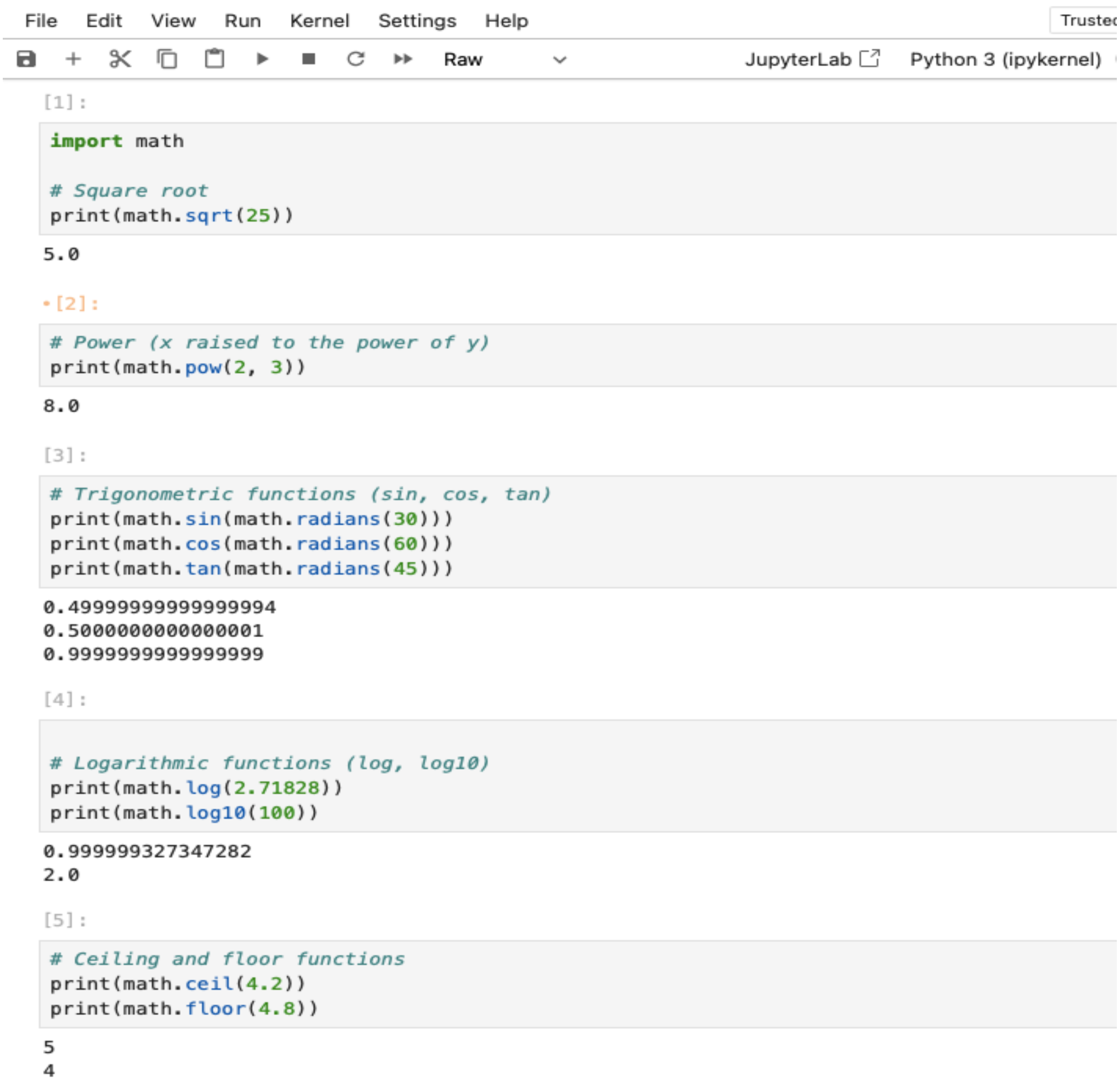
Assignment-8

Name-Ajay Chaudhary
Batch-Data Engineering (Batch-01)

Module –

In Python, a module is a file containing Python definitions and statements. The file name is the module name with the suffix .py added. Modules allow you to logically organize your Python code into reusable units. Each module can define variables, functions, and classes, and can also include runnable code.



