

**NumPy Package**

**Introduction**

During this lab, you will learn about the Python data analysis with NumPy package.

**Estimated Time**

30 minutes

**Objectives**

At the end of this lab, you will be able to:

* Get prepared to use the NumPy package in Python code.
* Use data structures provided by NumPy.
* Do data processing using functions and methods provided by NumPy.

**Logon Information**

Use the following credentials to sign into virtual environments.

* Username: **Administrator**
* Password: **Passw0rd**!

Table of Contents

[Lab: NumPy package 3](#_Toc521939994)

[Exercise 1: Calculate basic statistics 3](#_Toc521939995)

Lab: NumPy Package

During this lab, you will learn about the Python data analysis environment using VS code.

Exercise 1: Calculate basic statistics

In this exercise, you will learn how to use NumPy through the calculation of basic statistics, the basis of data analysis.

Tasks

1. Start Anaconda Prompt

Start Anaconda Prompt as an administrator from the **Start Menu**. If you can't find it in Start Menu, search for it by typing **Anaconda** in the search bar. Then, right-click Anaconda Prompt and select **Run as administrator**.

Select **Yes** if the User Account Control pop-up appears.

1. Install NumPy package

Python packages need to be installed for each virtual environment. For this lab, since we are going to use conda base environment, NumPy is **already installed** by default. **If you use another virtual environment**, you should run the following command in Anaconda prompt to install numpy.

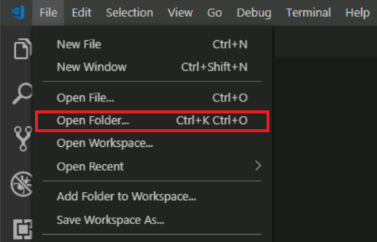
conda install numpy

1. Start Visual Studio Code

Start Visual Studio Code from the **Start Menu**. If you can't find it in Start Menu, search for it by typing **Visual** in the search bar.

If you have used VS code in another task before, the VS code will start with the workspace folder opened. In that case, select **Close Folder** from the **File** menu.

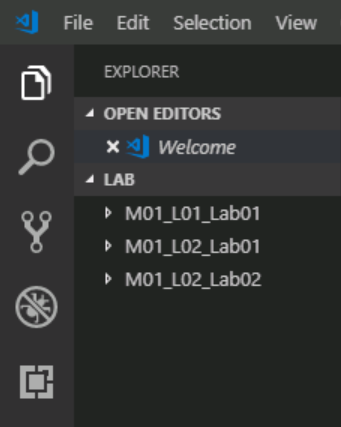
1. Open Folder
2. Open the Lab folder from the File menu. Select the File menu and choose Open Folder.



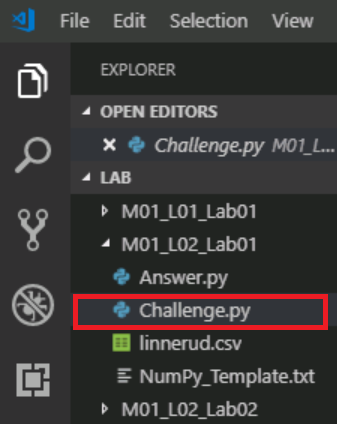
1. In the dialog box that appears, type the folder path as below:

C:\Labs\Module1\LAB

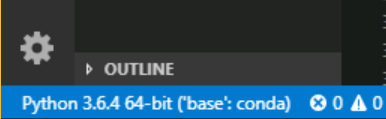
Then, click the **Select Folder** button. When the folder opens, Explorer area of the screen appears in the **Side bar**.



1. Open File
2. Select the **M01\_L02\_Lab01** folder on the side bar and select **Challenge.py**. This file is a template for a series of analysis.



Please make sure that conda base environment is activated.



1. Loading data
2. You were contracted to work to analyze the weight data of 20 people. You have already created code to load data received from the client. (Please check the data loading part of **Challenge.py** as below)

Please note that the answers for this challenge are located in **Answer.py** file.

