**Department of Electrical Engineering and   
Computer Science**

**Faculty Member:** Dr. Ahmad Salman **Dated:** 20/09/2023

**Semester:** 7th **Section:** BEE 12C

**CS-471 Machine Learning**

Lab 2: Data Structures, NumPy Arrays and SciPy Functions

**Group Members**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **PLO4 - CLO4** | | **PLO5 -CLO5** | **PLO8 -CLO6** | **PLO9 -CLO7** |
| **Name** | **Reg. No** | **Viva / Quiz / Lab Performance** | **Analysis of Data in Lab Report** | **Modern Tool Usage** | **Ethics and Safety** | **Individual and Teamwork** |
|  |  | **5 Marks** | **5 Marks** | **5 Marks** | **5 Marks** | **5 Marks** |
| Afif Arif Siddiqi | 344504 |  |  |  |  |  |
| Muhammad Ali Farooq | 331879 |  |  |  |  |  |
| Muhammad Ahmed Mohsin | 333060 |  |  |  |  |  |
| Danial Ahmad | 331388 |  |  |  |  |  |
| Ehtishaam Tanveer | 333074 |  |  |  |  |  |
| Muhammad Umer | 345834 |  |  |  |  |  |

**Table of Contents**

[2 Introduction to Python 3](#_Toc146295400)

[2.1 Introduction 3](#_Toc146295401)

[2.2 Objectives 3](#_Toc146295402)

[2.3 Theory 3](#_Toc146295403)

[3 Lab Tasks 4](#_Toc146295404)

[3.1 Task 1 4](#_Toc146295405)

[3.2 Task 2 4](#_Toc146295406)

[3.3 Task 3 5](#_Toc146295407)

[3.4 Task 4 6](#_Toc146295408)

[3.5 Task 5 7](#_Toc146295409)

[3.6 Task 6 8](#_Toc146295410)

[3.7 Task 7 9](#_Toc146295411)

[4 Conclusion 11](#_Toc146295412)

# Data Structures, NumPy Arrays and SciPy Functions

## Introduction

This laboratory exercise is focused on the introduction of data structures native in python particularly Lists and Dictionaries which are very commonly used. This lab also introduces the importing of modules that are used for machine learning tasks. In this lab, the NumPy and SciPy libraries will be introduced which are very important to the field of Machine Learning.

## Objectives

The following are the main objectives of this lab:

* Implement data structures such as lists and dictionaries in python
* Create, alter, and loop through lists
* Use slicing to access range of items in a list
* Utilize various list methods such as append, insert, extend, remove, pop etc.
* Create and implement a dictionary
* Create NumPy arrays and perform matrix operations and broadcasting
* Use SciPy for minimization, scarce matrices, and interpolation

## Theory

Data structures are an important part of python. The 4 main data structures are lists, tuples, sets and dictionaries. Lists and dictionaries are the most used for machine learning tasks. The *import* keyword is used to load modules and libraries. In machine learning, there are many popular libraries. The most basic of these is the NumPy library which provides an optimized array implementation for very fast matrix computations necessary for machine learning. The SciPy library provides numerous functions for scientific computations.

A summary of the list functions in python is provided below:

**append(I)** append item I to the end of the list

**insert(i, I)** insert item I at i position of the list

**extend(L)** extend/concatenate a second list L

**remove(I)** remove a specified item I from a list

**pop(i)** remove item at specific index i in the list

**count(I)** returns total number of a specific item I from a list

**index(I)** return index of first occurrence of a specific item I

**reverse** reverse the items of the list

# Lab Tasks

## Task 1

Create a 1-D list containing the characters of the name of any one person in your group. Loop through the list and display each character on a new line.

### Task 1 Code Starts Here ###

name\_list = ["M", "u", "h", "a", "m", "m", "a", "d", " ", "U", "m", "e", "r"]

for character in name\_list:

    print(*f*"{character}")

### Task 1 Code Ends Here ###

### Task 1 Screenshot Starts Here ###



### Task 1 Screenshot Ends Here ###

## Task 2

Write a program that repeatedly prompts the user for input. The user will keep entering numbers which are added to a list. Each time a number is added to the list, it must be placed in such a way that the list items are always in ascending order. Each time a number is input, the list is to be printed showing the newly added number. This continues until the word “done” is input at which point the prompts will stop. The final list is then displayed. Do NOT use any inbuilt sorting function for this task.

