Akilesh K

k.akilesh123@gmail.com

Data engineering - Batch 1

Date: 30-01-24

DAY 8-PYTHON-CSV,LAMBDA

FILE HANDLING

Python supports file handling and allows users to handle files to read and write files, along with many other file handling options, to operate on files.

In [21]:
```
file2 = open('file2.txt', 'w')


file2.write('Programming is Fun.\n')
file2.write('Programiz for beginners\n')
```

Out[21]: 24

In [22]:
```
file1 = open("file2.txt")


read_content = file1.read()
print(read_content)
```

Read Data from CSV File into Python List

Pandas is most commonly used for data wrangling and data manipulation purposes, and NumPy objects are primarily used to create arrays or matrices that can be applied to DL or ML models

In [2]:
```
import numpy as np
import pandas as pd

ser=pd.Series()
print("pandas series: ",ser)

data=np.array(['a','b','c','d','e'])

ser=pd.Series(data)
print("pandas series: ",ser)
```

```
pandas series:  Series([], dtype: object)
pandas series:  0    a
1    b
2    c
3    d
4    e
dtype: object
```

In [3]:
```
import pandas as pd
df =pd.DataFrame()
print(df)


lst = ['Python', 'For', 'Python', 'is', 'portal', 'for', 'Python']

df =pd.DataFrame(lst)
print(df)
```

```
Empty DataFrame
Columns: []
Index: []
        0
0  Python
1     For
2  Python
3      is
4  portal
5     for
6  Python
```

In [4]:
```
import pandas as pd

data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

df = pd.DataFrame(data)

print(df)
```

```
   calories  duration
0       420        50
1       380        40
2       390        45
```

In [14]:
```
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)
```

jupyter Students_Data.csv 2 minutes ago

File Edit View Language

```
1 Name,M1 Score,M2 Score
2 Alex,62,80
3 Brad,45,56
4 Joey,85,98
5
```

In [5]:
```
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')
```

jupyter Student_scores.csv 18 minutes ago

File Edit View Language

```
1 Name, M1 Score, M2 Score, nAlex, 62, 80, nBrad, 45, 56, nJoey, 85, 98, n
```

In [6]:
```
import pandas as pd
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)
```

jupyter Stu_data.csv ✔ 14 minutes ago

File Edit View Language

```
1 Name,M1 Score,M2 Score
2 Alex,62,80
3 Brad,45,56
4 Joey,85,98
5
```

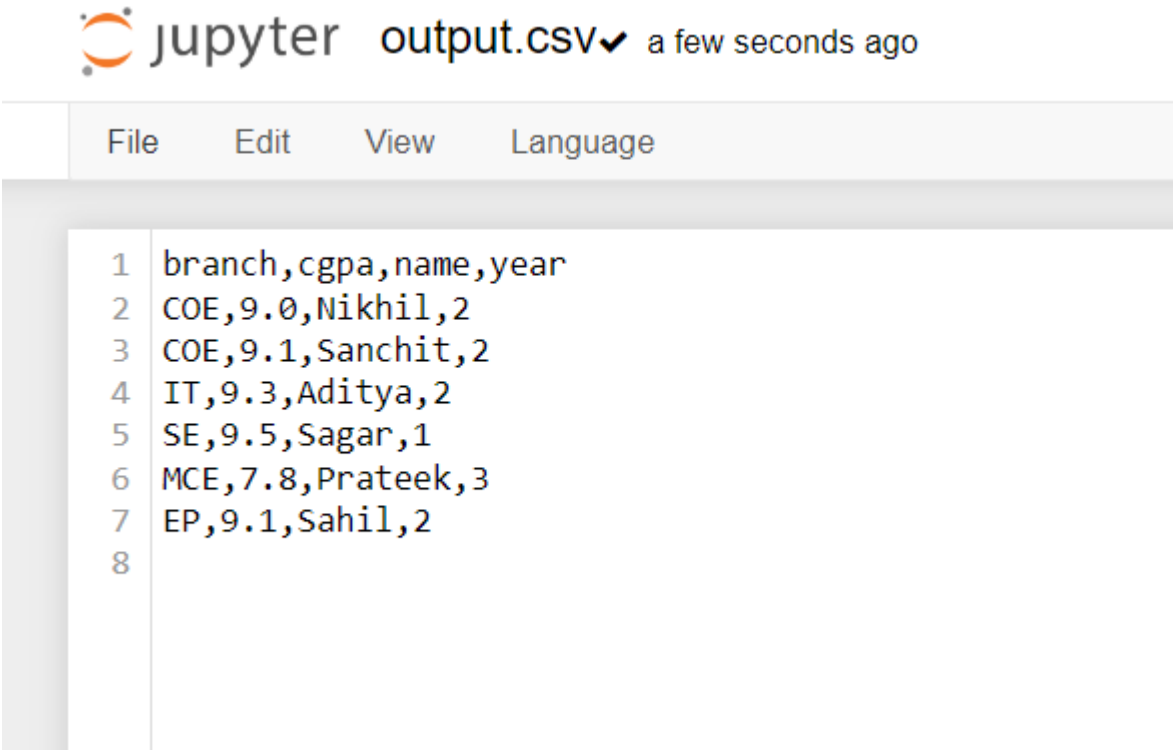
In [15]:
```
import pandas as pd


dict = {
    'series': ['Friends', 'Money Heist', 'Marvel'],
    'episodes': [200, 50, 45],
    'actors': [' David Crane', 'Alvaro', 'Stan Lee']
}


df = pd.DataFrame(dict)
print(df)
```

```
        series  episodes        actors
0      Friends       200   David Crane
1  Money Heist        50        Alvaro
2       Marvel        45      Stan Lee
```

