

Assignment-3

- 1) File IO using python
- 2) Read data from CSV file into python list
- 3) Processing python lists
- 4) Lambda functions in python
- 5) Usage of lambda functions
- 6) Filter data in python lists using map, filter and reduce

1) File IO using python

Python allows users to handle files i.e., to read and write files along with other operations.

Creating a file in Python:

- 1) Create a file using open()
- 2) Two parameters: filename, access mode to create a file

Mode	Description
r	Open a file for reading.
w	Opens a file for writing only. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing.
r+	opens for reading and writing (cannot truncate a file)
w+	For writing and reading (can truncate a file)
a	Open for appending at the end of the file without truncating it. Creates a new file if it does not exist.
t	Open in text mode.
b	Open in binary mode.
x	Open for exclusive creation, failing if the file already

seek() :change or move cursor to place where data must be read or written in the file

tell():return current position of file pointer from beginning of file

read():returns file content in the form of string

readLine():Reads single line

readLines():read file into list

writelines():write list of strings to file

close(): closes the file

write():write specified string to file

append(): It adds the data at the end of file.

a) Reading a file:

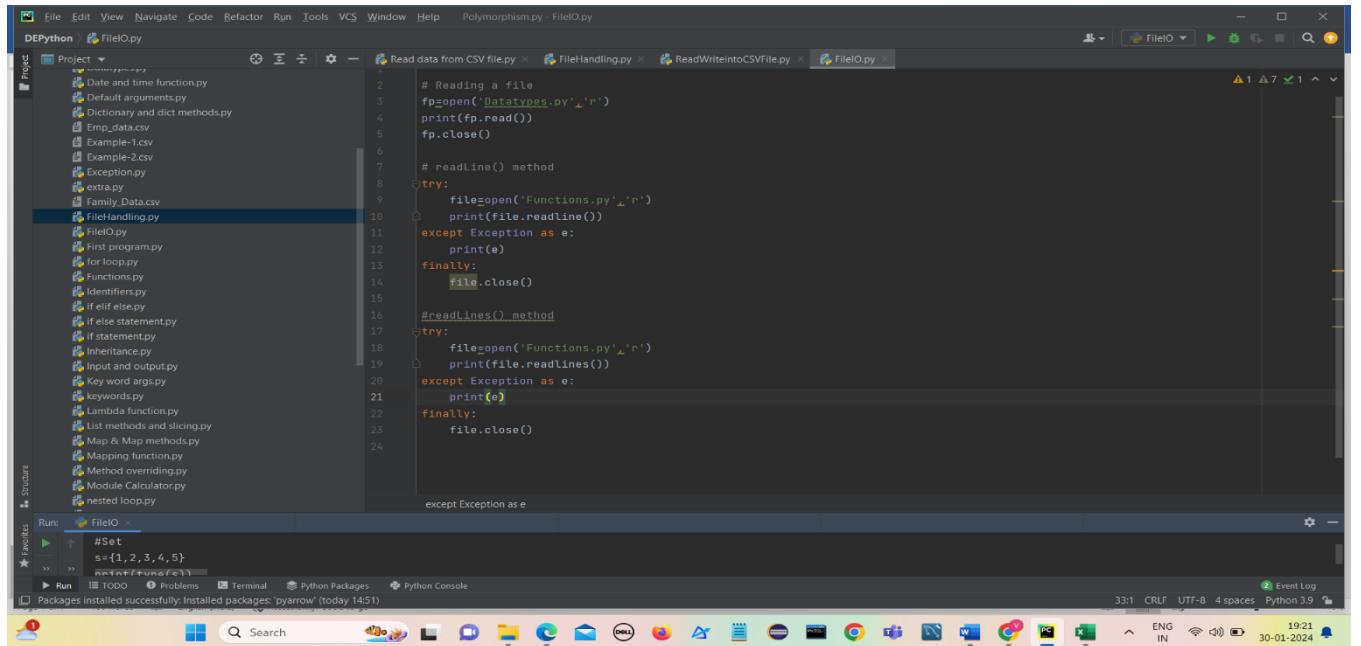
A file is read using read() function.

b) readLine():

readLine() method reads a single line from file.

c) readLines():

readLines() method reads the whole fist into list



The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and Polymorphism.py - FileIO.py. The left sidebar shows a project structure with various Python files like Date and time function.py, Default arguments.py, Dictionary and dict methods.py, Emp_data.csv, Example-1.csv, Example-2.csv, Exception.py, extra.py, Family_Data.csv, FileHandling.py, FileIO.py, First program.py, for loop.py, functions.py, Identifiers.py, if elif else.py, if else statement.py, If statement.py, Inheritance.py, Input and output.py, Key word args.py, keywords.py, Lambda function.py, List methods and slicing.py, Map & Map methods.py, Mapping function.py, Method overriding.py, Module Calculator.py, and nested loop.py. The main code editor window contains the following Python code:

```
# Reading a file
fp=open('datatype.py','r')
print(fp.read())
fp.close()

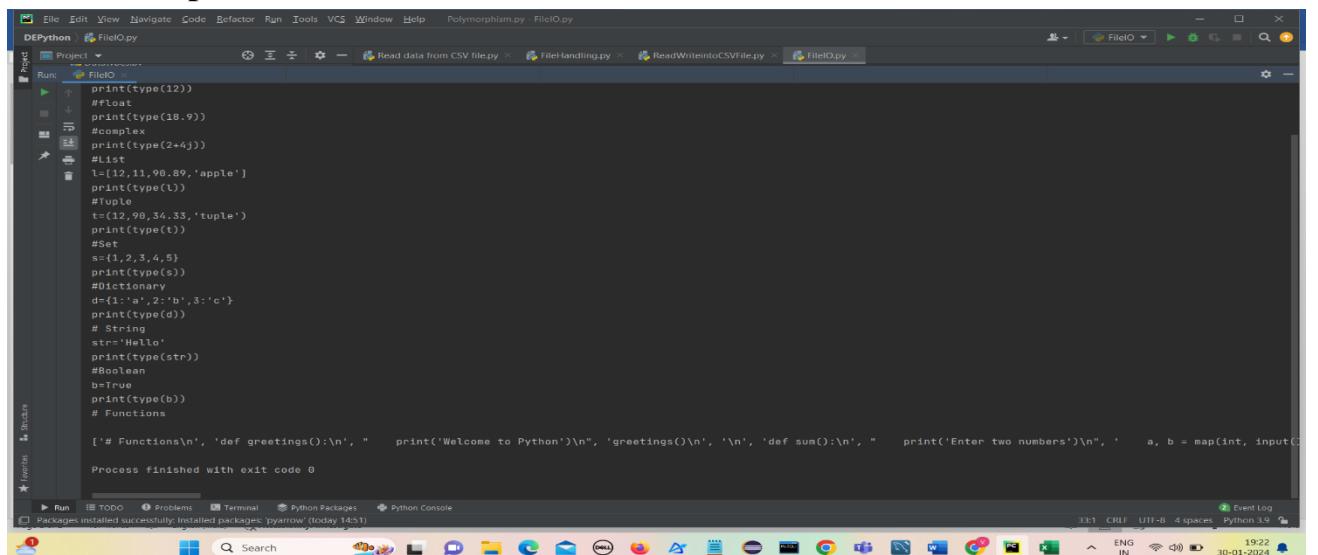
# readLine() method
try:
    file=open('Functions.py','r')
    print(file.readline())
except Exception as e:
    print(e)
finally:
    file.close()

#readlines() method
try:
    file=open('Functions.py','r')
    print(file.readlines())
except Exception as e:
    print(e)
finally:
    file.close()

except Exception as e
```

The bottom status bar shows the Python version as Python 3.9. The taskbar at the bottom includes icons for various applications like Microsoft Word, Excel, and Google Chrome.

Output:



The screenshot shows the PyCharm IDE interface with the same project structure and menu as the previous screenshot. The code editor now displays the output of the previous code execution:

```
Process finished with exit code 0
```

The bottom status bar shows the Python version as Python 3.9. The taskbar at the bottom includes icons for various applications like Microsoft Word, Excel, and Google Chrome.

d) write():

To use write method open the file in ‘w’ mode. It is used to write specified string to file.

e) writeLines():

It is used to write list of strings to file.

