Chapter
PYTHON

Dr. Minhhuy Le

606-A4, EEE, Phenikaa Uni.

huy.leminh@phenikaa-uni.edu.vn

Outline

Chapter 4

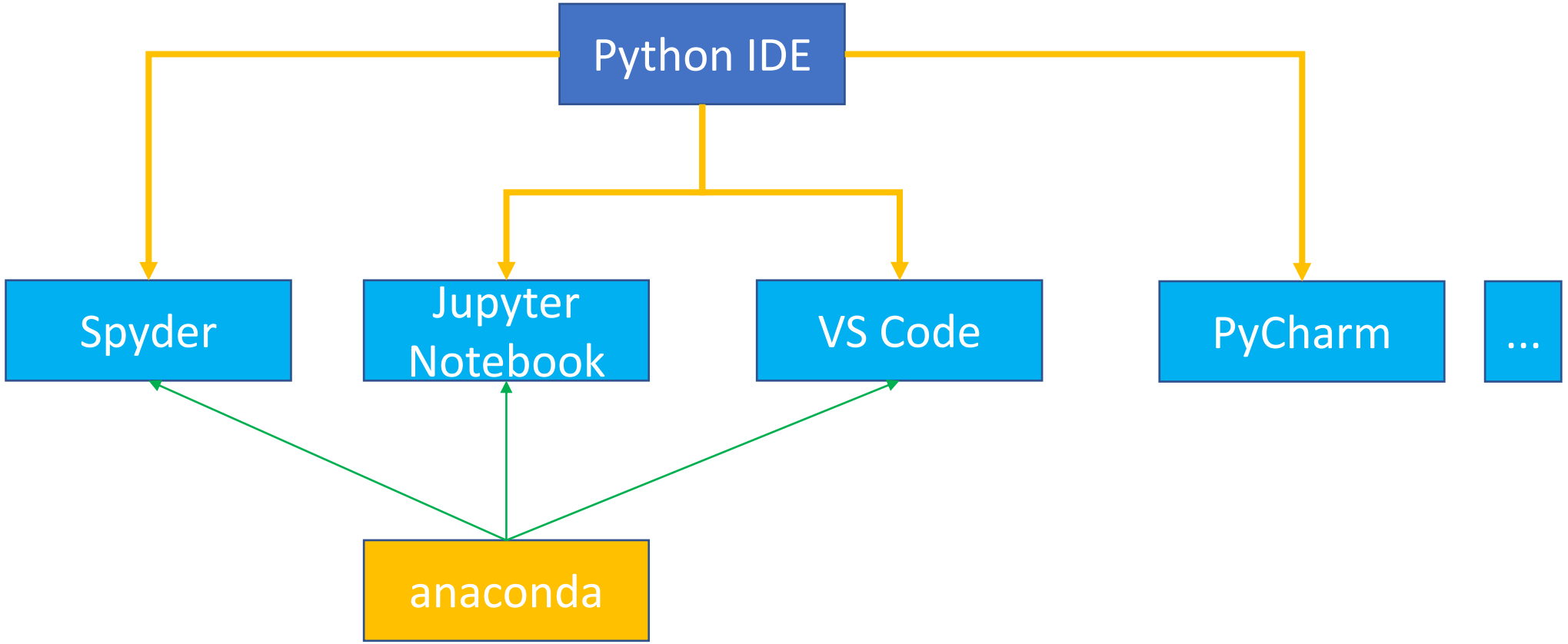
1. Python Programming Development
2. Characters, List, Files
3. Loops structures and Booleans, compare to MATLAB
4. Function and Class
5. Plots

1. Python Programming Development

- Created by Guido Van Rossum, in the Netherlands, 1991.
- High-level, general purpose programming language -> **Code readability**
- Name "**Python**": name of comedy group **Monty Python** (British) for **fun to use**
- Based on **C language**
- **Simple enough** to be used by beginning programmers, but **powerful enough** to attract professionals
- Current versions: **2.7x** (discontinued from 2020) and 3.x (supported)
- **Open-source software**
- Official website: www.python.org

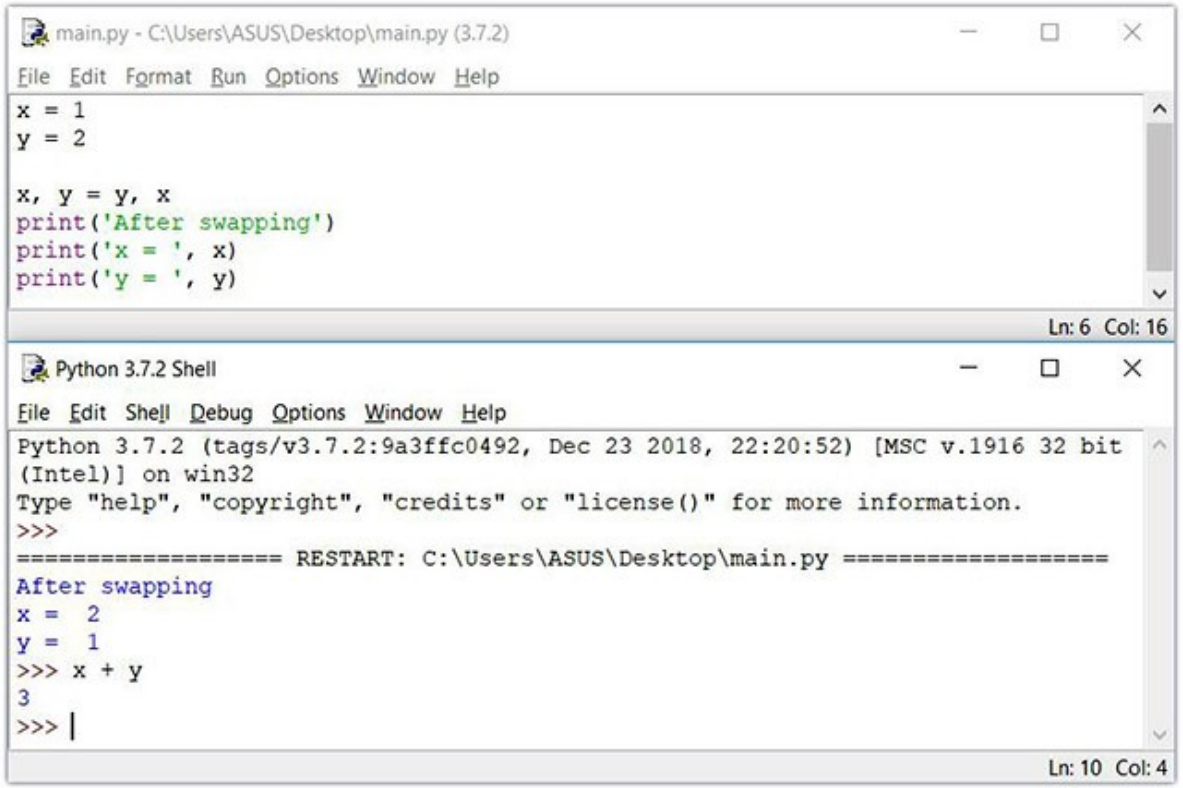
Most Popular Programming Languages 1965 - 2020

Integrated Development Environment (IDE)

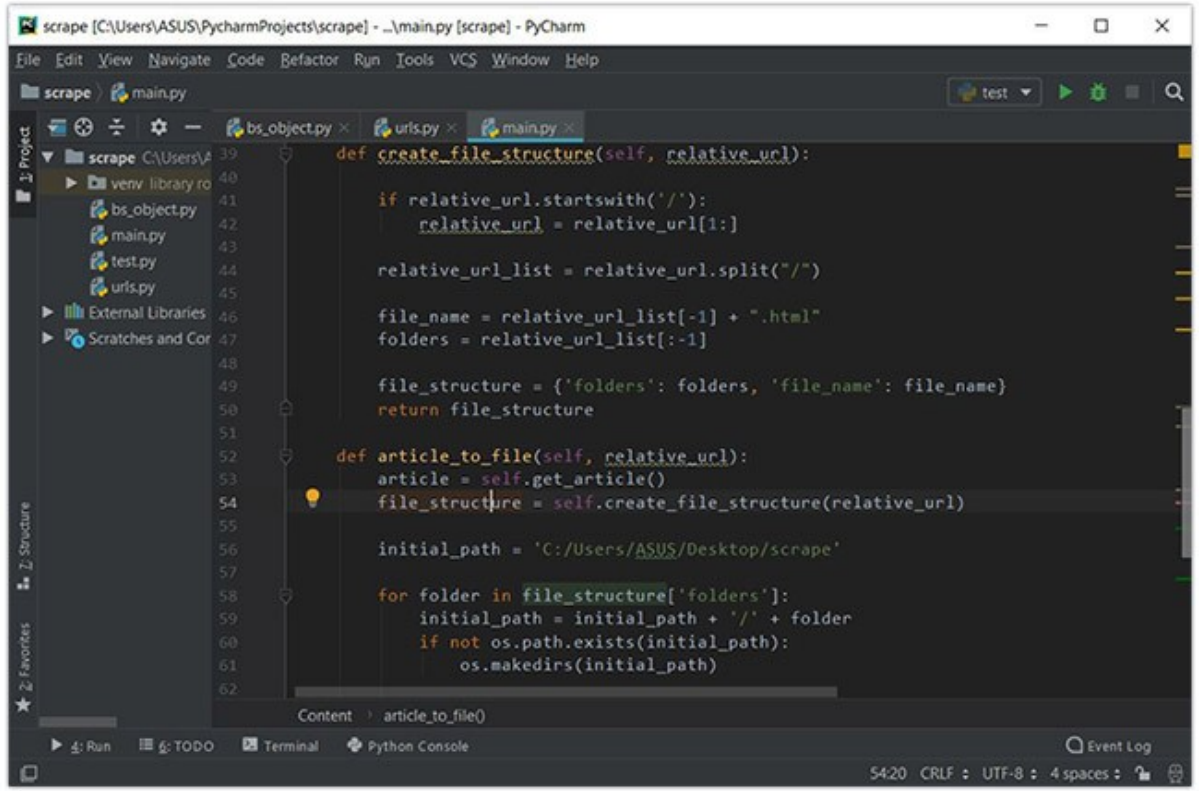


Integrated Development Environment (IDE)