### Task 2 Code Starts Here ###

number\_list = []

while True:

    number = input("Enter an input: ")

    if number == "done":

        break

    else:

        number = *int*(number)

        for index, item in enumerate(number\_list):

            if number < item:

                number\_list.insert(index, number)

                break

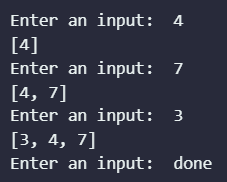
        else:

            number\_list.append(number)

        print(number\_list)

### Task 2 Code Ends Here ###

### Task 2 Screenshot Starts Here ###



### Task 2 Screenshot Ends Here ###

## Task 3

Create a list with the sequence 1, 2, 3… 20. Then using the slice operation (:) on this list, print the following sub-lists:

5, 6, 7… 20

1, 2, 3… 12

7, 8, 9 … 16

4, 5

11, 12, 13, 14

### Task 3 Code Starts Here ###

main\_list = *list*(range(1, 20+1))

print("Sub-list 1: ", main\_list[4:])

print("Sub-list 2: ", main\_list[0:12])

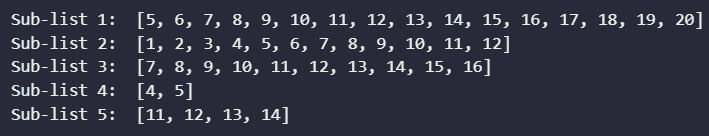
print("Sub-list 3: ", main\_list[6:16])

print("Sub-list 4: ", main\_list[3:5])

print("Sub-list 5: ", main\_list[10:14])

### Task 3 Code Ends Here ###

### Task 3 Screenshot Starts Here ###



### Task 3 Screenshot Ends Here ###

## Task 4

In this task, you will make use of dictionaries. Write a program that first prompts the user to input five strings which will be the keys of the dictionary. Then, the program must prompt the user to input the values of the respective keys. When entering the values, the user must be shown the key whose value is being input. Once all values are entered, display the dictionary.

### Task 4 Code Starts Here ###

*# create an empty dictionary*

task\_dict = {}

*# prompt the user to input five keys*

for i in range(5):

    key = input("Enter key " + *str*(i + 1) + ": ")

    task\_dict[key] = None

*# prompt the user to input values for each key*

for key in task\_dict:

    value = input("Enter value for " + key + ": ")

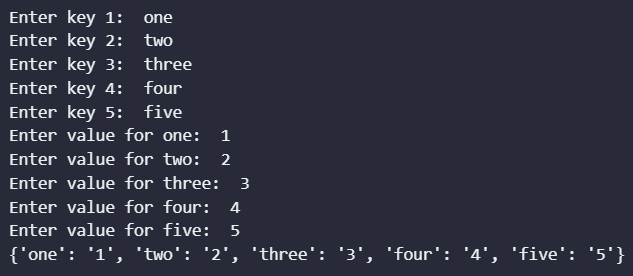
    task\_dict[key] = value

*# display the dictionary*

print(task\_dict)

### Task 4 Code Ends Here ###

### Task 4 Screenshot Starts Here ###

****

### Task 4 Screenshot Ends Here ###

## Task 5

Import the NumPy Library. Use the np.array function to define a 4x5 array with elements of your choice. Ensure that the elements are numbers (not strings). Then, perform the following:

* Print the array
* Print element(4,4)
* Print rows 2 and 3 via slicing
* Print the central 3x3 elements in a matrix
* Compute the sum of the matrix elements
* Compute the sum of the matrix elements along axis 0
* Compute the sum of the matrix elements along axis 1
* Compute the mean of the matrix elements
* Compute the standard deviation of the matrix elements