The screenshot shows the PyCharm IDE interface. The project navigation bar at the top lists various Python files. In the center, the code editor displays the `writelines.txt` file. The code defines two methods: `greetings()` and `writelines()`. The `writelines()` method opens a file named `xyz.txt` in write mode ('w') and writes a list of strings to it. The strings are: "See you soon!", "Over and out.", and "Come and join us". The run tab at the bottom shows the output of the script, which includes the greetings and the written lines. The status bar at the bottom right indicates the date and time as 30-01-2024 19:34.

```
def greetings():
    print('Welcome to Python')
    print('greetings()')
    print('def sum():')
    print('Enter two numbers')
    a, b = map(int, input().split())
    print(a + b)

def writelines():
    text = 'Hello, Good Morning'
    fp = open("xyz.txt", 'w')
    fp.write(text)
    print('Done Writing')
    fp.close()

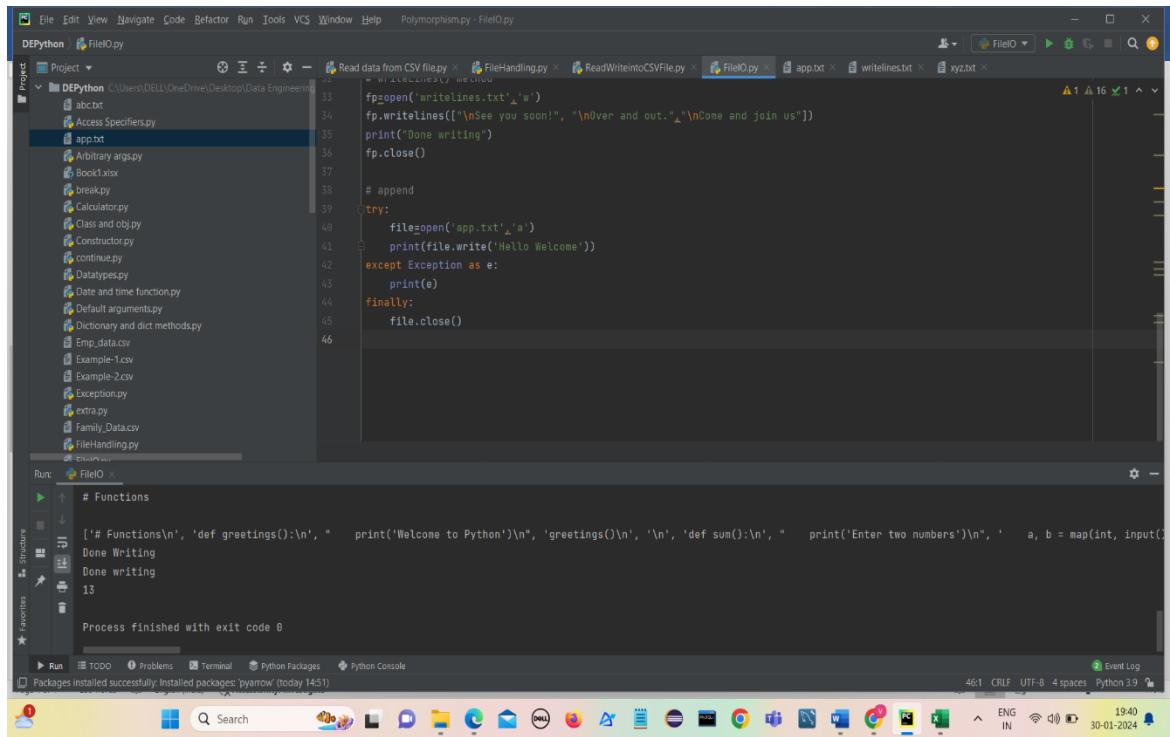
# writelines() method
fp = open('writelines.txt', 'w')
fp.writelines(["See you soon!", "\nOver and out.", "\nCome and join us"])
print("Done writing")
fp.close()
```

This screenshot is identical to the one above, showing the PyCharm IDE interface. It displays the same code in the `writelines.txt` file and the same output in the run tab. The output shows the three strings being printed to the console. The status bar at the bottom right shows the date and time as 30-01-2024 19:35.

f) append():

To append data, use the append mode in the `open()` function that allows writing the data to the existing file.

The access mode must be 'a'.



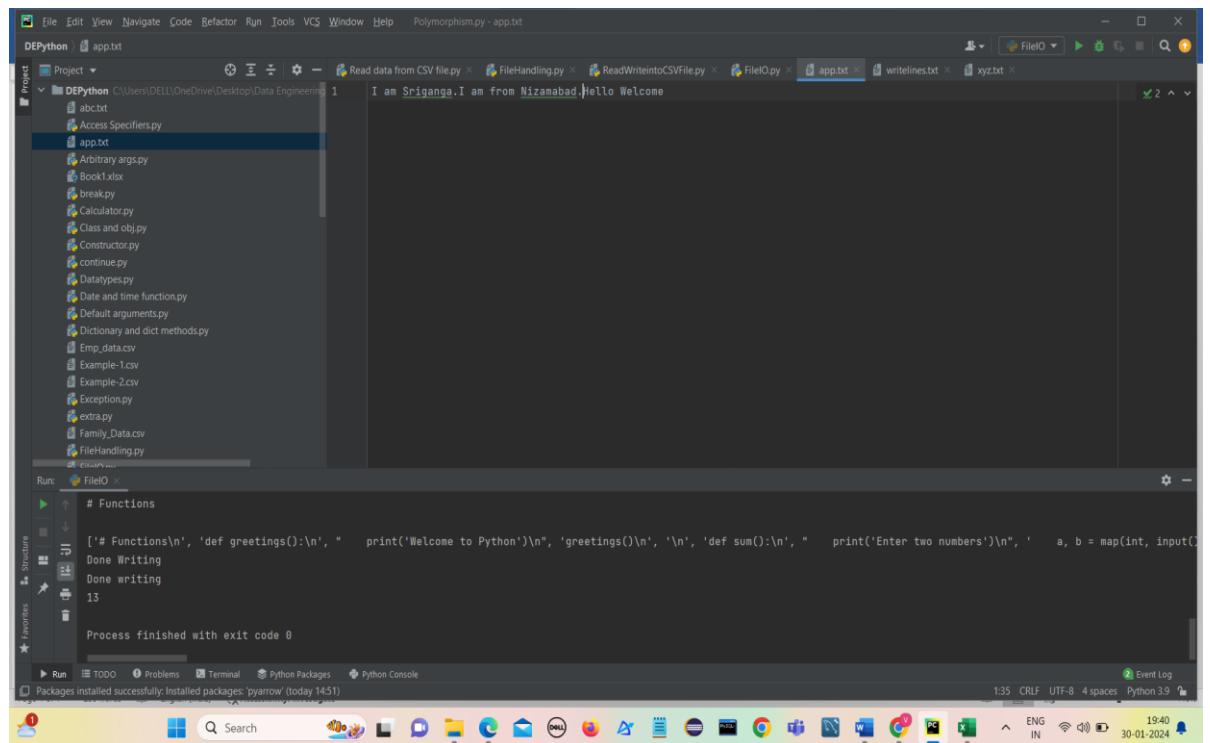
A screenshot of the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and Polymorphism.py - FileIO.py. The left sidebar shows a project structure for 'DEPython' with files like abc.txt, Access Specifiers.py, app.txt, Arbitrary args.py, Book.xlsx, break.py, Calculator.py, Class and obj.py, Constructor.py, continue.py, Datatypes.py, Date and time function.py, Default arguments.py, Dictionary and dict methods.py, Emp_data.csv, Example-1.csv, Example-2.csv, Exception.py, extra.py, Family_Data.csv, FileHandling.py, and xyz.txt. A 'Run' section at the bottom lists 'FileIO' and '# Functions'. The main code editor window contains Python code for writing to 'app.txt' in append mode ('a'). The run output shows the text 'Hello Welcome' was successfully appended to the file. The status bar at the bottom right shows the date as 30-01-2024 and the time as 19:40.

```
# WriteLines() method
fp = open('writelnlines.txt', 'w')
fp.writelines(["\nSee you soon!", "\nOver and out.", "\nCome and join us"])
print("Done writing")
fp.close()

# append
try:
    file = open('app.txt', 'a')
    print(file.write('Hello Welcome'))
except Exception as e:
    print(e)
finally:
    file.close()

# Functions
["# Functions\n", 'def greetings():\n', "    print("Welcome to Python")\n", 'greetings()\n', '\n', 'def sum():\n', "    print("Enter two numbers")\n", '    a, b = map(int, input())\n', '    return a + b\n', 'sum()\n', 'Done Writing\n', 'Done writing\n', '13\n']

Process finished with exit code 0
```



A second screenshot of the PyCharm IDE interface, identical to the first one but with a different run output. The run output now shows the text 'I am Sriganga.I am from Nizamabad.' was successfully appended to the file. The status bar at the bottom right shows the date as 30-01-2024 and the time as 19:40.

```
I am Sriganga.I am from Nizamabad.Hello Welcome

# Functions
["# Functions\n", 'def greetings():\n', "    print("Welcome to Python")\n", 'greetings()\n', '\n', 'def sum():\n', "    print("Enter two numbers")\n", '    a, b = map(int, input())\n', '    return a + b\n', 'sum()\n', 'Done Writing\n', 'Done writing\n', '13\n']

Process finished with exit code 0
```

g) seek():

It is used to change or move cursor to place where data must be read or written in the file.

h) tell():

It is used to return current position of file pointer from beginning of file.

The screenshot shows the PyCharm IDE interface with a project named 'DEPython' containing several files. The current file is 'FileO.py'. The code demonstrates the use of seek() and tell() methods:

```
try:
    file=open('app.txt','a')
    print(file.write('Hello Welcome'))
except Exception as e:
    print(e)
finally:
    file.close()

# seek() and tell()
fp=open('datatype.py','r')
fp.seek(67)
print(fp.tell())
print(fp.read())
fp.close()
```

The 'Run' tab is selected, showing the output of the code execution:

```
t(type(18.9))
#complex
print(type(2+4j))
#List
l=[12,11,99.89,'apple']
print(type(l))
#Tuple
t=(12,90,34.33,'tuple')
print(type(t))
#Set
s={1,2,3,4,5}
print(type(s))
```

The status bar at the bottom right indicates the date and time: 30-01-2024.

i) close()

close() in python is used to close the file
fp.close()

j) Copy contents of one file into another

The screenshot shows the PyCharm IDE interface with a project named 'DEPython' containing several files. The current file is 'FileO.py'. The code demonstrates how to copy the contents of one file into another:

```
fp.close()

#Copy the contents of one file into another
def read(f1,f2):
    try:
        f1=open(f1,'r')
        txt=f1.read()
        f1.close()
        f2=open(f2,'w')
        f2.write(txt)
        f2.close()
    except Exception as e:
        print(e)
finally:
    print("Contents copied successfully")
```

The 'Run' tab is selected, showing the output of the code execution:

```
Contents copied successfully
```

The status bar at the bottom right indicates the date and time: 30-01-2024.