[1]:
```
import math

# Square root
print(math.sqrt(25))
```
```
5.0
```

• [2]:
```
# Power (x raised to the power of y)
print(math.pow(2, 3))
```
```
8.0
```

[3]:
```
# Trigonometric functions (sin, cos, tan)
print(math.sin(math.radians(30)))
print(math.cos(math.radians(60)))
print(math.tan(math.radians(45)))
```
```
0.49999999999999994
0.5000000000000001
0.9999999999999999
```

[4]:
```

# Logarithmic functions (log, log10)
print(math.log(2.71828))
print(math.log10(100))
```
```
0.999999327347282
2.0
```

[5]:
```
# Ceiling and floor functions
print(math.ceil(4.2))
print(math.floor(4.8))
```
```
5
4
```

[6]:

```
# Constants
print(math.pi)
print(math.e)
```

```
3.141592653589793
2.718281828459045
```

CSV

CSV stands for “Comma Separated Values.” It is the simplest form of storing data in tabular form as plain text. It is important to know to work with CSV because we mostly rely on CSV data in our day-to-day lives as data scientists.

List of Methods to Read a CSV File in Python
1.Read CSV file using csv.reader
2.Read CSV file using .readlines() function
3.Read CSV file using Pandas
4.Read CSV file using csv.DictReader

Steps to Read CSV Files in python Using csv.reader

```
import csv #import the csv library
```

•[25]:

```
#open the csv file
file=open('/Users/ajaychaudhary/Downloads/test.csv')
type(file)
```

[25]:

```
_io.TextIOWrapper
```

•[26]:

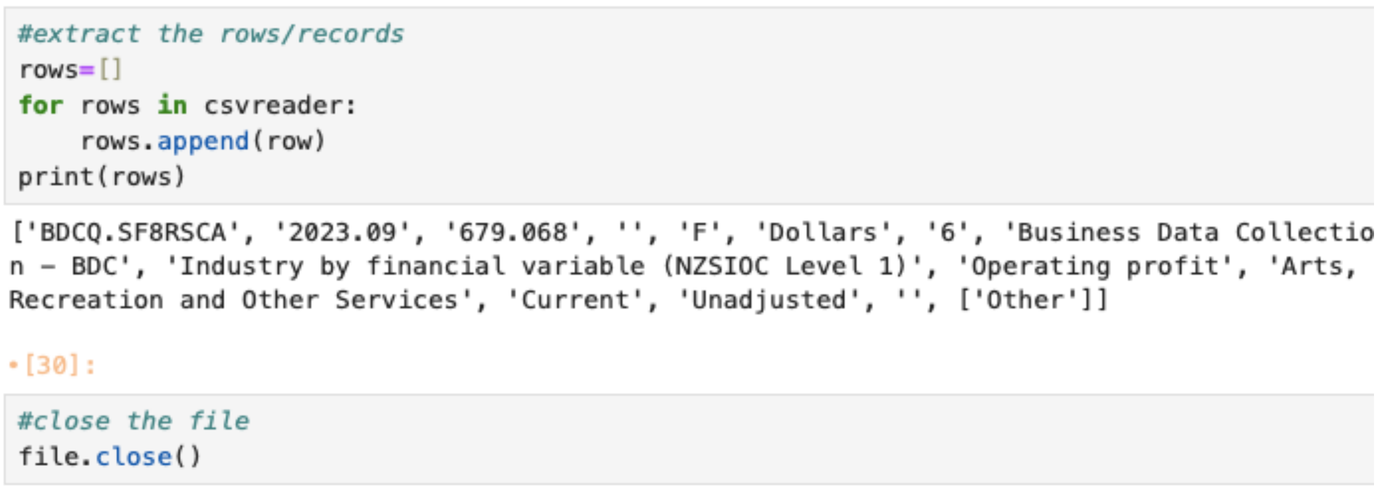
```
#using the csv.reader object to read the csv file
csvreader=csv.reader(file)
```

