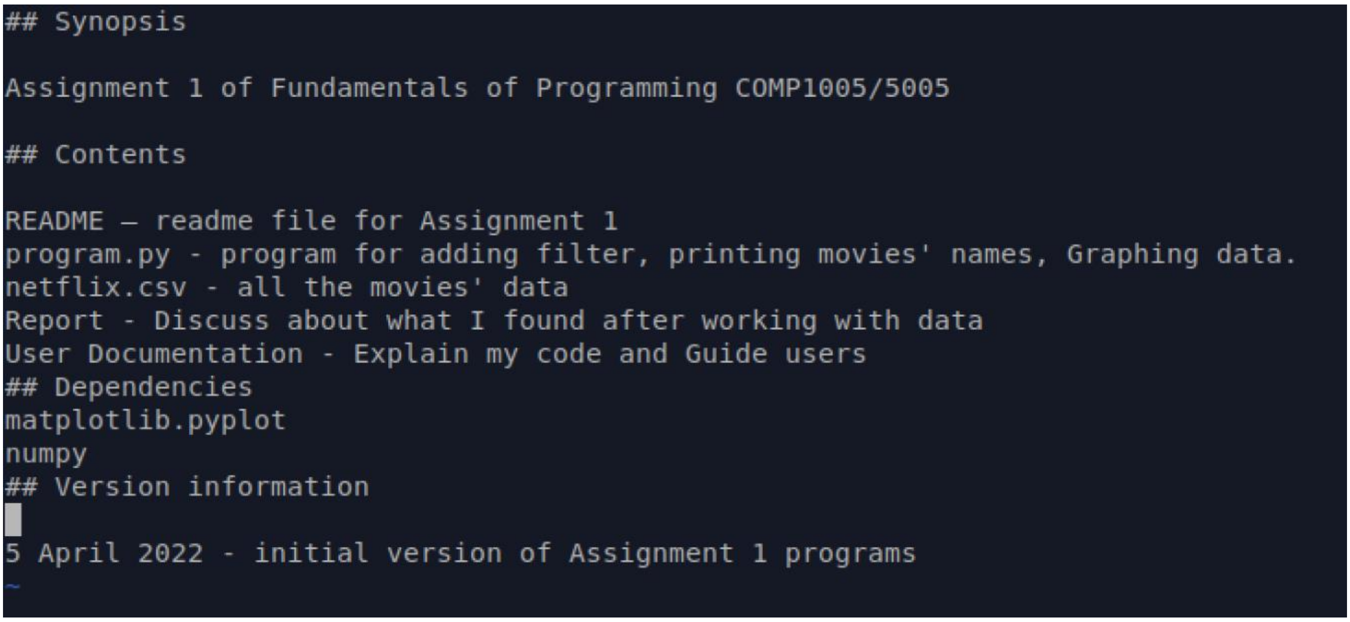
- waste the program resources and can slow down our program as well. This is simply because, too many things are opened at the same time, which in turn consuming more space of our computer, and will slow down the program's speed.
- prevent the interactions of other files (when we import a file, etc…)
- lead to inconsistences when running our codes on the different Python versions.
- The data will not be written until the file is close. It means that all works we have done will not be saved, and we will lose all our data.

e) **(2 marks)** What is the purpose of documentation along with coding? How have we done it with our Practicals and PracTests?

Documentation is the text associated with the your code. It is usually use for explaining what is your code doing? Why you choose this way to write code? and How to use it?

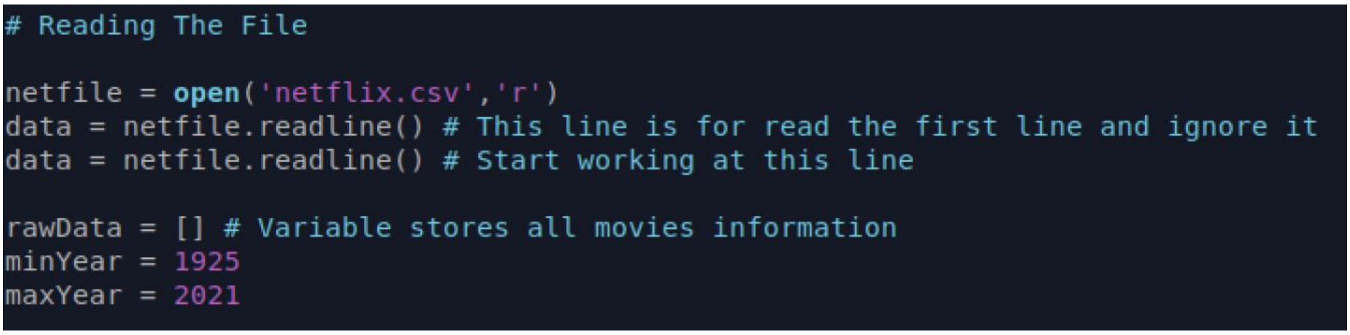
In Practicals and Practests, we have had:

- README file: to point out the content of the code file and tell the reader what we are doing in the files, the information about the code (dependencies, date and time, version information, etc…).



```
## Synopsis

Assignment 1 of Fundamentals of Programming COMP1005/5005

## Contents

README – readme file for Assignment 1
program.py - program for adding filter, printing movies' names, Graphing data.
netflix.csv - all the movies' data
Report - Discuss about what I found after working with data
User Documentation - Explain my code and Guide users
## Dependencies
matplotlib.pyplot
numpy
## Version information

5 April 2022 - initial version of Assignment 1 programs
~
```

- Comment the code by "#": to explain the idea of the code, the purpose of it and tell the reader what we will do.



```
# Reading The File

netfile = open('netflix.csv','r')
data = netfile.readline() # This line is for read the first line and ignore it
data = netfile.readline() # Start working at this line

rawData = [] # Variable stores all movies information
minYear = 1925
maxYear = 2021
```