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Behavior, Run, Tools, VCS, Window, Help, and Polymorphism.py - xyz.txt. The main window displays a code editor with Python code for demonstrating various data types. Below the code editor is a run console showing the output of the code execution. The bottom status bar shows the Python version as Python 3.9. The taskbar at the bottom of the screen includes icons for various applications like Search, Mail, and Browser.

```
1 #Datatypes
2 print("Datatypes")
3 #int
4 print(type(12))
5 #float
6 print(type(18.9))
7 #complex
8 print(type(2+4j))
9 #List
10 l=[12,11,98.89,'apple']
11 print(type(l))
12 #Tuple
13 t=(12,90,34.33,'tuple')
14 print(type(t))
15 #Set
16 s={1,2,3,4,5}
17 print(type(s))
18 #Dictionary
19 d={1:'a',2:'b',3:'c'}
20 print(type(d))
21 # String
```

Run: FileIO
print(type(str))
#Boolean
b=True
print(type(b))
Contents copied successfully
Process finished with exit code 0

2) Read data from CSV file into python list

- CSV file is a type of text file that stores tabular data for better readability, easier understanding, and faster processing.
- CSV stands for “Comma Separated Values.”
- It is the simplest form of storing data in tabular form as plain text.

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

1) Read CSV file using csv.reader

It is used to read CSV files using the `csv.reader` object from Python's `csv` module.

The screenshot shows the PyCharm IDE interface. The project navigation bar at the top lists several Python files. In the center, a code editor window displays the following Python script:

```
# Read CSV Files in python Using csv.reader
import csv
rows = []
with open("Example-1.csv", 'r') as file:
    csvreader = csv.reader(file)
    header = next(csvreader)
    for row in csvreader:
        rows.append(row)
print(header)
print(rows)
print('.....')
```

Below the code editor, the run tab shows the command: `C:/Users/DELL/AppData/Local/Programs/Python/Python39/python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/ReadWriteintoCSVFile.py"`. The output pane shows the execution results:

```
['ID', 'Name', 'Age', 'Salary']
[['1', 'Sona', '12', '12000'], ['2', 'Mona', '21', '43000'], ['3', 'Bannu', '13', '25000'], ['4', 'Chinnu', '23', '10000'], ['5', 'sunny', '14', '23000'], ['6', 'raju', '24', '78000'], ['7', 'ram', '15', '45000'], ['8', 'lucky', '25', '50000'], ['9', 'Appu', '16', '30000']]
```

At the bottom, the status bar indicates the date as 30-01-2024 and the time as 20:05.

2) Read CSV file using .readlines() function

`.readlines()` method is to fetch the header and rows using only `open()` and `with()` statements and without the `csv` library.

The screenshot shows the PyCharm IDE interface. The project navigation bar at the top lists several Python files. In the center, a code editor window displays the following Python script:

```
#_read_CSV_files_Using_readlines()
with open('Example-1.csv') as file:
    content = file.readlines()
header = content[:1]
rows = content[1:]
print(header)
print(rows)

print('.....')
```

Below the code editor, the run tab shows the command: `C:/Users/DELL/AppData/Local/Programs/Python/Python39/python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/ReadWriteintoCSVFile.py"`. The output pane shows the execution results:

```
['ID,Name,Age,Salary\n']
[['1,Sona,12,12000\n', '2,Mona,21,43000\n', '3,Bannu,13,25000\n', '4,Chinnu,23,10000\n', '5,sunny,14,23000\n', '6,raju,24,78000\n', '7,ram,15,45000\n', '8,lucky,25,50000\n', '9,Appu,16,30000\n']]
```

At the bottom, the status bar indicates the date as 30-01-2024 and the time as 20:09.

3) Read CSV file using Pandas

- Import pandas library
- Load CSV files to pandas using read_csv()
- Print the data

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and Polymorphism.py - ReadWriteintoCSVFile.py. The left sidebar has a Project tree with a DEPython folder containing several Python files like abc.txt, Access Specifiers.py, app.txt, etc. The main editor window contains the following code:

```
DEPython | ReadWriteintoCSVFile.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - ReadWriteintoCSVFile.py
Project DEPython C:\Users\DELL\OneDrive\Desktop\Data Engineering
Run: Read data from CSV file.py (1)
PEP 8: E265 block comment should start with '#'
43
44
45
46
47
48
49
50
51
52
53
54
import pandas as pd
#Load CSV files to pandas using read_csv()
data= pd.read_csv("Example-1.csv")
print(data)

# Extract the field names
print(data.columns)

# Extract rows
print(data.Age)
print(data.Salary)
```

The Run tab shows the output of the script:

```
C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/ReadWriteintoCSVFile.py"
   ID  Name  Age  Salary
0   1  Sona   12  12000
1   2  Mona   21  43000
2   3  Banu   13  25000
3   4  Chinnu  23  10000
4   5  sunny   14  23000
5   6  raju   24  78000
6   7  ran   15  45000
7   8  lucky   25  50000
8   9  Appy   17  60000
9   10 Lekha   38  21890
10  11 Vinnni  11  78654
11  12 pavan   19  34567
12  13 Sita   32  78901
13  14 latha  33  56455
```

The status bar at the bottom indicates the run time as 44:43, CRLF, UTF-8, 4 spaces, Python 3.9, and the date as 30-01-2024.

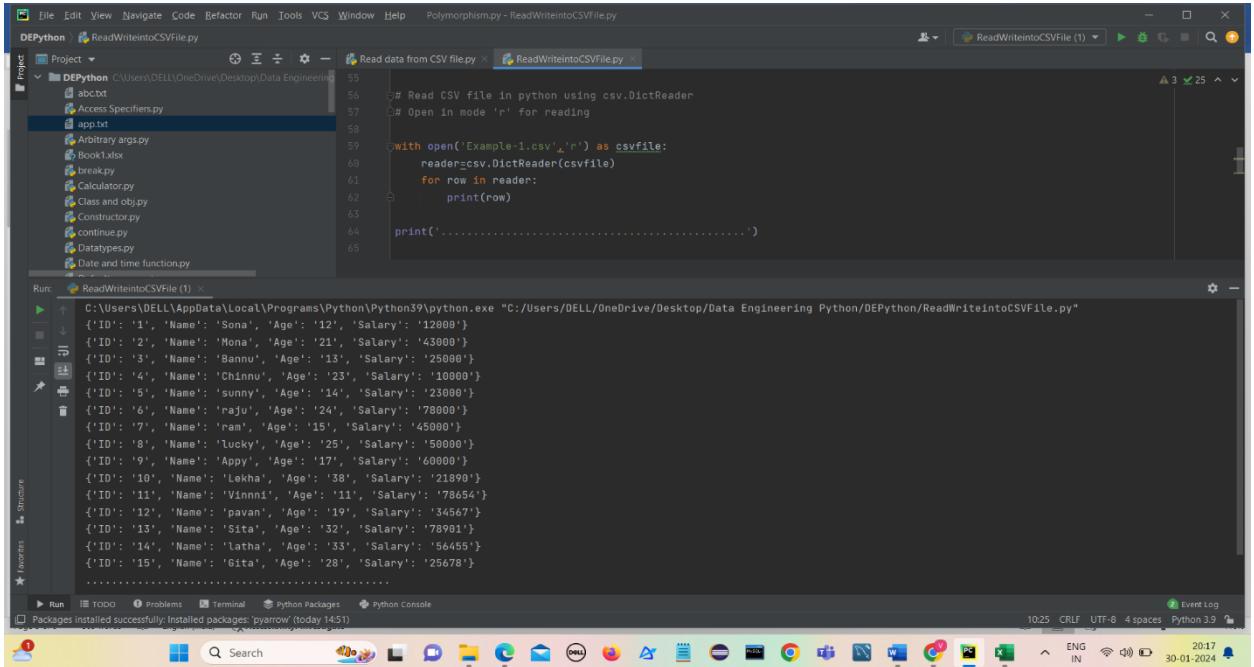
The screenshot shows the PyCharm IDE interface with a dark theme. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and Polymorphism.py - ReadWriteintoCSVFile.py. The left sidebar has a Project tree with a DEPython folder containing several Python files like abc.txt, Access Specifiers.py, app.txt, etc. The main editor window contains the following code:

```
DEPython | ReadWriteintoCSVFile.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - ReadWriteintoCSVFile.py
Project DEPython C:\Users\DELL\OneDrive\Desktop\Data Engineering
Run: ReadWriteintoCSVFile (1)
PEP 8: E265 block comment should start with '#'
13 14  latha  33  56455
14 15  Gita   28  25678
Index(['ID', 'Name', 'Age', 'Salary'], dtype='object')
0   12
1   21
2   13
3   23
4   14
5   24
6   15
7   25
8   17
9   38
10  11
11  19
12  32
13  33
14  28
Name: Age, dtype: int64
0   12000
1   43000
2   25000
3   10000
4   23000
5   78000
6   45000
7   50000
8   60000
```

The status bar at the bottom indicates the run time as 44:43, CRLF, UTF-8, 4 spaces, Python 3.9, and the date as 30-01-2024.