### Task 5 Code Starts Here ###

import numpy as np

arr = np.random.randint(1, 10, (5, 5))

print("Array: \n", arr)

print("\nElement (4, 4): ", arr[3, 3])

print("\nRows 2 and 3: \n", arr[1:3, :])

print("\nCentral 2x3 Elements: \n", arr[1:4, 1:4])

print("\nSum of Matrix Elements: ", np.sum(arr))

print("\nSum of Matrix Elements along Axis 0: ", np.sum(arr, *axis*=0))

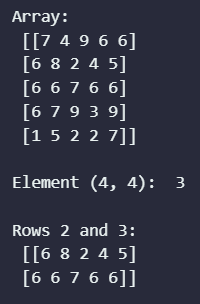
print("\nSum of Matrix Elements along Axis 1: ", np.sum(arr, *axis*=1))

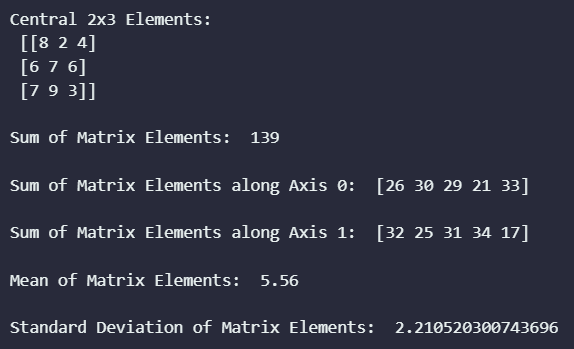
print("\nMean of Matrix Elements: ", np.mean(arr))

print("\nStandard Deviation of Matrix Elements: ", np.std(arr))

### Task 5 Code Ends Here ###

### Task 5 Screenshot Starts Here ###





### Task 5 Screenshot Ends Here ###

## Task 6

Use the np.array function to define two matrices of size 3x3. Place numerical elements of your choice in the matrices. Write code to perform the following:

* Print the arrays
* Compute the sum of the matrices
* Compute the difference of the matrices
* Compute the element-wise product of the matrices
* Compute the element-wise division of the matrices
* Compute the matrix multiplication of the matrices

### Task 6 Code Starts Here ###

arr\_1 = np.random.randint(1, 10, (3, 3))

arr\_2 = np.random.randint(1, 10, (3, 3))

print("\nSum of Matrices: \n", arr\_1 + arr\_2)

print("\nDifference of Matrices: \n", arr\_1 - arr\_2)

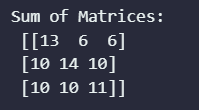
print("\nElement-wise Product of Matrices: \n", arr\_1 \* arr\_2)

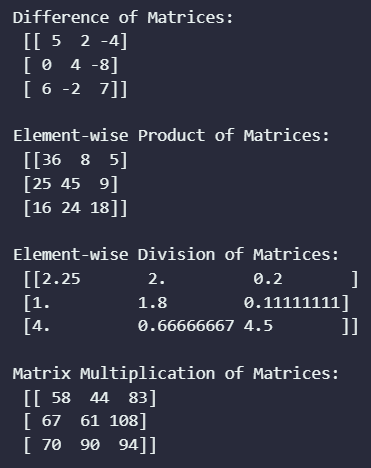
print("\nElement-wise Division of Matrices: \n", arr\_1 / arr\_2)

print("\nMatrix Multiplication of Matrices: \n", arr\_1 @ arr\_2)

### Task 6 Code Ends Here ###

### Task 6 Screenshot Starts Here ###

****

****

### Task 6 Screenshot Ends Here ###

## Task 7

In this task, you will use various functions of the SciPy library that are commonly used in machine learning. Import the various modules from SciPy Library:

from scipy.optimize import root

from scipy.optimize import minimize

from scipy.sparse import csr\_matrix

from scipy.interpolate import interp1d

1. Use the root function to determine the roots of the equation 3x2 + 2x – 10.
2. In machine learning, it is very often required to find the argument that minimizes a complex equation with the given data. Use the minimize function to determine the roots of the equation x2 - 20x + 45.
3. In machine learning, sometimes there are matrices in which most of the elements are zero. In such cases, it is more convenient to store them as sparse matrices which holds information of the non-zero elements. In this task, create a 3x10 sparse matrix (A) with elements of your choice. Ensure that about 2/3 of the elements are zero. Then, print the matrix information using csr\_matrix(A), csr\_matrix(A).data and csr\_matrix(A).count\_nonzero().
4. In this task, you will perform interpolation. Create two lists x and y. The list x contains elements 1,2,3… 10. The list y contains the elements for y = 2x + 1. Use the interp\_func = interp1d(x, y) to get the interpolating function. Then, use interp\_func(val) to get any 3 interpolated values.