•[27]:

```
#extract the field names
header=[]
header=next(csvreader)
header
```

[27]:

```
['Series_reference',
 'Period',
 'Data_value',
```



```
#extract the rows/records
rows=[]
for rows in csvreader:
    rows.append(row)
print(rows)
```

```
['BDCQ.SF8RSCA', '2023.09', '679.068', '', 'F', 'Dollars', '6', 'Business Data Collectio
n - BDC', 'Industry by financial variable (NZSIOC Level 1)', 'Operating profit', 'Arts,
Recreation and Other Services', 'Current', 'Unadjusted', '', ['Other']]
```

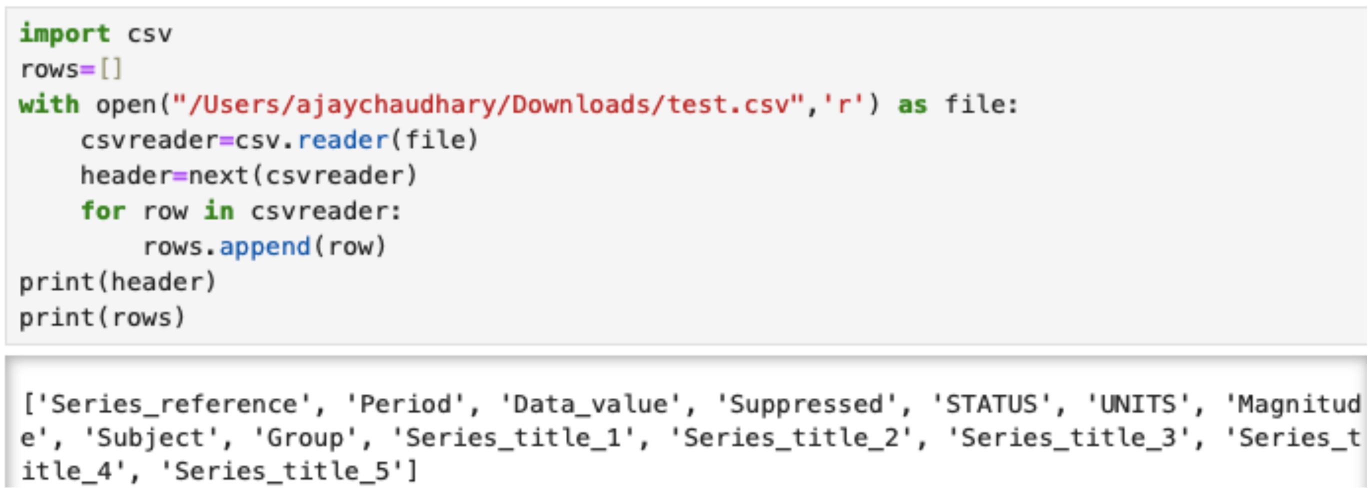
•[30]:

```
#close the file
file.close()
```

Reading CSV file in python

Modes-

'r'- to read an existing file
'w'-create a new file if the given file doesn't exist and write to it.
'a'- append to existing file content
'+'- to create a new file for reading and writing