Original IDE

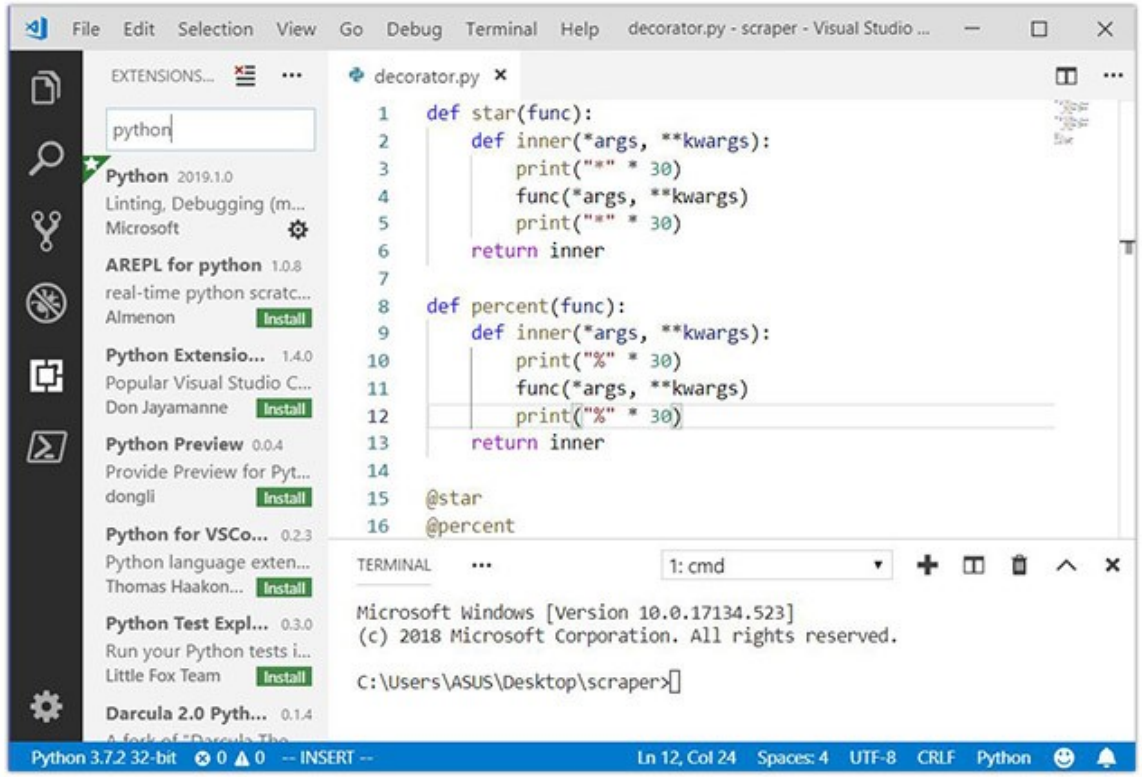


PyCharm

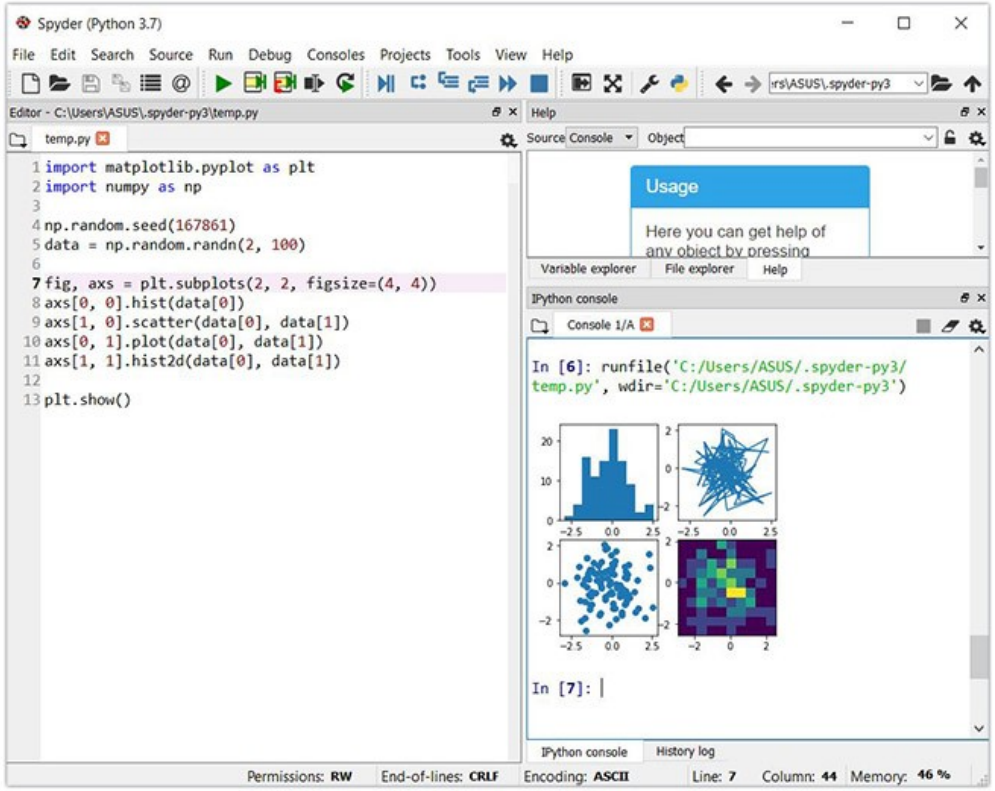


Integrated Development Environment (IDE)