###############  
# Data preparation  
###############  
import csv  
header = next(csv.reader(open('C:\\Labs\\M01\_L02\_Lab01\\linnerud.csv','r')))  
  
import numpy as np  
data = np.loadtxt('C:\\Labs\\M01\_L02\_Lab01\\linnerud.csv', delimiter=',', skiprows=1)  
weights\_pounds = data.T[1] # T is transpose  
print('Header : ', header[1])  
print(weights\_pounds)

You are going to see if the data is loaded correctly. **Run** the above code **sequentially** on VS Code to see how the data is loaded.

You can use T to get the transpose of the target two-dimensional array.

If the data is successfully loaded, an array similar to the following will be the output:

[191. 189. 193. 162. 189. 182. 211. 167. 176. 154. 169. 166. 154. 247. 193. 202. 176. 157. 156. 138.]

1. Processing data
2. You have discovered that the unit of weight data is a pound. The client has been asked to perform a kilogram-based analysis and must process the data.

Create a unit conversion process in Python. The converted data should be stored in the variable **weights\_kg**. It is recommended that you write the code in the Processing Data region.

If you want to convert a unit from pounds to kilograms, divide the value by 2.2046.

If you calculate properly, you will get the following results for the **weights\_kg** variable:

[ 86.63703166 85.72983761 87.54422571 73.48271795 85.72983761 82.55465844 95.70897215 75.75070308 79.8330763 69.85394176 76.65789712 75.29710605 69.85394176 112.03846503 87.54422571 91.62659893 79.8330763 71.21473283 70.76113581 62.59638937]

Hint - the code should be similar to:

new\_variable = current\_variable / 2.2046

1. Calculate basic statistics
2. Once you've successfully completed the unit conversion, let's get down to analyzing the data. The client has asked to calculate the **average**, **standard deviation**, **minimum**, **maximum**, and **median** as statistics. He also requires **25 percentile** values, **50 percentile** values, and **75 percentile** values.

Use the functions provided by NumPy to calculate these statistics. You do not have to store the calculation results in a variable.

NumPy function List:

Mean : np.mean()

Standard Deviation : np.std()

Minimum : np.min()

Maximum : np.max()

Median : np.median()

Percentile : np.percentile()

If you calculate properly, you will get the following results:

Mean : 81.01242855846866

Standard Deviation : 10.915961061382129

Minimum : 62.596389367685745

Maximum : 112.03846502766942

Median : 79.8330762950195

25 percentile : 72.91572167286583

50 percentile : 79.8330762950195

75 percentile : 86.86383017327407

Hint – the answer starts with:

np.mean(weights\_kg)

And for percentile, the first one is:

np.percentile(weights\_kg,25)

Please note that the answers for the challenge is located in **Answer.py** file.

1. Random Choice
2. You remembered that there was a request from the client to consider the process of **extracting the weight of any three persons**. You have decided to apply masking to this process.

Create an **index for masking with random numbers** and extract the body weight data for 3 persons.

To generate random numbers, use the np.random.randint() function provided by NumPy.

If it can be extracted correctly, the following results can be obtained, for example. However, the results are different for each process because they include random numbers processing.

[ 95.70897215, 62.59638937, 91.62659893]

Use the following code:

index = np.random.randint(0,20,3)

print(weights\_kg[index])

You can find more information about this function here: <https://docs.scipy.org/doc/numpy-1.15.1/reference/generated/numpy.random.randint.html>

1. Count by range
2. The last requirement from your client is to **summarize the number for each data range**. The range of data requires:
   * 1. less than 70 kg,
     2. 70 – 80 kg,
     3. 80 – 90 kg
     4. more than 90kg.

Count the number of data that fits each condition.

A bit logic operation is possible between the masking arrays. For example, the result of the logical product operation “&“ (AND) of [True, False] and [True, True] is [True, False].

1. If you calculate properly, you will get the following results:
   * 1. less than 70kg: 3
     2. 70 – 80kg: 8
     3. 80 – 90kg: 6
     4. more than 90kg: 3

You can find the answers for i) and ii) below, follow the same pattern for iii) and iv).

np.count\_nonzero(weights\_kg[(weights\_kg < 70)])

np.count\_nonzero(weights\_kg[(weights\_kg >= 70) & (weights\_kg < 80)])

You have now completed all customer requests.

Exercise 1 has been completed.