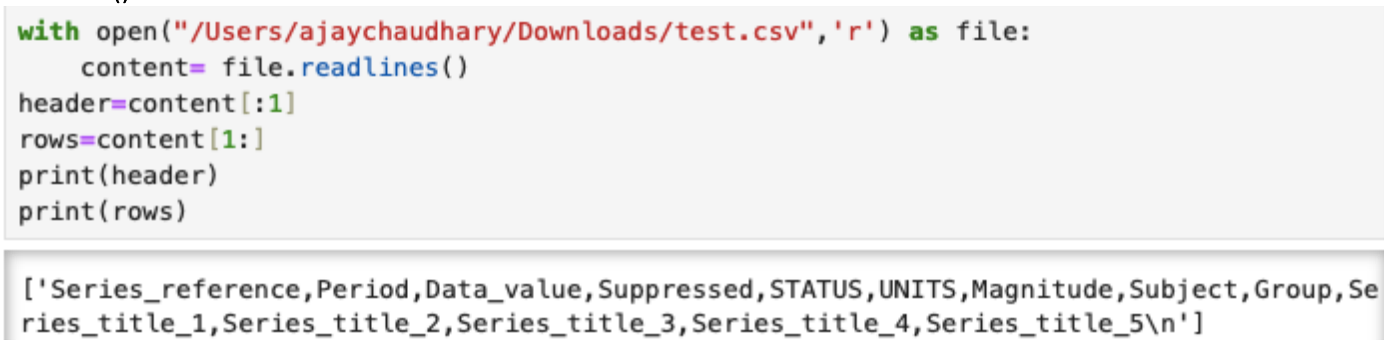


```
import csv
rows=[]
with open("/Users/ajaychaudhary/Downloads/test.csv",'r') as file:
    csvreader=csv.reader(file)
    header=next(csvreader)
    for row in csvreader:
        rows.append(row)
print(header)
print(rows)
```

```
['Series_reference', 'Period', 'Data_value', 'Suppressed', 'STATUS', 'UNITS', 'Magnitud
e', 'Subject', 'Group', 'Series_title_1', 'Series_title_2', 'Series_title_3', 'Series_t
itle_4', 'Series_title_5']
```

Reading CSV file in python using.readlines()

.readlines()method returns all the lines in a file as a list. Each item on the list is a row of our CSV file.



```
with open("/Users/ajaychaudhary/Downloads/test.csv",'r') as file:
    content= file.readlines()
header=content[:1]
rows=content[1:]
print(header)
print(rows)
```

```
['Series_reference,Period,Data_value,Suppressed,STATUS,UNITS,Magnitude,Subject,Group,Se
ries_title_1,Series_title_2,Series_title_3,Series_title_4,Series_title_5\n']
```

Reading files using pandas

[34]:

```
import pandas as pd
```

[35]:

```
data=pd.read_csv("/Users/ajaychaudhary/Downloads/test.csv")
data
```

[35]:

	Series_reference	Period	Data_value	Suppressed	STATUS	UNITS	Magnitude	Subject
0	BDCQ.SF1AA2CA	2016.06	1116.386	NaN	F	Dollars	6	Business Data Collection - BDC
1	BDCQ.SF1AA2CA	2016.09	1070.874	NaN	F	Dollars	6	Business Data Collection - BDC

[36]:

```
data.columns
```

[36]:

```
Index(['Series_reference', 'Period', 'Data_value', 'Suppressed', 'STATUS',
       'UNITS', 'Magnitude', 'Subject', 'Group', 'Series_title_1',
       'Series_title_2', 'Series_title_3', 'Series_title_4', 'Series_title_5'],
      dtype='object')
```

[38]:

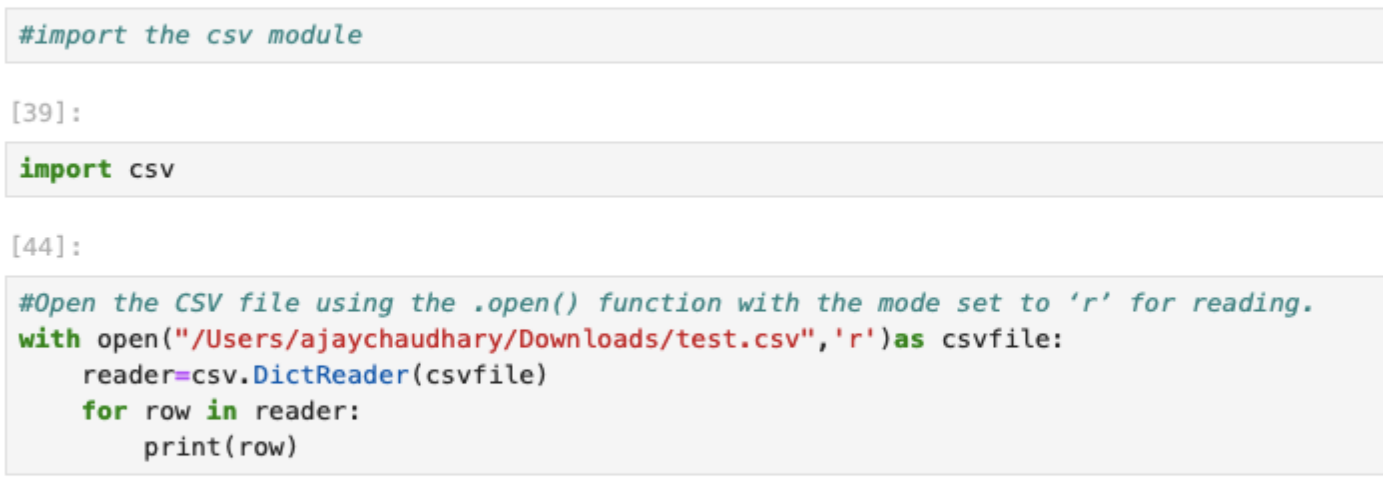
```
data.Subject
```

[38]:

```
0        Business Data Collection - BDC
```

Reading CSV files in python using csv.DictReader

Dict is a hash table of keys and values structured in Python. The dict() method is used to create a dictionary object from either a specified set or iterables of keys and values. The csv module .DictReader is used to read CSV files. Here's how you can do that.

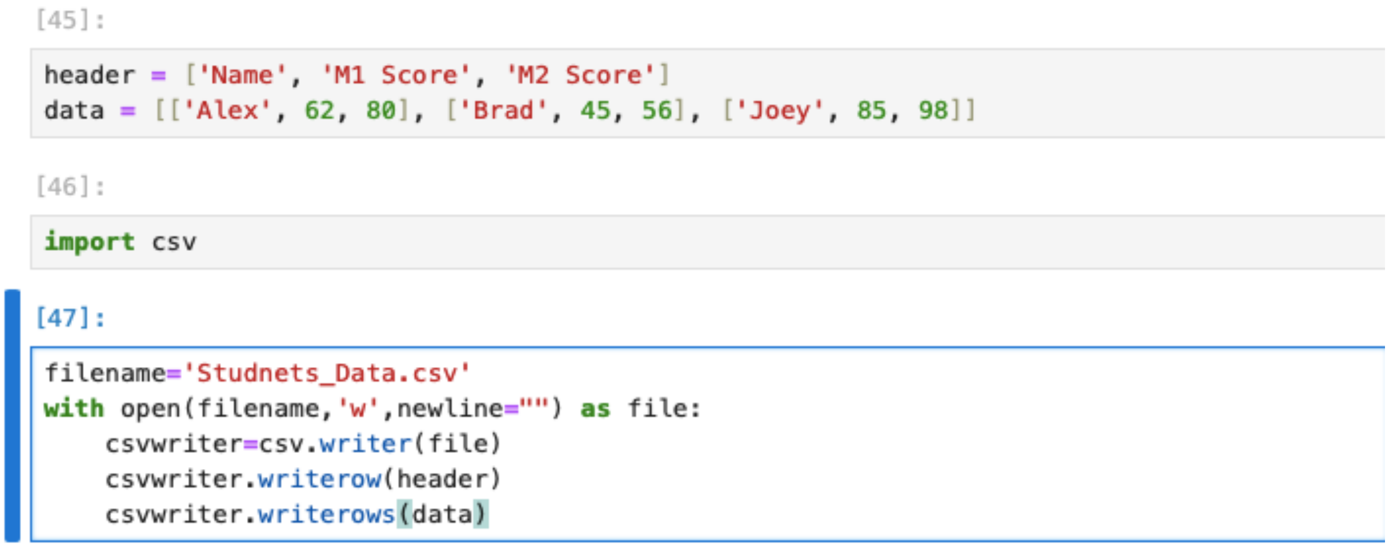


```
#import the csv module

[39]:

import csv

[44]:

#Open the CSV file using the .open() function with the mode set to 'r' for reading.
with open("/Users/ajaychaudhary/Downloads/test.csv",'r')as csvfile:
    reader=csv.DictReader(csvfile)
    for row in reader:
        print(row)
```