4) Read CSV file using csv.DictReader()

- 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.



```
# Read CSV file in python using csv.DictReader
# Open in mode 'r' for reading
with open('Example-1.csv','r') as csvfile:
    reader=csv.DictReader(csvfile)
    for row in reader:
        print(row)

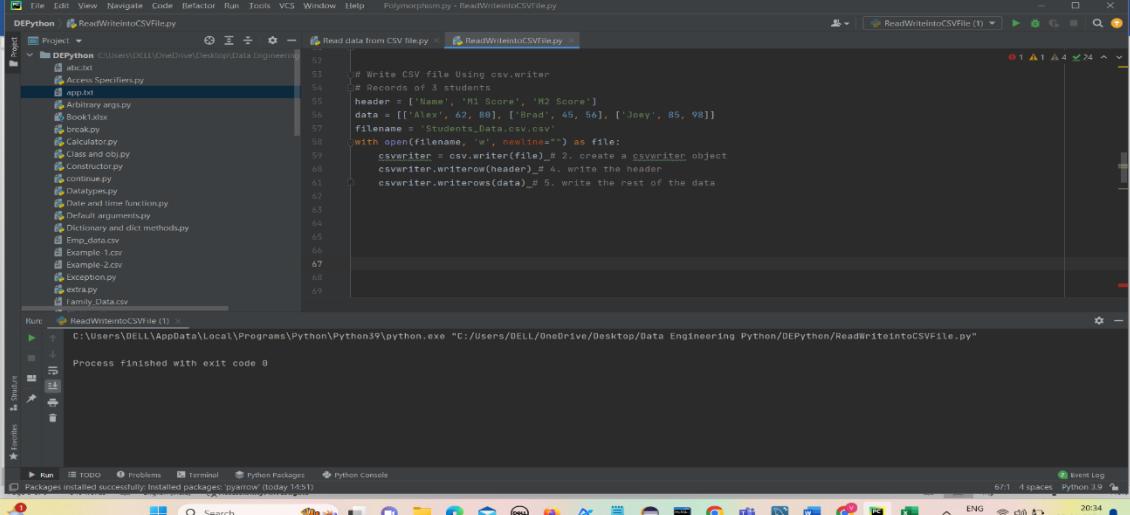
print('.....')
```

```
C:/Users/DELL/AppData/Local/Programs/Python/Python39/python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/ReadWriteintoCSVFile.py"
{'ID': '1', 'Name': 'Sona', 'Age': '12', 'Salary': '12000'}
{'ID': '2', 'Name': 'Mona', 'Age': '21', 'Salary': '43000'}
{'ID': '3', 'Name': 'Bannu', 'Age': '13', 'Salary': '25000'}
{'ID': '4', 'Name': 'Chinmu', 'Age': '23', 'Salary': '10000'}
{'ID': '5', 'Name': 'sunny', 'Age': '14', 'Salary': '23000'}
{'ID': '6', 'Name': 'raju', 'Age': '24', 'Salary': '78000'}
{'ID': '7', 'Name': 'ram', 'Age': '15', 'Salary': '45000'}
{'ID': '8', 'Name': 'lucky', 'Age': '25', 'Salary': '50000'}
{'ID': '9', 'Name': 'appy', 'Age': '17', 'Salary': '60000'}
{'ID': '10', 'Name': 'Lekha', 'Age': '38', 'Salary': '21890'}
{'ID': '11', 'Name': 'Vinni', 'Age': '11', 'Salary': '78654'}
{'ID': '12', 'Name': 'pavan', 'Age': '19', 'Salary': '34567'}
{'ID': '13', 'Name': 'Sita', 'Age': '32', 'Salary': '78901'}
{'ID': '14', 'Name': 'latha', 'Age': '33', 'Salary': '56455'}
{'ID': '15', 'Name': 'Gita', 'Age': '28', 'Salary': '25678'}
.....
```

List of 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

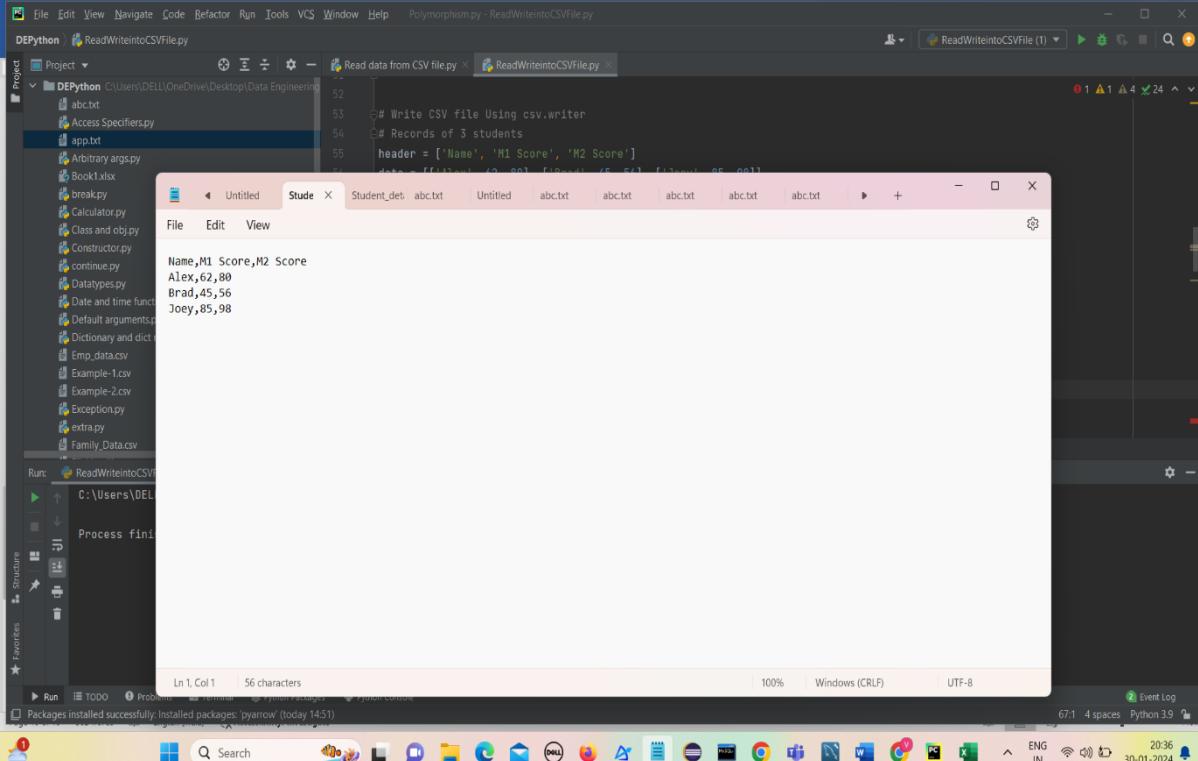
1) Write CSV file using csv.writer



```
DEPython C:\Users\DELL\OneDrive\Desktop\Data Engineering
Project ReadWriteintoCSVfile.py
  Read data from CSV file.py  ReadWriteintoCSVfile.py
  abc.txt
  Access Specifiers.py
  app.py
  Arbitrary args.py
  Book1.xlsx
  break.py
  Calculator.py
  Class and obj.py
  Constructor.py
  continue.py
  Datatypes.py
  Date and time function.py
  Default arguments.py
  Dictionary and dict methods.py
  Emp_data.csv
  Example-1.csv
  Example-2.csv
  Exception.py
  extra.py
  Family_Data.csv
Run: ReadWriteintoCSVfile (1) x
C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering/DEPython/ReadWriteintoCSVfile.py"
Process finished with exit code 0
  Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
  67/1 4 spaces Python 3.9 Event Log
```

The screenshot shows the PyCharm IDE interface. The project navigation bar at the top lists files like 'abc.txt', 'Access Specifiers.py', etc. Below it, the code editor displays a Python script named 'ReadWriteintoCSVfile.py'. The script uses the 'csv.writer' module to write data to a CSV file. The terminal below the code shows the command 'python.exe' was run and completed successfully with an exit code of 0. The status bar at the bottom indicates the Python version is 3.9.

The `csv.writer()` function returns a writer object that converts the input data into a delimited string.

