**PYTHON ASSIGNMENT**

**File IO using CSV files**

Read data from CSV file

def read\_csv\_file(file\_path):

try:

with open(file\_path, 'r') as csvfile:

# Creating a CSV reader object

csv\_reader = csv.reader(csvfile,delimiter = '\t')

# Reading and printing each row in the CSV file

for row in csv\_reader:

print(row)

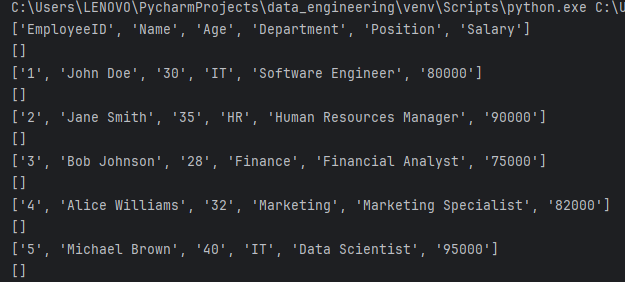
except FileNotFoundError:

print(f"File not found: {file\_path}")

except Exception as e:

print(f"An error occurred: {e}")

Output



Write data into CSV file

def csv\_file\_write(file\_path):

try:

with open(file\_path, 'r') as csvfile:

csv\_read = csv.reader(csvfile)

with open("example2.csv", 'w') as csvfile2:

csv\_write = csv.writer(csvfile2, delimiter='\t')

for row in csv\_read:

csv\_write.writerow(row)

except FileNotFoundError:

print("File not found ")

Write data into CSV file using Data dictionary

def csv\_dictionary\_write(file\_path):

with open(file\_path, 'r') as csv\_file:

csv\_reader = csv.DictReader(csv\_file)

with open('example3.csv', 'w') as csv\_file2:

fieldnames = ['EmployeeID','Name','Age','Department','Position','Salary']

csv\_writer = csv.DictWriter(csv\_file2, fieldnames=fieldnames, delimiter='\t')

# csv\_writer.writeheader()

for line in csv\_reader:

csv\_writer.writerow(line)

Read data using Dictionary

def read\_csv\_file(file\_path):

try:

with open(file\_path, 'r') as csvfile:

# Creating a CSV reader object

csv\_reader = csv.DictReader(csvfile)

# Reading a nd printing each row in the CSV file

for row in csv\_reader:

print(row['EmployeeID'])

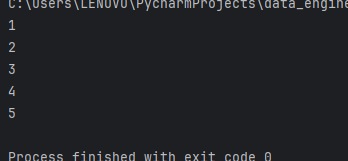
except FileNotFoundError:

print(f"File not found: {file\_path}")

except Exception as e:

print(f"An error occurred: {e}")

Output



Python lists

def list\_program():

numbers = [1, 2, 3, 4, 5]

print("Original List:", numbers)

numbers.append(6)

print("List after adding an element:", numbers)

numbers.remove(3)

print("List after removing an element:", numbers)

print("Element at index 2:", numbers[2])

numbers[1] = 10

print("List after modifying an element:", numbers)

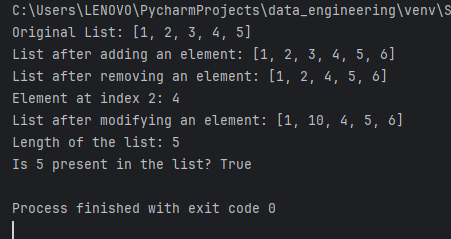
list\_length = len(numbers)

print("Length of the list:", list\_length)

is\_present = 5 in numbers

print("Is 5 present in the list?", is\_present)

Output



Lambda function

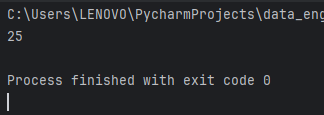
def square():

square = lambda x: x \*\* 2

print(square(5))

square()

Output



**Lambda function uses**

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. They're also useful when you want to use the function once.

**Practical use of lambda function**

You can write a Lambda function to **generate custom metrics by aggregating raw data**. Websites – Suppose you are creating a website and you want to host the backend logic on Lambda. You can invoke your Lambda function over HTTP using Amazon API Gateway as the HTTP endpoint.

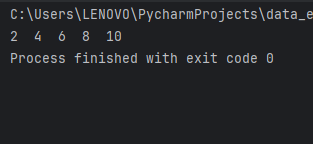
**Using lambda function**

map()

data = [1, 2, 3, 4, 5]

result1 = map(lambda x: x \* 2, data)

Output

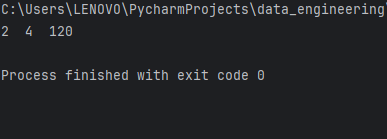


filter()

data = [1, 2, 3, 4, 5]

result2 = filter(lambda x: x % 2 == 0, data)

Output

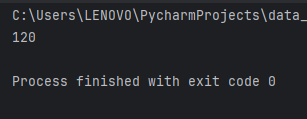
****

Reduce()

data = [1, 2, 3, 4, 5]

result3 = reduce(lambda x, y: x \* y, data)

Output



Filter data in python Lists using lambda and filter

data = [1, 2, 3, 4, 5]

result2 = filter(lambda x: x >3, data)

Output