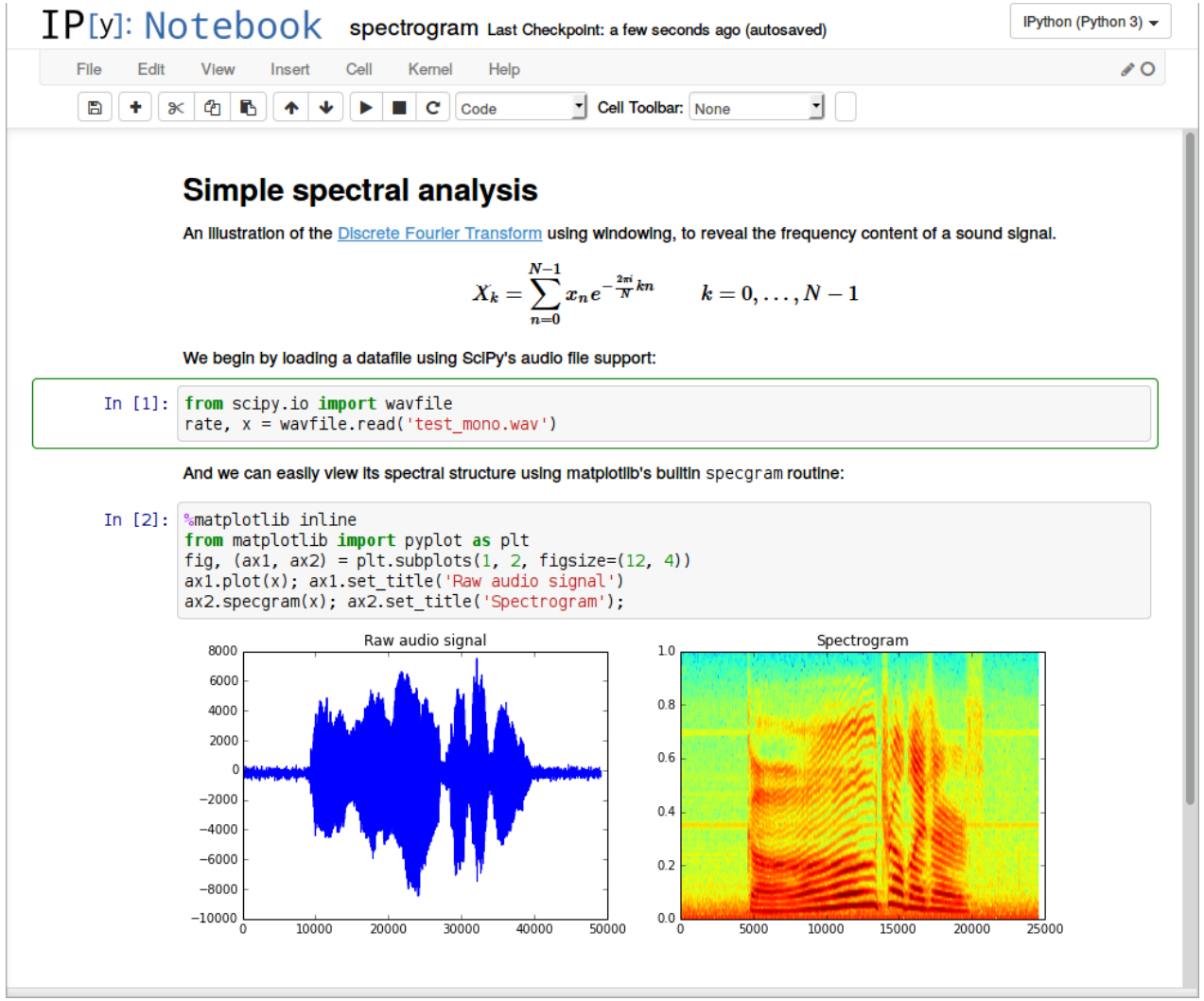
VS Code



Spyder

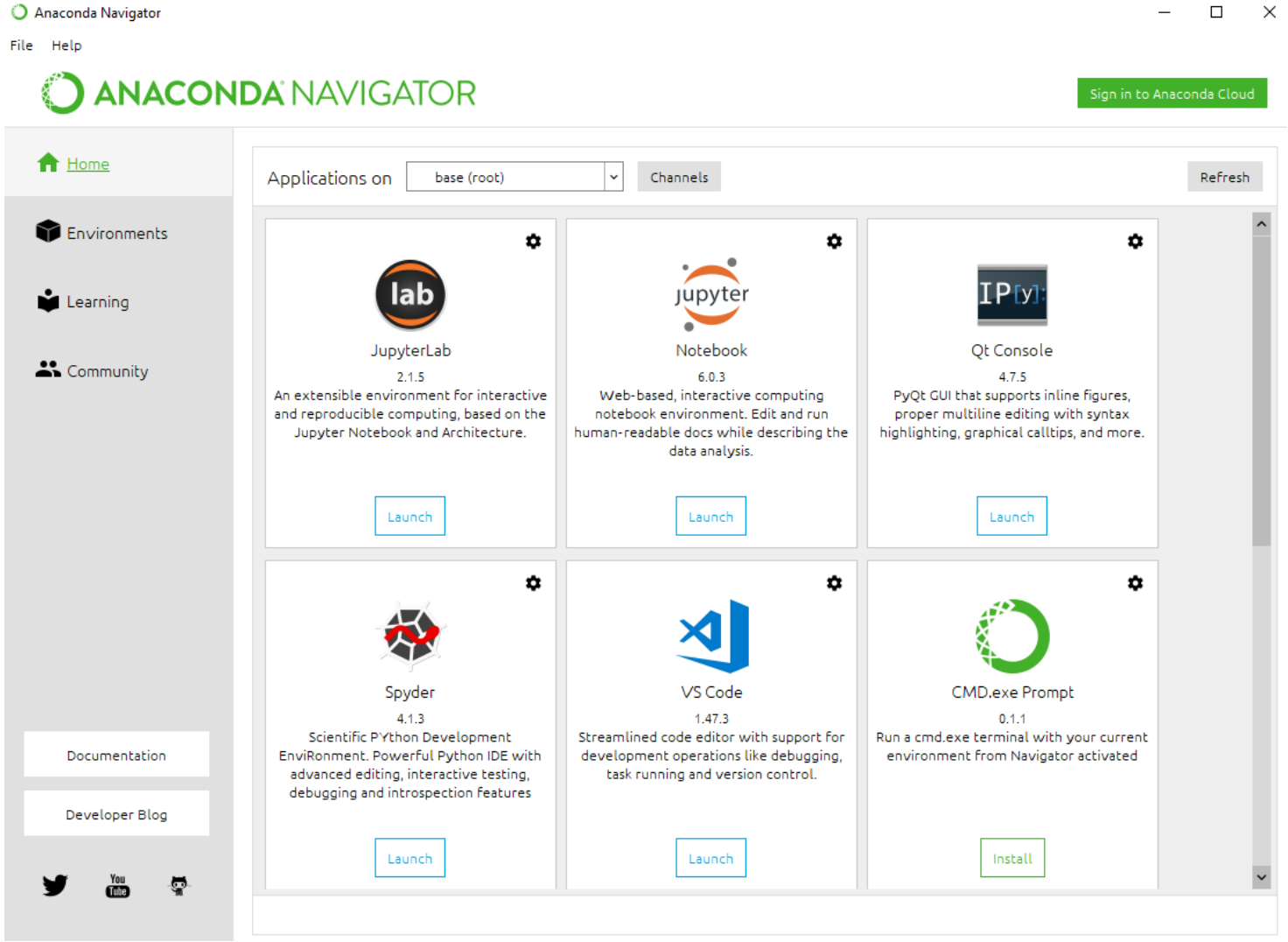


Integrated Development Environment (IDE)



Jupyter Notebook (.ipynb)

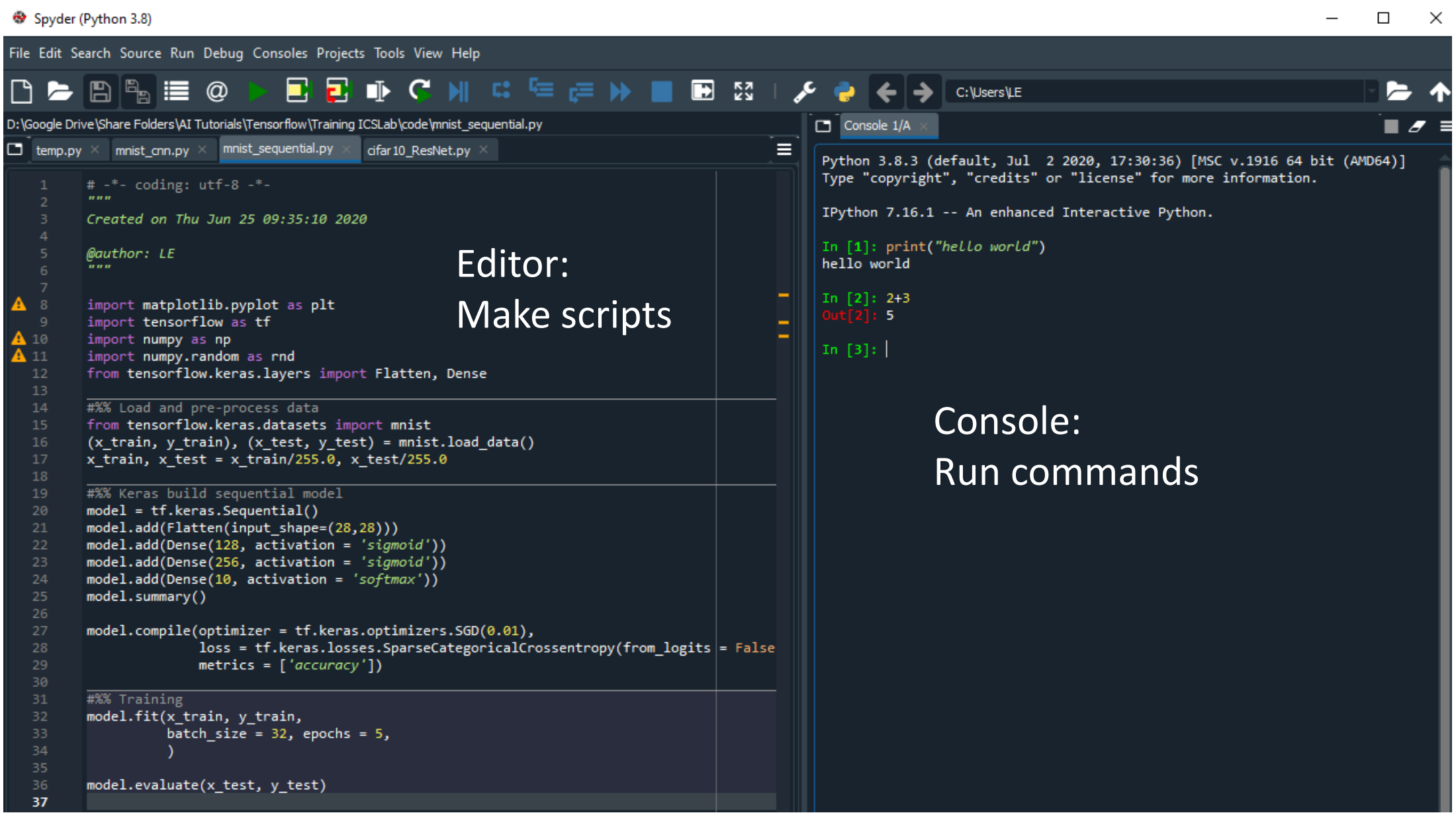
Integrated Development Environment (IDE)



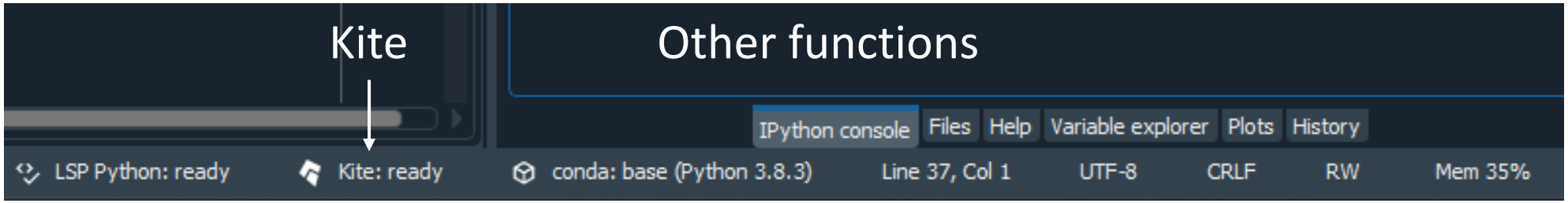
anaconda

1. Python Programming Development

"Hello World"



- **Editor**: to make script
- **Console**: to run commands or script
- **Others functions**: History, Plot display, Help, Variables ...
- **Kite**: Free AI coding assistant for auto-complete code
- To use a library: import *library_name*
- Many Free/Open-source Libraries: data science, numerical, AI ...