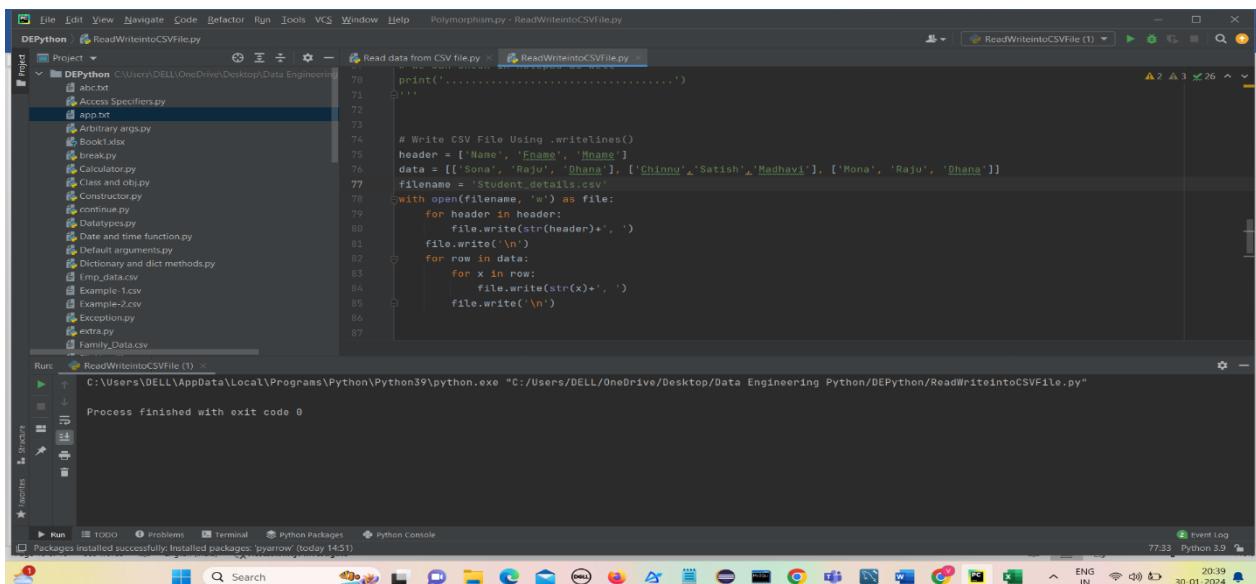


```
DEPython C:\Users\DELL\OneDrive\Desktop\Data Engineering
Project ReadWriteintoCSVfile.py
  Read data from CSV file.py  ReadWriteintoCSVfile.py
  abc.txt
  Access Specifiers.py
  app.txt
  Arbitrary args.py
  Book1.xlsx
  break.py
  Calculator.py
  Class and obj.py
  Constructor.py
  continue.py
  Datatypes.py
  Date and time function.py
  Default arguments.py
  Dictionary and dict methods.py
  Emp_data.csv
  Example-1.csv
  Example-2.csv
  Exception.py
  extra.py
  Family_Data.csv
Run: ReadWriteintoCSVfile
C:\Users\DELL\OneDrive\Desktop\Data Engineering\DEPython\ReadWriteintoCSVfile.py
Process finished with exit code 0
  Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
  67/1 4 spaces Python 3.9 Event Log
```

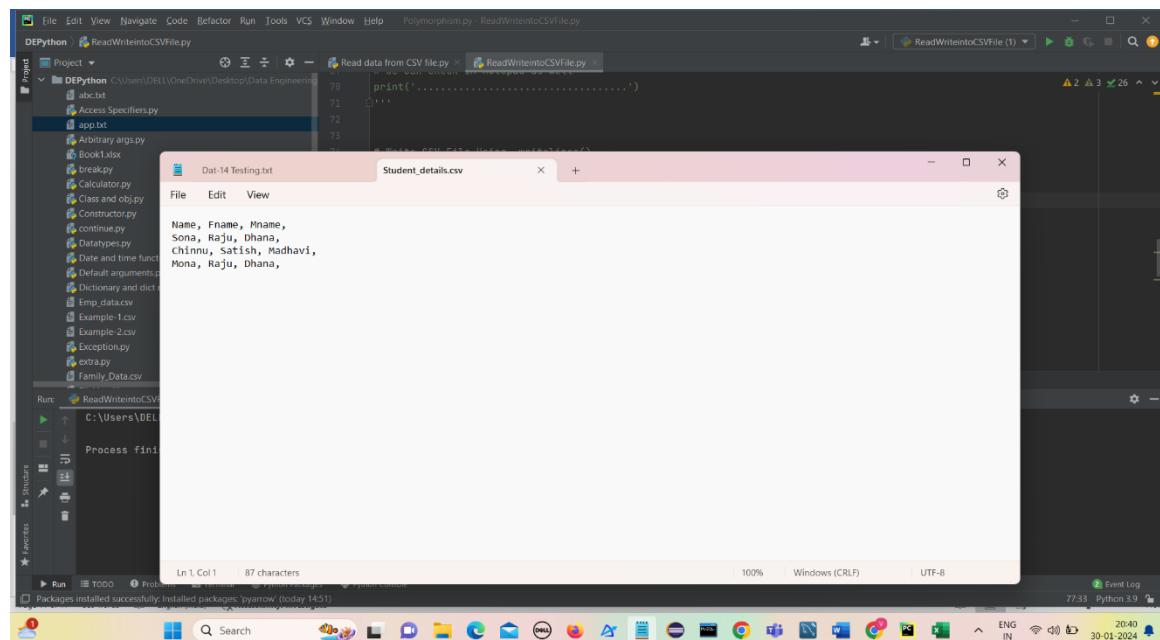
This screenshot shows the PyCharm IDE again, but this time the focus is on the 'ReadWriteintoCSVfile.py' script. The code has been modified to output data to a file named 'abc.txt'. The terminal shows the command 'python.exe' was run and completed successfully. The status bar indicates the Python version is 3.9. A separate window titled 'abc.txt' is open, displaying the contents of the CSV file: 'Name,M1 Score,M2 Score' followed by three rows of data: 'Alex,62,88', 'Brad,45,56', and 'Joey,85,98'.

2) Write CSV file using .writelines() function

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



```
DEPython | ReadWriteintoCSVfile.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - ReadWriteintoCSVfile.py
Project DEPython C:\Users\DELL\OneDrive\Desktop\Data Engineering
  abc.txt
  Access Specifiers.py
  app.txt
  Arbitrary args.py
  Book1.xlsx
  break.py
  Calculator.py
  Class and obj.py
  Constructor.py
  continue.py
  Datatypes.py
  Date and time function.py
  Default arguments.py
  Dictionary and dict methods.py
  Emp_data.csv
  Example-1.csv
  Example-2.csv
  Exception.py
  extra.py
  Family_Data.csv
ReadWriteintoCSVfile (1) ReadWriteintoCSVfile.py
print('.....')
.....
# Write CSV File Using .writelines()
header = ['Name', 'Fname', 'Mname']
data = [['Sona', 'Raju', 'Dhana'], ['Chinnu', 'Satish', 'Madhavi'], ['Mona', 'Raju', 'Dhana']]
filename = 'Student_details.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')
Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console
Packages installed successfully. Installed packages: 'pyarrow' (today 14:51)
Event Log 77:33 Python 3.9
Windows Taskbar: Run, Search, Start button, Icons, Network, Mail, Calendar, File Explorer, Task View, Taskbar settings, Language: ENG IN, Date: 30-01-2024, Time: 20:39
```



```
DEPython | ReadWriteintoCSVfile.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - ReadWriteintoCSVfile.py
Project DEPython C:\Users\DELL\OneDrive\Desktop\Data Engineering
  abc.txt
  Access Specifiers.py
  app.txt
  Arbitrary args.py
  Book1.xlsx
  break.py
  Calculator.py
  Class and obj.py
  Constructor.py
  continue.py
  Datatypes.py
  Date and time function.py
  Default arguments.py
  Dictionary and dict methods.py
  Emp_data.csv
  Example-1.csv
  Example-2.csv
  Exception.py
  extra.py
  Family_Data.csv
ReadWriteintoCSVfile (1) ReadWriteintoCSVfile.py
print('.....')
.....
# Write CSV File Using .writelines()
header = ['Name', 'Fname', 'Mname']
data = [['Sona', 'Raju', 'Dhana'], ['Chinnu', 'Satish', 'Madhavi'], ['Mona', 'Raju', 'Dhana']]
filename = 'Student_details.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')
Dat-14 Testing.txt Student_details.csv
File Edit View
Name, Fname, Mname,
Sona, Raju, Dhana,
Chinnu, Satish, Madhavi,
Mona, Raju, Dhana,
Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console
Packages installed successfully. Installed packages: 'pyarrow' (today 14:51)
Event Log 77:33 Python 3.9
Windows Taskbar: Run, Search, Start button, Icons, Network, Mail, Calendar, File Explorer, Task View, Taskbar settings, Language: ENG IN, Date: 30-01-2024, Time: 20:40
```

3) Write CSV file using Pandas

Create a pandas dataframe using pd.DataFrame

Syntax: pd.DataFrame(data, columns)

The data parameter takes the records/observations, and the columns parameter takes the columns/field names.

Write to a CSV file using to_sv0

Syntax: DataFrame.to_csv(filename, sep=',', index=False)

A screenshot of the PyCharm IDE. The project navigation bar shows several files like abc.txt, AccessSpecifiers.py, arp.txt, etc. The main editor window contains Python code for reading data from a CSV file and creating a pandas DataFrame. Below the code, the run tab shows the command: C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/ReadWriteintoCSVFile.py". The status bar at the bottom right indicates Python 3.9, ENG IN, 20:45, and 30-01-2024.