### Task 7 Code Starts Here ###

from scipy.optimize import root

from scipy.optimize import minimize

from scipy.sparse import csr\_matrix

from scipy.interpolate import interp1d

*# a)*

equation = *lambda* *x*: 3\*x\*\*2 + 2\*x - 10

root\_eqn = root(equation, 0)

print("a)")

print(*r*"Roots of Equation 3x^2 + 2x - 10: ", root\_eqn.x)

print(*f*"Value at {root\_eqn.x}: ", equation(root\_eqn.x))

*# b)*

equation = *lambda* *x*: x\*\*2 - 20\*x + 45

print("\nb)")

root\_eqn = minimize(equation, 0)

print(*r*"Roots of Equation x^2 - 20x + 45: ", root\_eqn.x)

print(*f*"Value at {root\_eqn.x}: ", equation(root\_eqn.x))

*# c)*

print("\nc)")

arr = np.array([[1, 0, 0, 0, 0, 0, 3, 0, 0, 0],

              [2, 0, 0, 0, 0, 0, 0, 0, 8, 0],

              [3, 0, 0, 0, 0, 5, 0, 0, 0, 0]])

sparse\_arr = csr\_matrix(arr)

print("Sparse Matrix: \n", sparse\_arr)

print("\nSparse Matrix Data: \n", sparse\_arr.data)

print("\nSparse Matrix Non-Zero Elements: \n", sparse\_arr.count\_nonzero())

*# d)*

print("\nd)")

x = *list*(range(1, 10+1))

y = [(2\*i) + 1 for i in x]

print("\nx = ", x)

print("y = ", y)

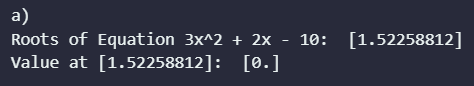
interp\_fun = interp1d(x, y)

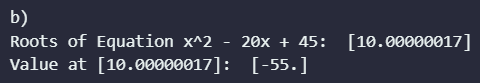
interp\_fun\_val = interp\_fun([1.5, 2.5, 3.5])

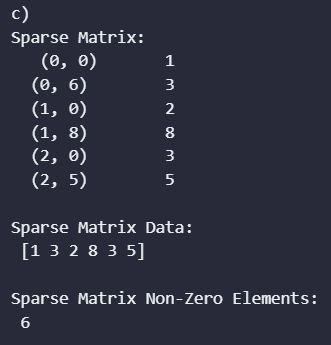
print("\nInterpolated Values at: ", [1.5, 2.5, 3.5], "\n\t\t\t", interp\_fun\_val)

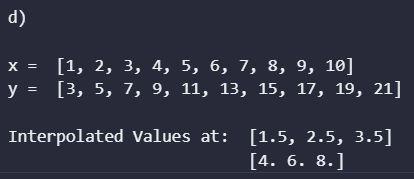
### Task 7 Code Ends Here ###

### Task 7 Screenshots Starts Here ###









### Task 7 Screenshots Ends Here ###

# Conclusion

This laboratory exercise has provided a solid introduction to data structures in Python, specifically lists and dictionaries, which are two of the most commonly used data structures in the language. We have also learned about NumPy and SciPy, two important Python libraries for machine learning. Lists and dictionaries are essential for storing and manipulating data in Python. Lists are used to store a collection of items in a specific order, while dictionaries are used to store key-value pairs. NumPy provides a variety of data structures and functions for working with numerical data, while SciPy provides a variety of high-level mathematical functions for scientific computing. Together, these libraries provide a powerful toolkit for machine learning.