temp.py | mnist_cnn.py | mnist_sequential.py | cifar10_ResNet.py | example_1.py

```python
import matplotlib.pyplot as plt
import numpy as np

def main():
    print("This is main function")
    x = eval(input("Enter a number between 0 and 1: "))
    y = np.zeros((10,1))
    for i in range(10):
        y[i] = x * i
        print(y[i])

    plt.plot(y)
main()
```

File name

Import libraries

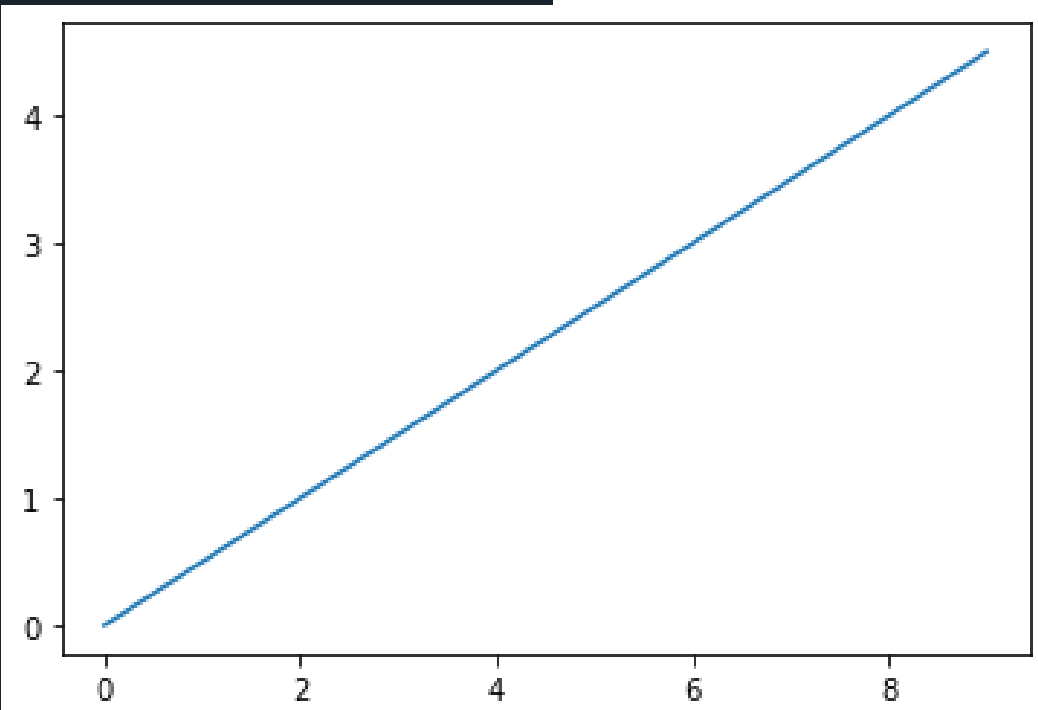
Define a function “main”

Call function “main”

Hit **F5** on the script file (Editor) to **run** the script

```
In [1]: runfile('D:/Google Drive/Share Folders/EEE/Lectures/23 Basic
programming for Electronics/Ref/Python examples/example_1.py', wdir='D:/
Google Drive/Share Folders/EEE/Lectures/23 Basic programming for
Electronics/Ref/Python examples')
This is main function

Enter a number between 0 and 1: 0.5
[0.]
[0.5]
[1.]
[1.5]
[2.]
[2.5]
[3.]
[3.5]
[4.]
[4.5]
```

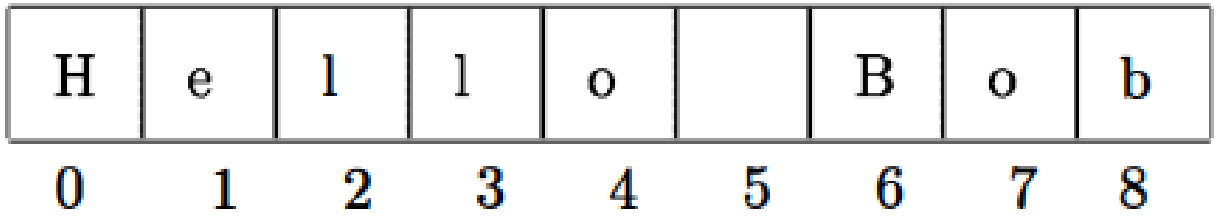


String is a sequence of **Characters:** “*string*” or ‘*string*’

```
In [2]: str1 = "Hello Bob"

In [3]: str2 = 'Hello Bob'

In [4]: str1 == str2
Out[4]: True
```



String slicing:

Str1[0] -> “H”

Str1[0:2] -> “He”

Str1[:2] -> “He”

String concatenate:

“*Hello*” + “*World*” -> “*Hello World*”

“Spam”*3 -> “SpamSpamSpam”

String operations

operator	meaning
+	concatenation
*	repetition
<string>[ ]	indexing
<string>[:]	slicing
len(<string>)	length
for <var> in <string>	iteration through characters

String concatenate:

"Hello" + *"World"* -> *"Hello World"*

"Spam"*3 -> "SpamSpamSpam"

String slicing:

Str1[0] -> "H"

Str1[0:2] -> "He"

Str1[:2] -> "He"

String methods

strip(): remove whitespace from the beginning or the end
“ Hello ”.strip() -> “Hello”

upper(): returns the string in upper case
“Hello”.upper() -> “HELLO”

lower(): returns the string in lower case
“Hello”.upper() -> “hello”

replace(): replaces a string with another string
“Hello”.replace(“l”,”A”) -> “HeAAo”

split(): split string into substring
“Hello, World”.split(“,”) -> [“Hello”, “Wold”] -> List of substring

List is a kind of sequence: -> Indexing, slicing, concatenating

```
>>> [1,2] + [3,4]
[1, 2, 3, 4]
>>> [1,2]*3

[1, 2, 1, 2, 1, 2]
>>> grades = ['A','B','C','D','F']
>>> grades[0]
'A'
>>> grades[2:4]
['C', 'D']
>>> len(grades)
5
```

List can be a sequence of arbitrary objects

```
myList = [1, "Spam", 4, "U"]
```

myList[1] -> “Spam”
myList[:-1] -> [1, “Spam”, 4]

Loop through a list items

```
thislist = ["apple", "banana", "cherry"]
for x in thislist:
  print(x)
```

→

```
apple
banana
cherry
```

Check if an item in list

```
thislist = ["apple", "banana", "cherry"]
if "apple" in thislist:
  print("Yes, 'apple' is in the fruits list")
```

→ "Yes, 'apple' is in the fruits list"

Length of the list

len(thislist) -> 3

Add an item to the end of the list: append()

```
thislist = ["apple", "banana", "cherry"]
thislist.append("orange")
```

→ ["apple", "banana", "cherry", “orange"]

Insert an item as the second position: insert()

```
thislist = ["apple", "banana", "cherry"]
thislist.insert(1, "orange")
```

→ ["apple", “orange”, "banana", "cherry"]

Remove an item: remove()

```
thislist = ["apple", "banana", "cherry"]
thislist.remove(" apple ")
```

→ ["banana", "cherry"]

Remove an item by : remove()

```
thislist = ["apple", "banana", "cherry"]
thislist.remove(" apple ")
```

→ ["banana", "cherry"]

Join two lists: extend() or “+”

```
list1 = ["a", "b" , "c"]
list2 = [1, 2, 3]
list1.extend(list2)
```

list3 = list1 + list2

→ ['a', 'b', 'c', 1, 2, 3]

Collections:

- **List** is a collection which is ordered and changeable. Allows duplicate members.
- **Tuple** is a collection which is ordered and unchangeable. Allows duplicate members.
- **Set** is a collection which is unordered and unindexed. No duplicate members.
- **Dictionary** is a collection which is unordered, changeable and indexed. No duplicate members.

Reference: https://www.w3schools.com/python/python_lists.asp

Example

Write file

```
f = open("demofile2.txt", “w")
f.write("Now the file has more content!")
f.close()
```

Read file

```
f = open("demofile2.txt", "r")
print(f.read())
```

Open -> Write/Read -> Close

Create a New file

To create a new file in Python, use the open() method, with one of the following parameters:
"x" - Create - will create a file, returns an error if the file exist
"a" - Append - will create a file if the specified file does not exist
"w" - Write - will create a file if the specified file does not exist

Delete a file

To delete a file, you must import the OS module, and run its os.remove() function:

```
import os
os.remove("demofile.txt")
```

```
import os
if os.path.exists("demofile.txt"):
  os.remove("demofile.txt")
else:
  print("The file does not exist")
```

Summary

- Few important elements of the Python string, list, and file objects
- **String**: is a sequence of characters
- **List**: is sequence of objects (mixing objects are fine)
- String & List methods
- Indexing
- Slicing
- **File**: read/write data to a file

Exercises

1. Given the initial statements:

```
s1 = "spam"
s2 = "ni!"
```

Show the result of evaluating each of the following string expressions.

a) `"The Knights who say, " + s2`
b) `3 * s1 + 2 * s2`
c) `s1[1]`
d) `s1[1:3]`
e) `s1[2] + s2[:2]`
f) `s1 + s2[-1]`
g) `s1.upper()`
h) `s2.upper().ljust(4) * 3`

2. Which of the following is the same as `s[0:-1]`?
a) `s[-1]` b) `s[:]` c) `s[:len(s)-1]` d) `s[0:len(s)]`

Summary to a report (pdf)

3. A certain CS professor gives 100-point exams that are graded on the scale 90–100:A, 80–89:B, 70–79:C, 60–69:D, <60:F. Write a program that accepts an exam score as input and prints out the corresponding grade.