methods to write a csv file in python

1. write csv file using csv.writer
2. write csv file using writelines() function
3. write csv file using pandas
4. write csv file using csv.DictWriter

write csv file using csv.writer



```
[45]:

header = ['Name', 'M1 Score', 'M2 Score']
data = [['Alex', 62, 80], ['Brad', 45, 56], ['Joey', 85, 98]]

[46]:

import csv

[47]:

filename='Studnets_Data.csv'
with open(filename,'w',newline="") as file:
    csvwriter=csv.writer(file)
    csvwriter.writerow(header)
    csvwriter.writerows(data)
```

Write CSV File Using .writelines()

.writelines() iterates through each list, converts the list elements to a string, and then writes it to the csv file.

```
header=['Name','M1 Score','M2 Score']
data=[['Alex',62,80],['Brad',45,56],['Joey',85,98]]
filename='Student_scores.csv'
with open(filename,'w') as file:
    for header in header:
        file.write(str(header)+', ')
    file.write('n')
    for row in data:
        for x in row:
            file.write(str(x)+', ')
        file.write('n')
```

write CSV using Pandas

[49]:

```
import pandas as pd
```

[50]:

```
header = ['Name', 'M1 Score', 'M2 Score']
data = [['Alex', 62, 80], ['Brad', 45, 56], ['Joey', 85, 98]]
data = pd.DataFrame(data, columns=header)
data.to_csv('Stu_data.csv', index=False)
```

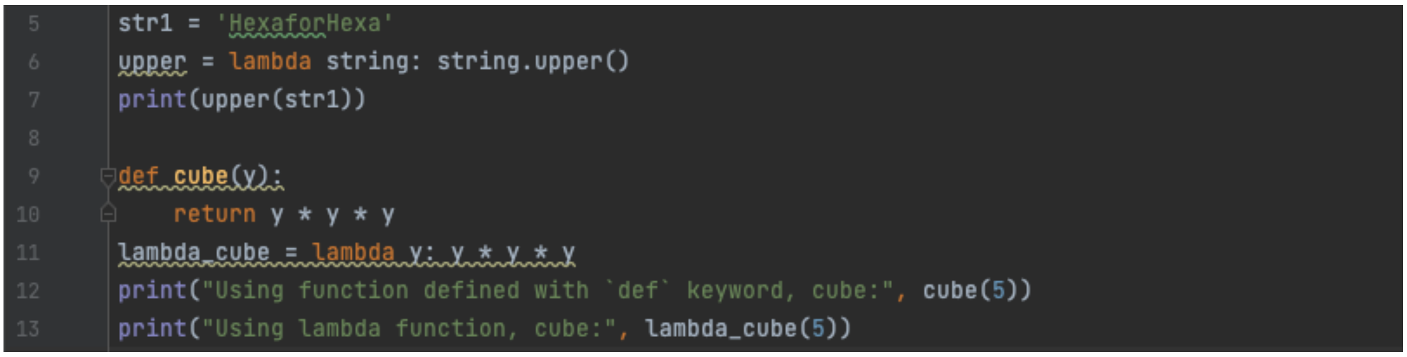
Write CSV file using csv.DictWriter

You can write data into a CSV file using the csv module .DictReader following the below steps.

```
import csv
with open('Students_Data.csv','w',newline='')as csvfile:
    data=[{'Name':'Alex','M1 Score':62,'M2 Score':80},
          {'Name':'Brad','M1 Score':45,'M2 Score':56},
          {'Name':'Joey','M1 Score':85,'M2 Score':98}]
fieldnames=['Name','M1 Score','M2 Score']
writer=csv.DictWriter(csvfile,fieldnames=fieldnames)
writer.writeheader()
writer.writerows(data)
```