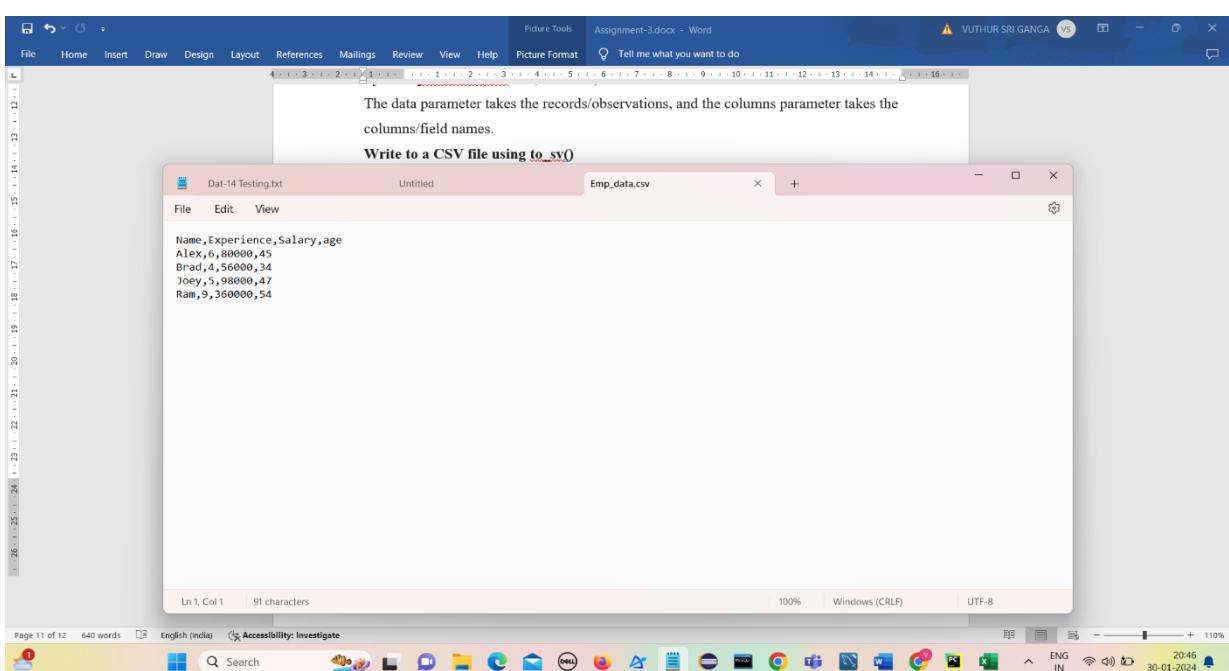
```
File Edit View Navigate Code Selector Run Tools Help Polymorphism.py ReadWriteintoCSVFile.py
DEPython C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/ReadWriteintoCSVFile.py
Project
  DEPython C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/
    abc.txt
    AccessSpecifiers.py
    arp.txt
    # Create a pandas dataframe using pd.DataFrame
    header = ['Name', 'Experience', 'Salary', 'Age']
    data = [['Alex', 6, 80000, 45], ['Brad', 4, 56000, 34], ['Joey', 5, 98000, 47], ['Ram', 9, 360000, 54]]
    data = pd.DataFrame(data, columns=header)
    # Write to a CSV file using to_csv()
    data.to_csv('Emp_data.csv', index=False)

Run ReadWriteintoCSVFile (1) - C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/ReadWriteintoCSVFile.py"
Process finished with exit code 0

Run Problems Terminal Python Packages Python Console
Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
Event Log 105.1 Python 3.9
ENG IN 20:45 30-01-2024
```

**separator is ',' by default.

index=False to remove the index numbers.



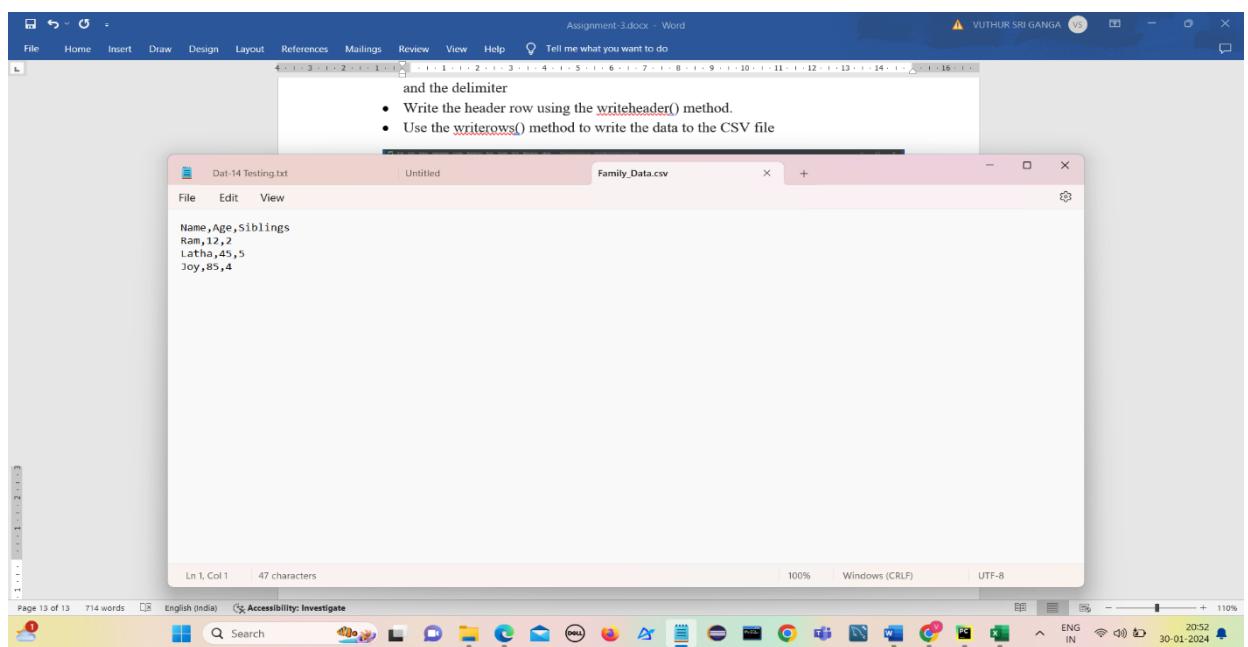
4) Write CSV file using csv.DictWriter

- Using the `.open()` function, create a new file object with the mode as ‘w’ for writing
 - Type in the data you want to write to the CSV file as a list of dictionaries
 - Create a `csv.DictWriter` object specifying the file object, the `fieldname` parameters, and the delimiter
 - Write the header row using the `writeheader()` method.
 - Use the `writerows()` method to write the data to the CSV file

The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Project Bar:** DEPython, ReadWriteintoCSVfile.py.
- Code Editor:** The current file is "ReadWriteintoCSVfile.py". The code demonstrates reading from and writing to a CSV file named "Family_Data.csv".

```
print(data)
print('.....')
...
# Write CSV File Using csv.DictWriter
# Using the .open() function, create a new file object with the mode as 'w' for writing
with open('Family_Data.csv', 'w', newline='') as csvfile:
    data = [{Name: 'Ran', Age: 12, Siblings: 2},
            {Name: 'Latoh', Age: 45, Siblings: 5},
            {Name: 'Joy', Age: 85, Siblings: 4}]
    fieldnames = [Name, Age, Siblings]
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()
    writer.writerows(data)
```
- Run Bar:** Shows "ReadWriteintoCSVfile (1)" and the command "C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/ReadWriteintoCSVFile.py"
- Output Bar:** "Process finished with exit code 0"
- Bottom Status Bar:** Packages installed successfully: Installed packages: pyarrow (today 14:51), 121:1 CRLF, UTF-8, 4 spaces, Python 3.9, ENG, IN, 30-01-2024.
- System Tray:** Shows icons for battery, signal, volume, and date/time.