4. Continues from Ex.3
Input: Student Name & score
Save to a file: Name – Grade
Then read the file, print on the screen

Chapter 4
PYTHON

Dr. Minhhuy Le

606-A4, EEE, Phenikaa Uni.

huy.leminh@phenikaa-uni.edu.vn

Outline

Chapter 4

1. Python Programming Development
2. Characters, List, Files
3. Loops structures and Booleans, compare to MATLAB
4. Function and Class
5. Plots

Exercises

Previous

1. Given the initial statements:

```
s1 = "spam"
s2 = "ni!"
```

Show the result of evaluating each of the following string expressions.

a) `"The Knights who say, " + s2`
b) `3 * s1 + 2 * s2`
c) `s1[1]`
d) `s1[1:3]`
e) `s1[2] + s2[:2]`
f) `s1 + s2[-1]`
g) `s1.upper()`
h) `s2.upper().ljust(4) * 3`

2. Which of the following is the same as `s[0:-1]`?
a) `s[-1]` b) `s[:]` c) `s[:len(s)-1]` d) `s[0:len(s)]`

Summary to a report (pdf)

3. A certain CS professor gives 100-point exams that are graded on the scale 90–100:A, 80–89:B, 70–79:C, 60–69:D, <60:F. Write a program that accepts an exam score as input and prints out the corresponding grade.

4. Continues from Ex.3
Input: Student Name & score
Save to a file: Name – Grade
Then read the file, print on the screen

3. Loops structures and Booleans, compare to MATLAB

- For loop
- While loop
- Common loop patterns

```
for <var> in <sequence>:
        <body>
```

Sequence: string, list….

```
while <condition>:
        <body>
```

condition: true/false

3. Loops structures and Booleans, compare to MATLAB

For loop example:

$$m = \frac{1}{n}\sum_{i=1}^{n} x_i$$

```
input the count of the numbers, n
initialize total to 0
loop n times
    input a number, x
    add x to total
output average as total / n
```

Explain the calculation of average value

```
n = int(input("How many numbers do you have? "))
total = 0.0
for i in range(n):
    x = float(input("Enter a number >> "))
    total = total + x
print("\nThe average of the numbers is", total / n)
```

Translate to Python code

3. Loops structures and Booleans, compare to MATLAB

For loop to build list

```
L = []
For item in range(10):
        L.append(item)
```

List from 0 to 9

```
L = [item for item in range(10)]
```

Common used in while coding

```
L = [item for item in range(10) if item > 5]
```