In [18]:
```
import csv

with open('industry.csv') as csvfile:

    readCSV = csv.reader(csvfile, delimiter=',')
    for row in readCSV:
        print(row)
        print(row[0])
        print("\n")
```

```
['Skilled Labor']
Skilled Labor


['Technology']
Technology


['Telecommunications']
Telecommunications


['Transportation/Logistics']
Transportation/Logistics
```

In [19]:
```
import pandas as pd

mydict = [
    {'branch': 'COE', 'cgpa': '9.0', 'name': 'Nikhil', 'year': '2'},
    {'branch': 'COE', 'cgpa': '9.1', 'name': 'Sanchit', 'year': '2'},
    {'branch': 'IT', 'cgpa': '9.3', 'name': 'Aditya', 'year': '2'},
    {'branch': 'SE', 'cgpa': '9.5', 'name': 'Sagar', 'year': '1'},
    {'branch': 'MCE', 'cgpa': '7.8', 'name': 'Prateek', 'year': '3'},
    {'branch': 'EP', 'cgpa': '9.1', 'name': 'Sahil', 'year': '2'}
]


df = pd.DataFrame(mydict)


df.to_csv('output.csv', index=False)
```



Processing Python Lists

In [23]:
```
List = []
print("Initial blank List: ")
print(List)

# in the List
List.append(1)
List.append(2)
List.append(4)
print("\nList after Addition of Three elements: ")
print(List)
```

```
Initial blank List:
[]

List after Addition of Three elements:
[1, 2, 4]
```

In [24]:
```
mylist = [1, 2, 3, 4, 5, 'Geek', 'Python']
mylist.reverse()
print(mylist)
```

```
['Python', 'Geek', 5, 4, 3, 2, 1]
```

In [25]:

```
List = [1, 2, 3, 4, 5]


List.pop()
print("\nList after popping an element: ")
print(List)


List.pop(2)
print("\nList after popping a specific element: ")
print(List)
```

```
List after popping an element:
[1, 2, 3, 4]

List after popping a specific element:
[1, 2, 4]
```

In [26]:

```
List = ['G', 'E', 'E', 'K', 'S', 'F',
        'O', 'R', 'G', 'E', 'E', 'K', 'S']
print("Initial List: ")
print(List)


Sliced_List = List[3:8]
print("\nSlicing elements in a range 3-8: ")
print(Sliced_List)
```

```
Initial List:
['G', 'E', 'E', 'K', 'S', 'F', 'O', 'R', 'G', 'E', 'E', 'K', 'S']

Slicing elements in a range 3-8:
['K', 'S', 'F', 'O', 'R']
```

Lambda Functions in Python

In [1]:

```
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))
```

```
Using function defined with `def` keyword, cube: 125
Using lambda function, cube: 125
```

Filter

In [27]:
```
li = [5, 7, 22, 97, 54, 62, 77, 23, 73, 61]

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

```
[5, 7, 97, 77, 23, 73, 61]
```

MAP

In [28]:
```
li = [5, 7, 22, 97, 54, 62, 77, 23, 73, 61]

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

```
[10, 14, 44, 194, 108, 124, 154, 46, 146, 122]
```

REDUCE

In [29]:
```
from functools import reduce
li = [5, 8, 10, 20, 50, 100]
sum = reduce((lambda x, y: x + y), li)
print(sum)
```

```
193
```

In [30]:
```
import functools
lis = [1, 3, 5, 6, 2, ]
print("The maximum element of the list is : ", end="")
print(functools.reduce(lambda a, b: a if a > b else b, lis))
```

```
The maximum element of the list is : 6
```

Usage of Lambda Functions

Referred to as anonymous functions. Lambda functions are efficient whenever you want to create a function that will only contain simple expressions – that is, expressions that are usually a single line of a statement.

in contrast to regular functions, they can capture, either by value or by reference, variables from the scope where they are defined.