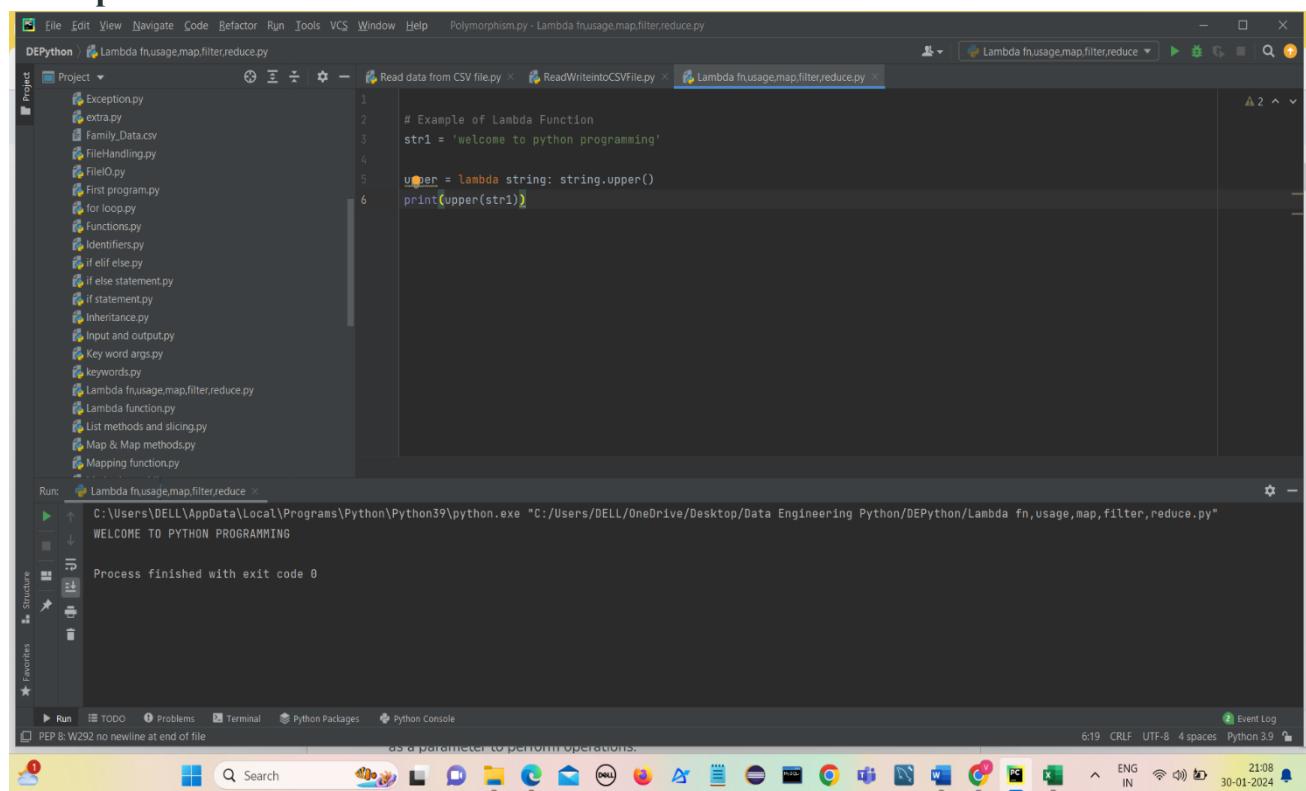
4) Lambda Functions

Python Lambda Functions are anonymous functions means that the function is without a name.

Syntax: lambda arguments : expression

This function can have any number of arguments but only one expression, which is evaluated and returned.

Example:



```
# Example of Lambda Function
str1 = 'welcome to python programming'

upper = lambda string: string.upper()
print(upper(str1))
```

Run: C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/Lambda fn.usage.map.filter.reduce.py"
WELCOME TO PYTHON PROGRAMMING
Process finished with exit code 0

In the example, we defined a lambda function(**upper**) to convert a string to its upper case using upper().

This code defines a lambda function named **upper** that takes a string as its argument and converts it to uppercase using the **upper()** method

5) Usage of Lambda Functions:

a) Condition Checking Using Python lambda function

‘format_numeric’ calls the lambda function, and the num is passed as a parameter to perform operations.

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help. The title bar says "Polymorphism.py - Lambda fn.usage.map.filter.reduce.py". The left sidebar has a "Project" view with many Python files listed. The main code editor window contains the following code:

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

# Usage of Lambda Function
#1) Condition Checking
format_numeric = lambda num: f'{num:e}' if isinstance(num, int) else f'{num:.2f}'

print("Int formatting:", format_numeric(1000000))
print("float formatting:", format_numeric(999999.789541235))
```

The "Run" tab at the bottom shows the output of the script:

```
C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/Lambda fn_usage.map.filter.reduce.py"
WELCOME TO PYTHON PROGRAMMING
Int formatting: 1.000000e+06
float formatting: 999,999.79
Process finished with exit code 0
```

The status bar at the bottom right shows the date and time: 30-01-2024, 21:16.

b) Python Lambda Function with List Comprehension

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help. The title bar says "Polymorphism.py - Lambda fn.usage.map.filter.reduce.py". The left sidebar has a "Project" view with many Python files listed. The main code editor window contains the following code:

```
print("Int formatting:", format_numeric(1000000))
print("float formatting:", format_numeric(999999.789541235))

#2) 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())
```

The "Run" tab at the bottom shows the output of the script:

```
C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/Lambda fn_usage.map.filter.reduce.py"
10
20
30
40
Process finished with exit code 0
```

The status bar at the bottom right shows the date and time: 30-01-2024, 21:20.

Lambda function is used in list comprehension.

c) Python Lambda Function with if-else

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - Lambda fn,usage,map,filter,reduce.py
DEPython | Lambda fn,usage,map,filter,reduce.py
Project | Read data from CSV file.py | ReadWriteintoCSVfile.py | Lambda fn,usage,map,filter,reduce.py
Run | Lambda fn,usage,map,filter,reduce.py
C:/Users/DELL/AppData/Local/Programs/Python/Python39/python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/Lambda fn,usage,map,filter,reduce.py"
89
Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console
Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
Event Log 27:18 CRLF UTF-8 4 spaces Python 3.9
2024-01-30 22:03
```

```
is_even_List = [lambda arg=x: arg * 10 for x in range(1, 5)]
for item in is_even_List:
    print(item())

#3) Python Lambda Function with if-else

Max = lambda a, b,: a if(a > b) else b
print(Max(12,89))
```

Max lambda function is used to find the maximum of two integers.

d) Python Lambda with Multiple Statements to find second maximum number

Lambda functions do not allow multiple statements, however, we can create two lambda functions and then call the other lambda function as a parameter to the first function to find second maximum number.

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - Lambda fn,usage,map,filter,reduce.py
DEPython | Lambda fn,usage,map,filter,reduce.py
Project | Read data from CSV file.py | ReadWriteintoCSVfile.py | Lambda fn,usage,map,filter,reduce.py
Run | Lambda fn,usage,map,filter,reduce.py
C:/Users/DELL/AppData/Local/Programs/Python/Python39/python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/Lambda fn,usage,map,filter,reduce.py"
[3, 16, 9]
Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console
Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
Event Log 3436 CRLF UTF-8 4 spaces Python 3.9
2024-01-30 22:07
```

```
# Python Lambda with Multiple Statements to find second maximum number

List = [[2, 3, 4], [1, 4, 16, 64], [3, 6, 9, 12]]
sortList = lambda x: (sorted(i) for i in x)
secondLargest = lambda x, f: [y[Len(y) - 2] for y in f(x)]
res = secondLargest(List, sortList)
print(res)
```

6)Filter data in python lists using map, filter and reduce

a) Using lambda() Function with filter()

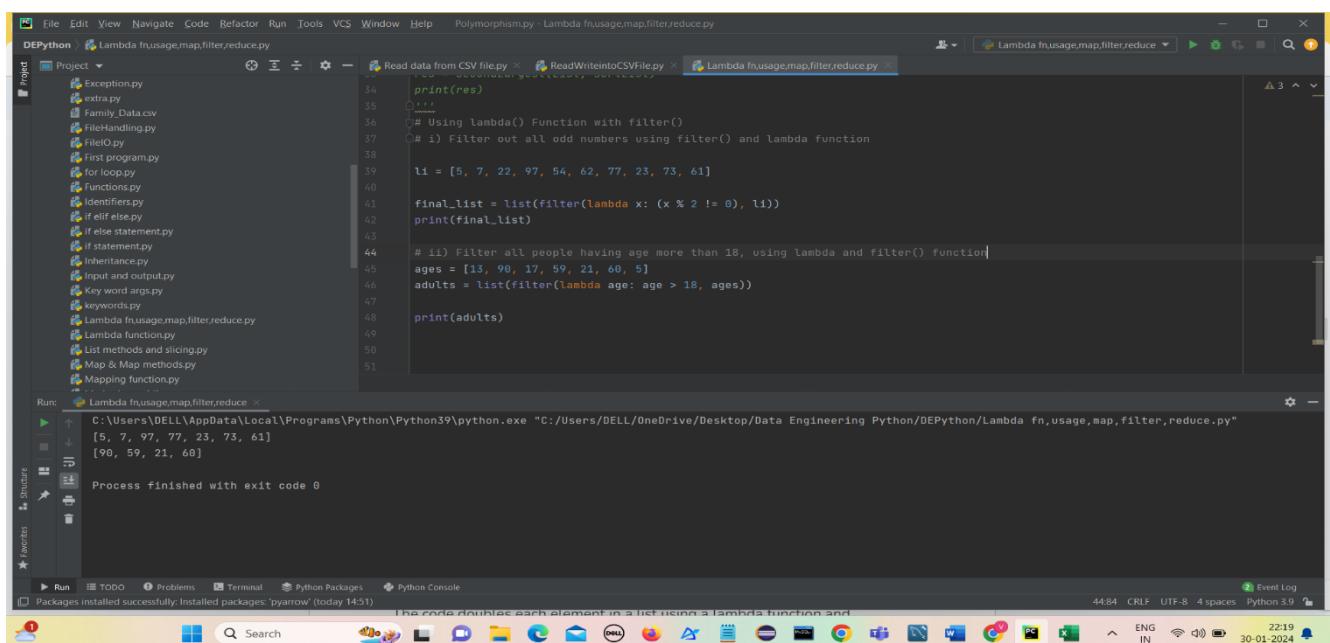
The filter() function in Python takes in a function and a list as arguments to filter out all the elements of a sequence for which result is true.

i) Filter out all odd numbers using filter() and lambda function

It filters a list and gives a list of odd numbers

ii) Filter all people having age more than 18, using lambda and filter() function.

It filters a list of ages and extracts the ages of adults (ages greater than 18) using a lambda function and the 'filter' function. It then prints the list of adult ages.



The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and the current file Polymorphism.py - Lambda fn.usage,map,filter,reduce.py. The left sidebar shows a project structure with files like Exception.py, Family_Data.csv, FileHandling.py, FileIO.py, First program.py, for loop.py, Functions.py, Identifiers.py, if elif else.py, if else statement.py, if statement.py, Inheritance.py, Input and output.py, key word args.py, keywords.py, Lambda fn.usage.map.filter.reduce.py, Lambda function.py, List methods and slicing.py, Map & Map methods.py, and Mapping function.py. The main code editor window contains the following Python code:

```
print(res)
# i) Using lambda() Function with filter()
# # i) Filter out all odd numbers using filter() and lambda function
li = [5, 7, 22, 97, 54, 62, 77, 23, 73, 61]
final_list = list(filter(lambda x: (x % 2 != 0), li))
print(final_list)

# ii) Filter all people having age more than 18, using lambda and filter() function
ages = [13, 99, 17, 59, 21, 66, 5]
adults = list(filter(lambda age: age > 18, ages))

print(adults)
```

The Run tab at the bottom shows the command C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/Lambda fn.usage,map,filter,reduce.py" and the output [5, 7, 23, 73, 61]. The status bar at the bottom right indicates the date and time as 30-01-2024.

b) Using lambda() Function with map()

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.

i) Multiply all elements of a list by 2 using lambda and map() function

The code doubles each element in a list using a lambda function and the 'map' function.

It then prints the new list with the doubled elements.

ii) Transform all elements of a list to upper case using lambda and map() function

The code converts a list of animal names to uppercase using a lambda function and the ‘map’ function. It then prints the list with the animal names in uppercase.

A screenshot of the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and Polymorphism.py - Lambda fn.usage.map.filter.reduce.py. The left sidebar shows a Project tree with various Python files like Exception.py, extra.py, Family_Data.csv, etc. The main code editor window contains the following Python script:

```
47     print(adults)
48
49     """
50     # Using lambda() Function with map()
51     # i) Multiply all elements of a list by 2 using lambda and map() function
52
53     li = [5, 7, 22, 97, 54, 62, 77, 23, 73, 61]
54     final_list = list(map(lambda x: x * 2, li))
55     print(final_list)
56
57     # ii) Transform all elements of a list to upper case using lambda and map() function
58     animals = ['dog', 'cat', 'parrot', 'rabbit']
59     uppered_animals = list(map(lambda animal: animal.upper(), animals))
60     print(uppered_animals)
61
62
63
64
```

The Run tab at the bottom shows the command: C:/Users/DELL/appData/Local/Programs/Python/Python39/python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/Lambda fn.usage.map.filter.reduce.py". The output window shows the results of the execution.

c) Using lambda() Function with reduce()

- 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.
- This performs a repetitive operation over the pairs of the iterable.
- The reduce() function belongs to the **functools** module.

i) A sum of all elements in a list using lambda and reduce() function

It imports ‘reduce’, defines a list, applies a lambda function that adds two elements at a time, and prints the sum of all elements in the list.

ii) Find the maximum element in a list using lambda and reduce() function

The code uses the ‘**functools**’ module to find the maximum element in a list (‘lis’) by employing the ‘reduce’ function and a lambda function.

```
# Using lambda() function with reduce
# i) A sum of all elements in a list using lambda and reduce() function
from functools import reduce
li = [5, 8, 10, 20, 50, 100]
sum = reduce(lambda x, y: x + y, li)
print(sum)

# ii) Find the maximum element in a list using lambda and reduce() function
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))
```

Run: C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/Lambda fn.usage.map.filter.reduce.py"
193
The maximum element of the list is : 6
Process finished with exit code 0

3. Processing python lists

Processing Python lists involves various operations such as creating lists, accessing elements, modifying elements, iterating through lists, and performing various transformations and manipulations.

- a) Creating list
- b) Accessing elements of list
- c) Modifying elements
- d) Count of element
- e) Index of element
- f) Add item in the last
- g) Remove element
- h) Delete last element
- i) Insert element
- j) Reverse list
- k) Sort list
- l) Extend list
- m) Iterate over loop
- n) List comprehension
- o) Sorting list using sorted()
- p) Slicing

The screenshot shows the PyCharm IDE interface with the following details:

- Project:** DEPython
- File:** List methods and slicing.py
- Code Content:**

```
10     print("List")
11     # Creating a list
12     l1=[10,20,30,40,50,10.5,10,20]
13     print(l1)
14
15     # Accessing elements of list
16     print(l1[6])
17     print(l1[1])
18
19     # Modifying elements
20     l1[6]=278
21     print(l1)
22     l2=[100,200,300]
23
24     # print all the functions of list
25     print(dir(l1))
26
27     # count of an element
28     print(l1.count(10))
29
30     # index of an element
31     print(l1.index(40))
32
33     #function of list:
34     # add item in last
35     l1.append(45)
36     print(l1)
37
```

- Toolbars and Status Bar:** The status bar at the bottom shows "Packages installed successfully: Installed packages: pyarrow (today 14:51)", "28:22 Python 3.9", and the date "30-01-2024".

The screenshot shows the PyCharm IDE interface with the following details:

- Project:** DEPython
- File:** List methods and slicing.py
- Code Content:**

```
37     print(l1)
38
39     # remove an element
40     l1.remove(10.5)
41     print(l1)
42
43     #delete last element
44     l1.pop()
45     print(l1)
46
47     # Insert element
48     l1.insert(2,99)
49     print(l1)
50
51     #reverse the list
52     l1.reverse()
53     print(l1)
54
55     #sort the list
56     l1.sort(reverse=False)
57     print(l1)
58
59     #extend
60     l1.extend(l2)
61     print(l1)
62
63     # Iterate over loop
64     for i in l1:
65         print(i)
```

- Toolbars and Status Bar:** The status bar at the bottom shows "Packages installed successfully: Installed packages: pyarrow (today 14:51)", "28:22 Python 3.9", and the date "30-01-2024".

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - List methods and slicing.py
DEPython > List methods and slicing.py
Project Run Favorites
Run TODO Problems Terminal Python Packages Python Console
Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
28:22 Python 3.9 Event Log
23:01 30-01-2024
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - List methods and slicing.py
DEPython > List methods and slicing.py
Project Run Favorites
Run TODO Problems Terminal Python Packages Python Console
Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
28:22 Python 3.9 Event Log
23:01 30-01-2024
```

```
for loop.py
Functions.py
Identifiers.py
if elif else.py
if else statement.py
if statement.py
Inheritance.py
Input and output.py
Key word args.py
keywords.py
Lambda frusage.map.filter.reduce.py
Lambda function.py
List methods and slicing.py
Map & Map methods.py
Mapping function.py
Method overriding.py
Module Calculator.py
nested loop.py
Number function.py
Operators.py
pass.py
Polymorphism.py
Read data from CSV file.py
ReadWriteintoCSVfile.py
Set & Set methods.py
simple pandas pgm.py
special parameters.py
String Function.py
Student_details.csv
Students_Data.csv
Students_Data.csv.csv
tuple.py

62
# Iterate over loop
63     for i in li:
64         print(i,end=' ')
65
66 # List Comprehension
67 print("\n")
68 doubled = [x * 2 for x in li]
69 print(doubled)
70
71 # Sorting a list
72 sorted_list = sorted(li)
73 print(sorted_list)
74
75 #Slicing
76 print("\nSlicing in list")
77 l2=[11,34.5,"Welcome",12,1,4,7,9,111,2222,989899]
78 print(l2)
79 print(l2[2:7])
80 print(l2[::-1])# whole list
81 print(l2[::-1])#reverse
82 print(l2[-5:-1])
83
```

Output:

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - List methods and slicing.py
DEPython > List methods and slicing.py
Project Run Favorites
Run TODO Problems Terminal Python Packages Python Console
Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
28:22 Python 3.9 Event Log
23:01 30-01-2024
File Edit View Navigate Code Refactor Run Tools VCS Window Help Polymorphism.py - List methods and slicing.py
DEPython > List methods and slicing.py
Project Run Favorites
Run TODO Problems Terminal Python Packages Python Console
Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
28:22 Python 3.9 Event Log
23:01 30-01-2024
C:\Users\DELL\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/DELL/OneDrive/Desktop/Data Engineering Python/DEPython/List methods and slicing.py"
List
[10, 20, 30, 40, 50, 10.5, 10, 20]
10
20
[10, 20, 30, 40, 50, 10.5, 278, 20]
['__add__', '__class__', '__class_getitem__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', 1
3
[10, 20, 30, 40, 50, 10.5, 278, 20, 45]
[10, 20, 30, 40, 50, 278, 20, 45]
[10, 20, 30, 40, 50, 278, 20]
[10, 20, 30, 40, 50, 278, 20]
[10, 20, 28, 30, 40, 50, 99, 278]
[10, 20, 28, 30, 40, 50, 99, 278]
[10, 20, 28, 30, 40, 50, 99, 278, 100, 200, 300]
10 20 20 30 40 50 99 278 100 200 300

[20, 40, 40, 60, 80, 100, 120, 140, 160, 180, 200, 220, 240, 260, 280, 300]

Slicing in list
[11, 34.5, 'Welcome', 12, 1, 4, 7, 9, 111, 2222, 989899]
['Welcome', 12, 1, 4, 7]
[11, 34.5, 'Welcome', 12, 1, 4, 7, 9, 111, 2222, 989899]
[989899, 2222, 111, 9, 7, 4, 1, 12, 'Welcome', 34.5, 11]
[7, 9, 111, 2222]

Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console
Packages installed successfully: Installed packages: 'pyarrow' (today 14:51)
30:1 Python 3.9 Event Log
23:02 30-01-2024
```