Including condition, List from 6 to 9

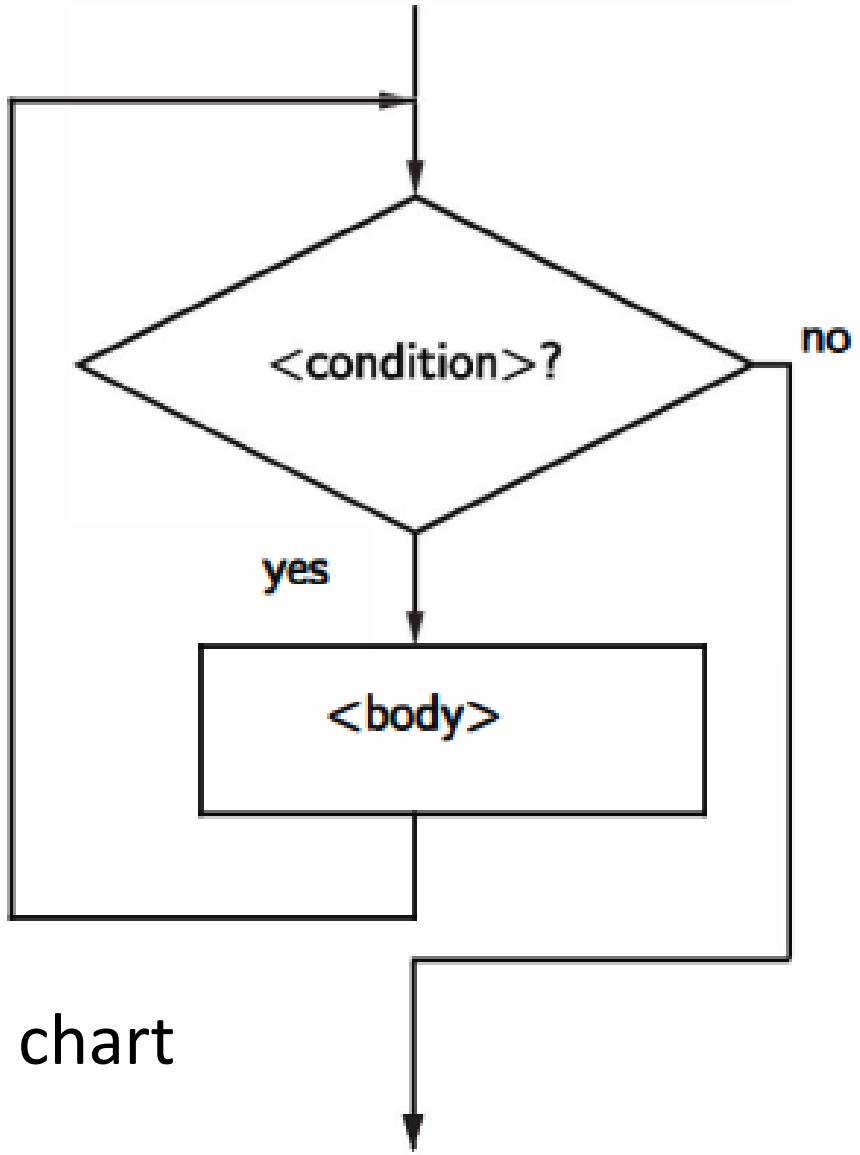
Make code short and simple

3. Loops structures and Booleans, compare to MATLAB

While loop:

```
while <condition>:
        <body>
```

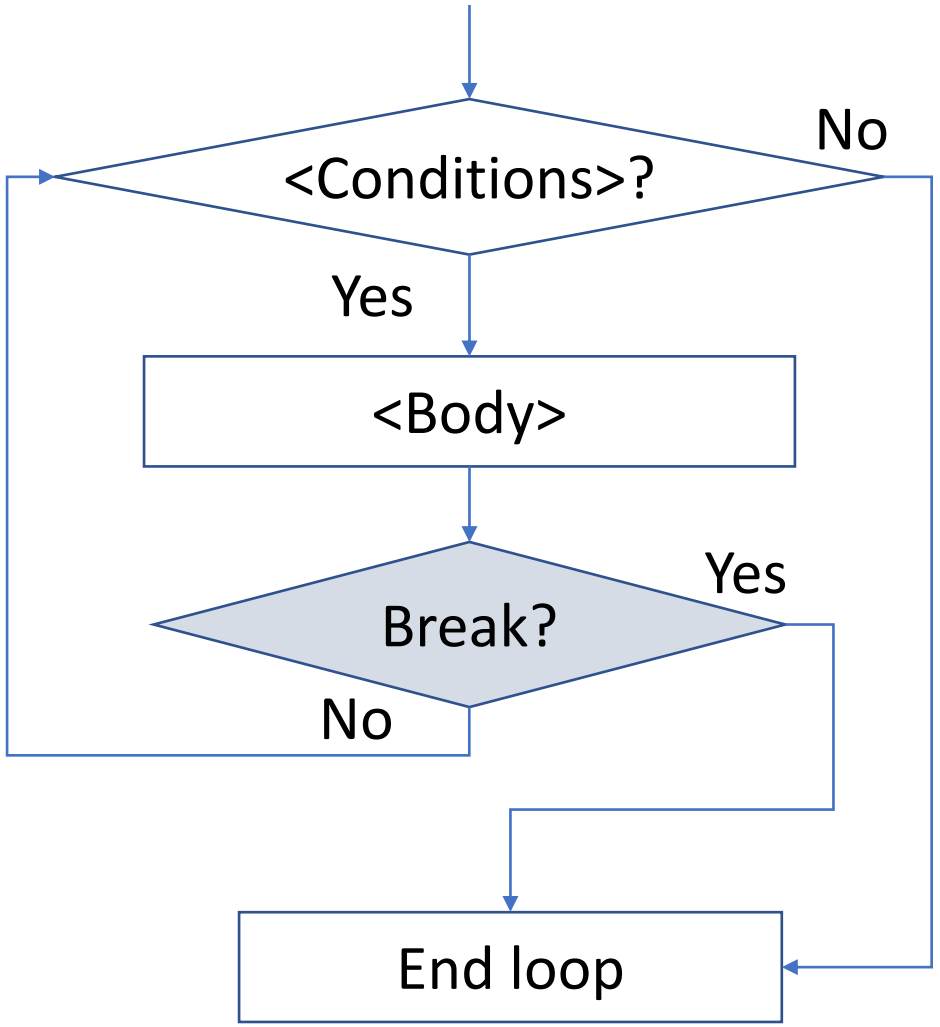
Code format

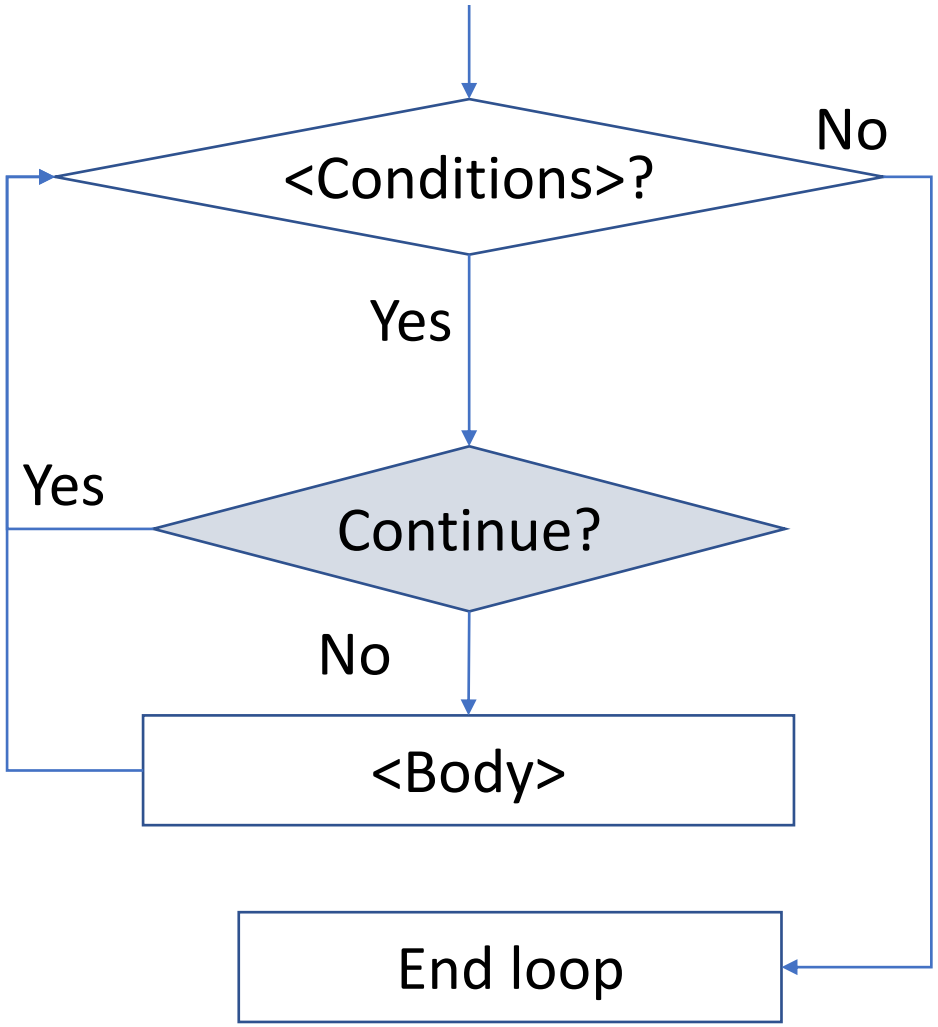


Flow chart

3. Loops structures and Booleans, compare to MATLAB

Break & Continue





Break & Continue

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    break
  print(x)
```



```
apple
```

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    continue
  print(x)
```



```
apple
cherry
```

3. Loops structures and Booleans, compare to MATLAB

For loop – While loop

```
for <var> in <sequence>:
        <body>
```

```
while <condition>:
        <body>
```

Count from 0 to 10

```
for i in range(11):
    print(i)
```

```
i = 0
while i <= 10:
    print(i)
    i = i + 1
```

For loop – While loop: Compare to MATLAB?

Large different in For loop

```
for i in range(11):
    print(i)
```

Python

```
i = 0
while i <= 10:
    print(i)
    i = i + 1
```

```
for i = 0:10
    disp(i)
end
```

MATLAB

```
i = 0
while i <= 10
    disp(i)
    i = i +1;
end
```

Boolean keywords:

- and
- or
- not

algebra	Boolean algebra
$a * 0 = 0$	a and false == false
$a * 1 = a$	a and true == a
$a + 0 = a$	a or false == a

Example

```
while True:
    number = float(input("Enter a positive number: "))
    if number >= 0:
        break # Exit loop if number is valid.
    else:
        print("The number you entered was not positive")
```

3. Write a `while` loop fragment that calculates the following values:

a) Sum of the first n counting numbers: $1 + 2 + 3 + \ldots + n$

HomeWorks

d) The number of times a whole number n can be divided by 2 (using integer division) before reaching 1 (i.e., $log_2 n$).

1. The Fibonacci sequence starts $1, 1, 2, 3, 5, 8, \ldots$. Each number in the sequence (after the first two) is the sum of the previous two. Write a program that computes and outputs the nth Fibonacci number, where n is a value entered by the user.

```
n = 10
x = [1, 1, 2, 3, 5, 8]
for i in range(n -6):
    x.append(x[-1] + x[-2])
print(x)
```

Note:
x[-1] -> **Last** element in the list x
x[:-1] -> **First** to **last-1** th elements

4. The Syracuse (also called "Collatz" or "Hailstone") sequence is generated by starting with a natural number and repeatedly applying the following function until reaching 1:

$$syr(x) = \begin{cases} x/2 & \text{if x is even} \\ 3x+1 & \text{if x is odd} \end{cases}$$

For example, the Syracuse sequence starting with 5 is: $5, 16, 8, 4, 2, 1$. It is an open question in mathematics whether this sequence will always go to 1 for every possible starting value.

Write a program that gets a starting value from the user and then prints the Syracuse sequence for that starting value.

HomeWorks 3b, 3c:
Summary to a report (pdf)

3. Write a `while` loop fragment that calculates the following values:

Practice

b) Sum of the first n odd numbers: $1 + 3 + 5 + \ldots + 2n - 1$

c) Sum of a series of numbers entered by the user until the value 999 is entered. *Note*: 999 should not be part of the sum.

Home work

Practice

HomeWorks 2:
Summary to a report (pdf)

2. The National Weather Service computes the windchill index using the following formula:

$$35.74 + 0.6215T - 35.75(V^{0.16}) + 0.4275T(V^{0.16})$$

Where T is the temperature in degrees Fahrenheit, and V is the wind speed in miles per hour.

Write a program that prints a nicely formatted table of windchill values. Rows should represent wind speed for 0 to 50 in 5-mph increments, and the columns represent temperatures from -20 to +60 in 10-degree increments. *Note*: The formula only applies for wind speeds in excess of 3 miles per hour.

Summary

- Check homeworks
- For loop
- While loop
- Booleans
- Compare to MATLAB
- Practices

Chapter 4
PYTHON

Dr. Minhhuy Le

606-A4, EEE, Phenikaa Uni.

huy.leminh@phenikaa-uni.edu.vn

Volunteer Time!

HomeWorks 3b, 3c:
Summary to a report (pdf)

3. Write a `while` loop fragment that calculates the following values:

Practice

b) Sum of the first n odd numbers: $1 + 3 + 5 + \ldots + 2n - 1$

c) Sum of a series of numbers entered by the user until the value 999 is entered. *Note*: 999 should not be part of the sum.

Home work

Practice

HomeWorks 2:
Summary to a report (pdf)

2. The National Weather Service computes the windchill index using the following formula:

$$35.74 + 0.6215T - 35.75(V^{0.16}) + 0.4275T(V^{0.16})$$

Where T is the temperature in degrees Fahrenheit, and V is the wind speed in miles per hour.

Write a program that prints a nicely formatted table of windchill values. Rows should represent wind speed for 0 to 50 in 5-mph increments, and the columns represent temperatures from -20 to +60 in 10-degree increments. *Note*: The formula only applies for wind speeds in excess of 3 miles per hour.

Outline

Chapter 4

1. Python Programming Development
2. Characters, List, Files
3. Loops structures and Booleans, compare to MATLAB
4. Function and Class
5. Plots

Function:

- Name
- Input arguments
- Output values
- Defined before called >< Matlab: defined at the bottom of the script

```
define function_name(arguments):

        # Body of function

        return output_values
```

Function:

- Name
- Input arguments
- Output values
- Defined before called >< Matlab: defined at the bottom of the script

```
def function_name(arguments):

        # Body of function

        return output_values
```

Function:

- Name
- Input arguments
- Output values
- Defined before called >< Matlab: defined at the bottom of the script

```
def func_sum(a, b):
    c = a + b
    return c
```

Define function: calculate sum of two values

```
s = func_sum(2,3)
```

Call function: s = 2+3 = 5

Default argument

Using the default argument if not input the value to that argument

```
def func_sum(a, b = 2):
    c = a + b
    d = a - b
    return c, d
```

```
s1, m1 = func_sum(2,3)


s2, m2 = func_sum(2)
```

Function returns: sum and minus values
Default argument: b = 2

s1 = 5, m1 = -1
s2= 4, m2 = 0 -> Use default b = 2

Can input argument like this:

```
s1, m1 = func_sum(2, b = 3)
```

Global variable:

Defined inside the function: *global x*

```
x = "awesome"
def myfunc():
  print("Python is " + x)


myfunc()
print(x)
```

Python is awesome
awesome

```
x = "awesome"
def myfunc():
  x = "fantastic"
  print("Python is " + x)

myfunc()
print(x)
```

Python is fantastic
awesome

```
x = "awesome"
def myfunc():
  global x
  x = "fantastic"

myfunc()
print(x)
```

fantastic

Practice: Write a function

5.31 Gravitational Force The gravitational force F between two bodies of masses m_1 and m_2 is given by the equation

$$F = \frac{Gm_1m_2}{r^2} \tag{5-23}$$

where G is the gravitation constant (6.672×10^{-11} N m^2 / kg^2), m_1 and m_2 are the masses of the bodies in kilograms, and r is the distance between the two bodies. Write a function to calculate the gravitational force between two bodies given their masses and the distance between them. Test you function by determining the force on an 800 kg satellite in orbit 38,000 km above the Earth. (The mass of the Earth is 5.98×10^{24} kg.)

Practice: Write a function

2. The National Weather Service computes the windchill index using the following formula:

$$35.74 + 0.6215T - 35.75(V^{0.16}) + 0.4275T(V^{0.16})$$

Where T is the temperature in degrees Fahrenheit, and V is the wind speed in miles per hour.

Write a program that prints a nicely formatted table of windchill values. Rows should represent wind speed for 0 to 50 in 5-mph increments, and the columns represent temperatures from -20 to +60 in 10-degree increments. *Note*: The formula only applies for wind speeds in excess of 3 miles per hour.