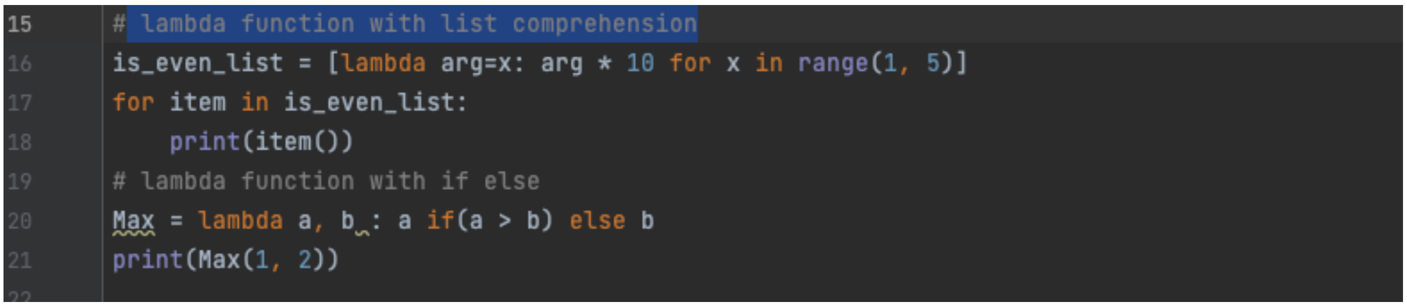
Python Lambda Functions

Python Lambda Functions are anonymous functions means that the function is without a name.
This function can have any number of arguments but only one expression, which is evaluated and returned.
One is free to use lambda functions wherever function objects are required.
You need to keep in your knowledge that lambda functions are syntactically restricted to a single expression.



```python
str1 = 'HexaforHexa'
upper = lambda string: string.upper()
print(upper(str1))

def cube(y):
    return y * y * y
lambda_cube = lambda y: y * y * y
print("Using function defined with `def` keyword, cube:", cube(5))
print("Using lambda function, cube:", lambda_cube(5))
```

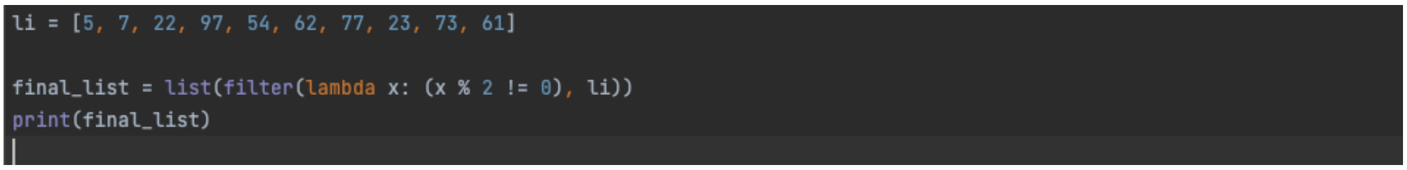
lambda function with list comprehension



```python
# lambda function with list comprehension
is_even_list = [lambda arg=x: arg * 10 for x in range(1, 5)]
for item in is_even_list:
    print(item())
# lambda function with if else
Max = lambda a, b : a if(a > b) else b
print(Max(1, 2))
```

Filter() function

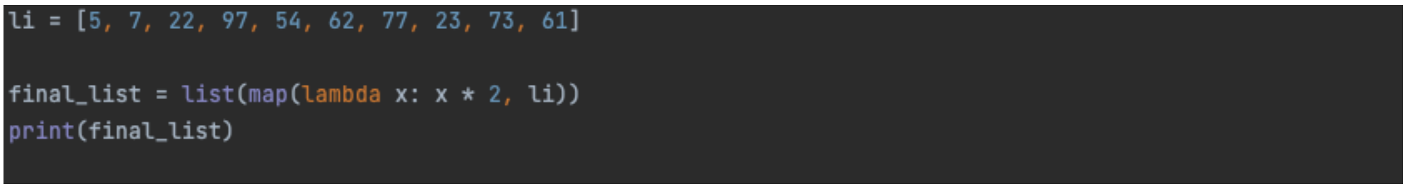
The filter() function in Python takes in a function and a list as arguments.
This offers an elegant way to filter out all the elements of a sequence “sequence”,for which the function returns True.



```python
li = [5, 7, 22, 97, 54, 62, 77, 23, 73, 61]

final_list = list(filter(lambda x: (x % 2 != 0), li))
print(final_list)
```