Classes and Objects

- Python is an object oriented programming language.
- Almost everything in Python is an object, with its properties and methods.
- A Class is like an object constructor, or a "blueprint" for creating objects.

```
class MyClass:
  x = 5
```

Create a class

```
p1 = MyClass()
print(p1.x)
```

Using the class

Classes: __init__() function

Set initial properties value of the class when it created

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

p1 = Person("John", 36)

print(p1.name)
print(p1.age)
```

__init__ -> Initial function
self -> Refer to the Person
self.name -> Person.name
self.age -> Person.age

Create object p1 with class Person including initial value

Create class Person with initial values

Classes: methods

Methods are functions inside the class

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def myfunc(self):
    print("Hello my name is " + self.name)

p1 = Person("John", 36)
p1.myfunc()
```

Create a method for class Person

Classes: Modify object properties

Assign values to properties or delete it

```
p1.age = 40
```


Set age = 40

```
del p1.age
```


Delete property age of the p1 object

```
del p1
```


Delete object p1

3. Show the output that would result from the following nonsense program:

```
class Bozo:

    def __init__(self, value):
        print("Creating a Bozo from:", value)
        self.value = 2 * value

    def clown(self, x):
        print("Clowning:", x)
        print(x * self.value)
        return x + self.value

def main():
    print("Clowning around now.")
    c1 = Bozo(3)
    c2 = Bozo(4)
    print c1.clown(3)
    print c2.clown(c1.clown(2))

main()
```

9. Write a class to represent spheres. Your class should implement the following methods:

`__init__(self, radius)` Creates a sphere having the given `radius`.

`getRadius(self)` Returns the radius of this sphere.

`surfaceArea(self)` Returns the surface area of the sphere.

`volume(self)` Returns the volume of the sphere.

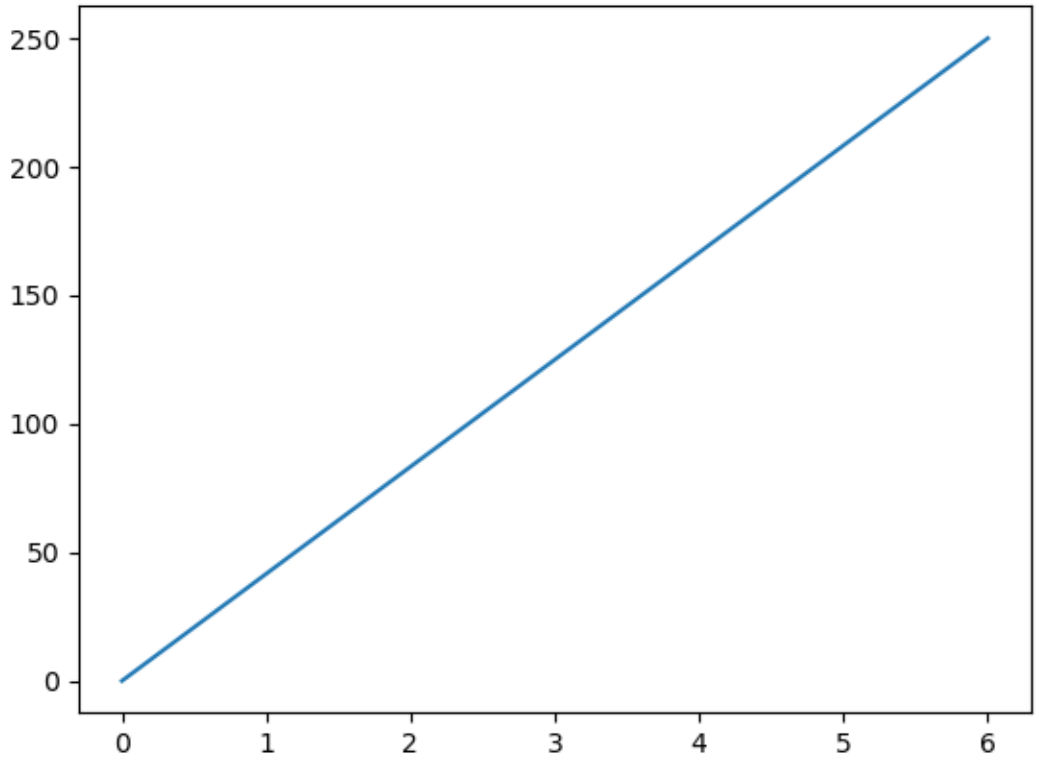
Matplotlib library

Import plot library as plt
Use “plt” as a class

```
import matplotlib.pyplot as plt
import numpy as np


xpoints = np.array([0, 6])
ypoints = np.array([0, 250])

plt.plot(xpoints, ypoints)
plt.show()
```

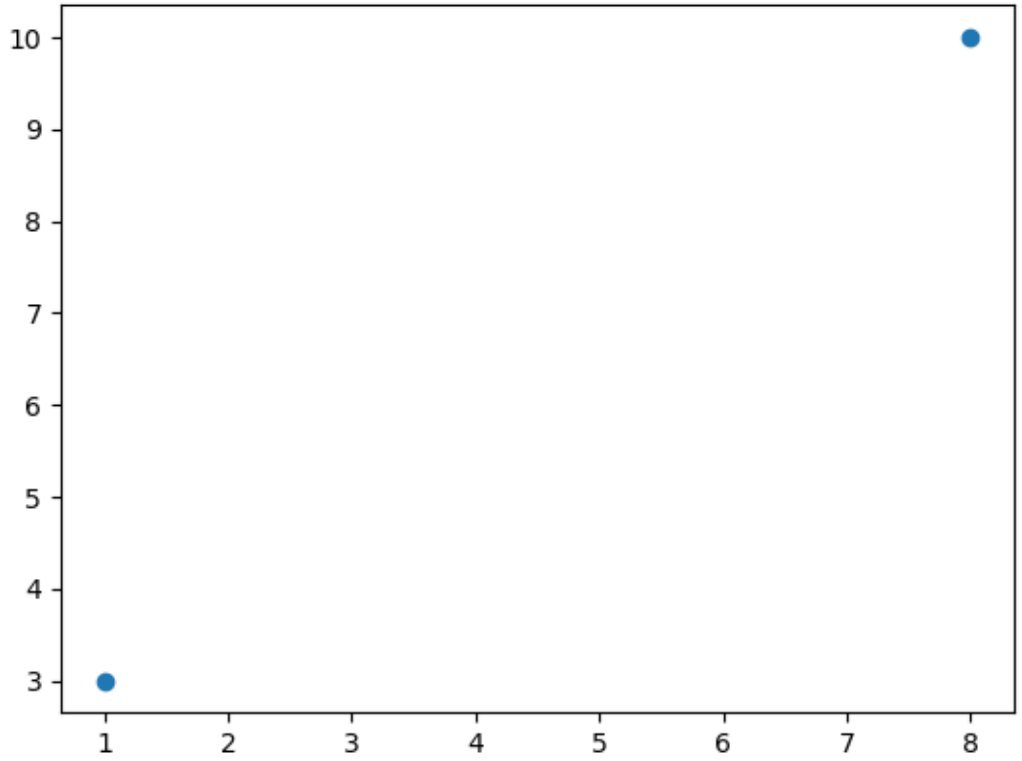


Plot styles

```
import matplotlib.pyplot as plt
import numpy as np

xpoints = np.array([1, 8])
ypoints = np.array([3, 10])

plt.plot(xpoints, ypoints, 'o')
plt.show()
```



Marker
Linestyle
Linewidth
....

Import plot library as plt
Use “plt” as a class

https://matplotlib.org/tutorials/introductory/pyplot.html

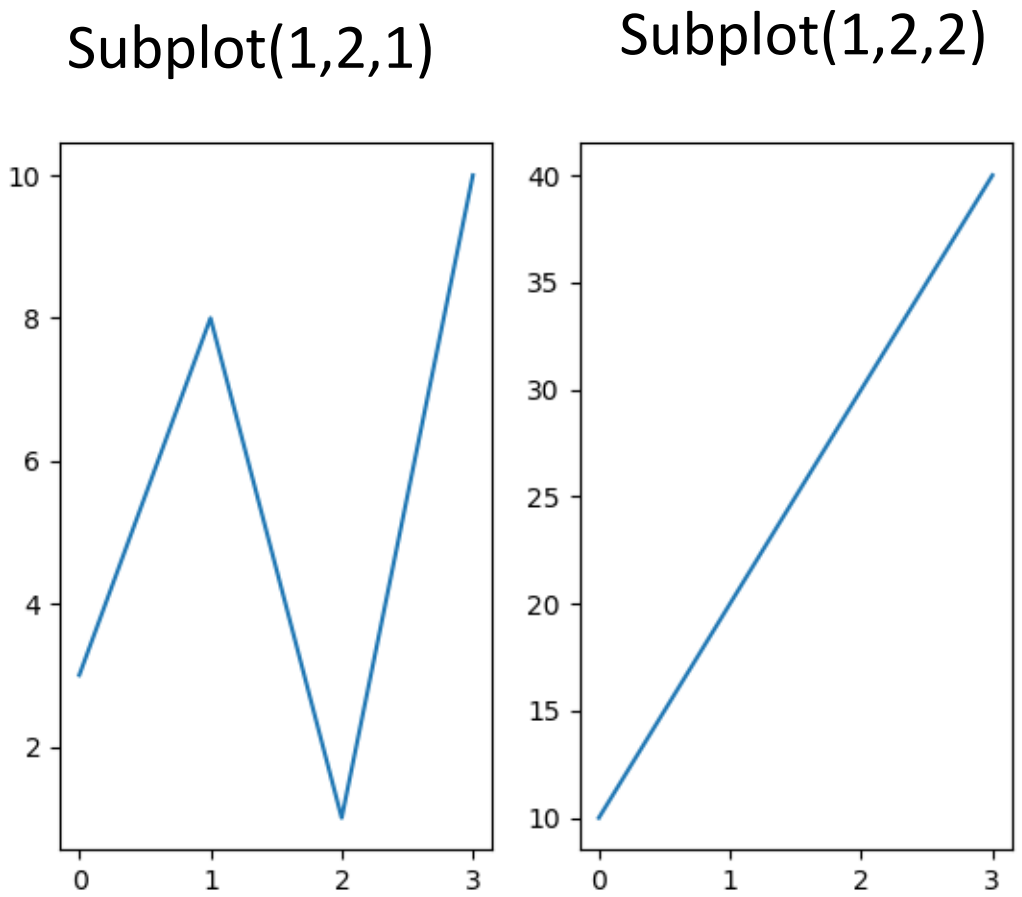
Subplots

```
import matplotlib.pyplot as plt
import numpy as np

#plot 1:
x = np.array([0, 1, 2, 3])
y = np.array([3, 8, 1, 10])
plt.subplot(1, 2, 1)
plt.plot(x,y)

#plot 2:
x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])
plt.subplot(1, 2, 2)
plt.plot(x,y)

plt.show()
```