Map() function

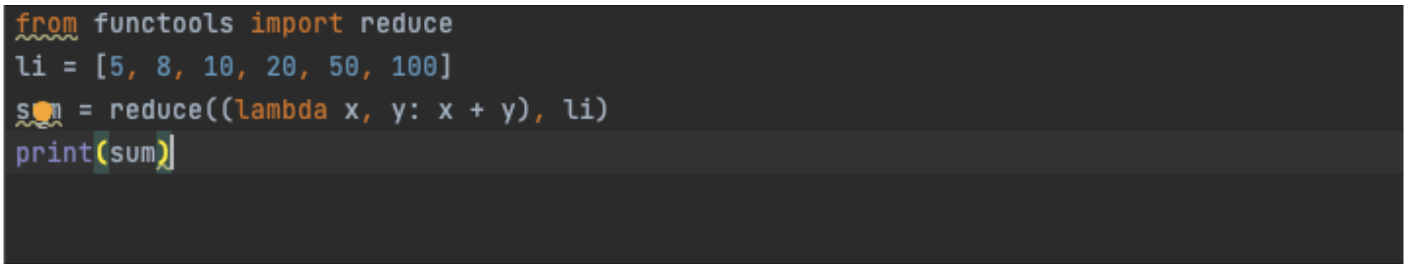
The map() function in Python takes in a function and a list as an argument.
The function is called with a lambda function and a list and a new list is returned which contains all the lambda-modified items returned by that function for each item.



```python
li = [5, 7, 22, 97, 54, 62, 77, 23, 73, 61]

final_list = list(map(lambda x: x * 2, li))
print(final_list)
```

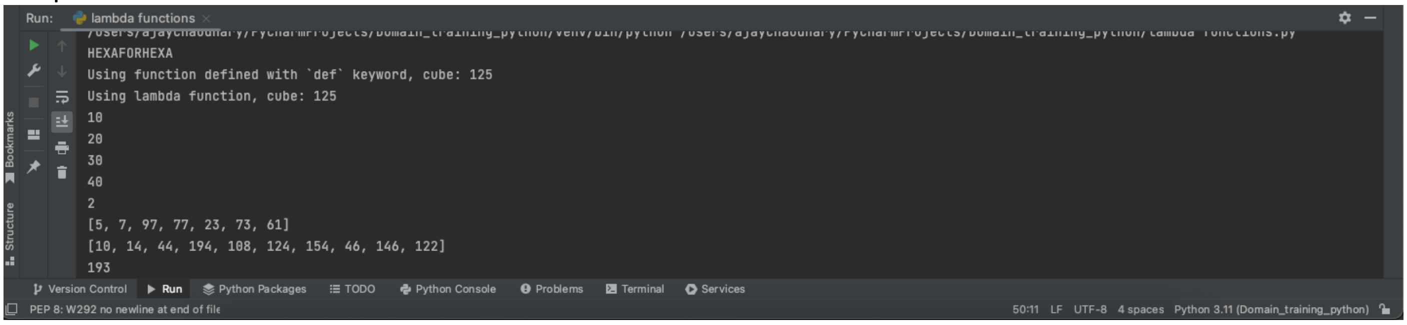
Reduce() function

The reduce() function in Python takes in a function and a list as an argument.
The function is called with a lambda function and an iterable and a new reduced result is returned.



```python
from functools import reduce
li = [5, 8, 10, 20, 50, 100]
sum = reduce((lambda x, y: x + y), li)
print(sum)
```

Output:



```
/Users/ajaychaudhary/PycharmProjects/Domain_training_python/venv/bin/python /Users/ajaychaudhary/PycharmProjects/Domain_training_python/lambda functions.py
HEXAFORHEXA
Using function defined with `def` keyword, cube: 125
Using lambda function, cube: 125
10
20
30
40
2
[5, 7, 97, 77, 23, 73, 61]
[10, 14, 44, 194, 108, 124, 154, 46, 146, 122]
193
```