Scatter plots

scatter() function plots one dote for each data

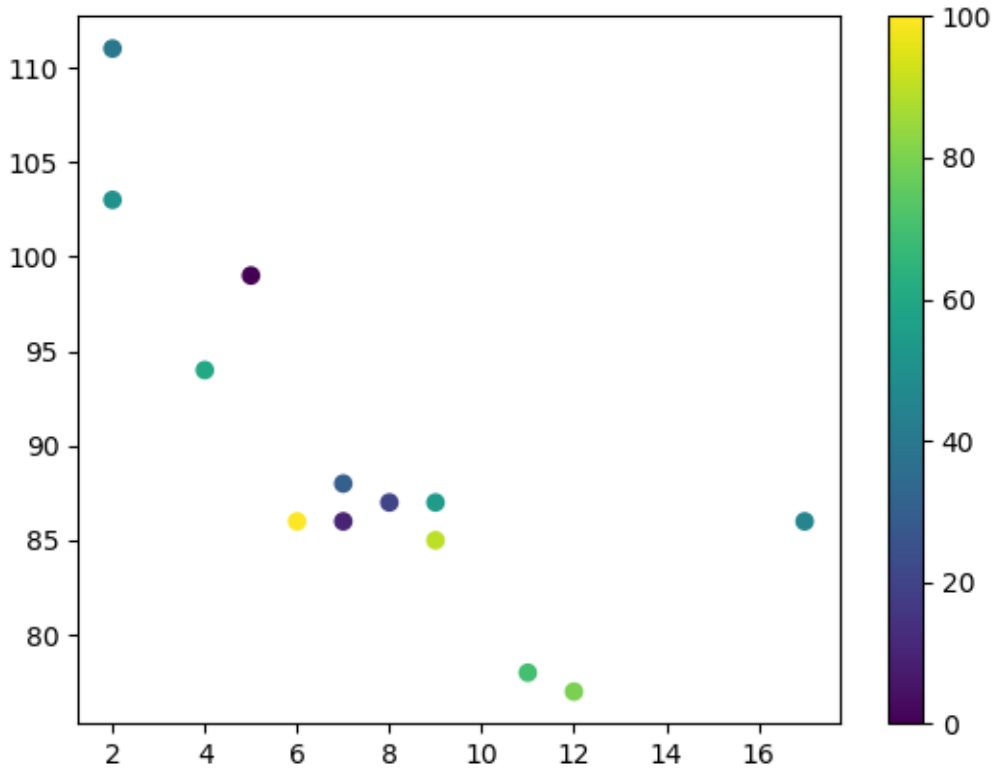
```
import matplotlib.pyplot as plt
import numpy as np

x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])
y =
np.array([99,86,87,88,111,86,103,87,94,78,77,85
,86])
colors = np.array([0, 10, 20, 30, 40, 45, 50, 55, 60,
70, 80, 90, 100])

plt.scatter(x, y, c=colors, cmap='viridis')

plt.colorbar()

plt.show()
```

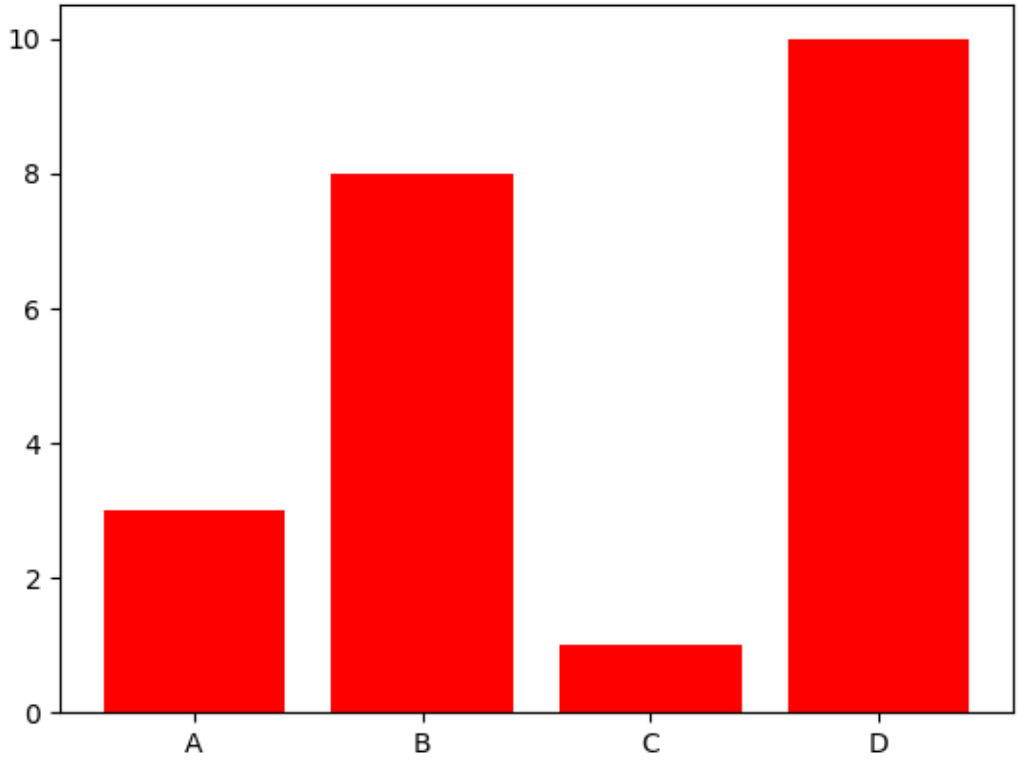


Bar plots

```
import matplotlib.pyplot as plt
import numpy as np

x = np.array(["A", "B", "C", "D"])
y = np.array([3, 8, 1, 10])

plt.bar(x, y, color = "red")
plt.show()
```



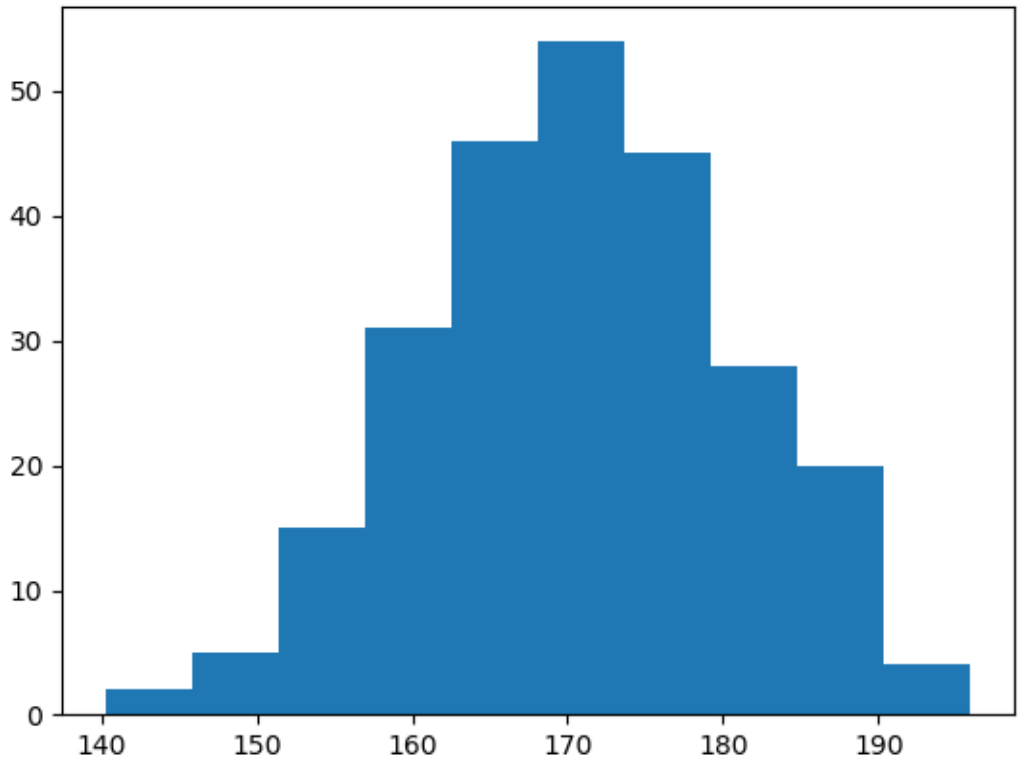
Histogram plots

It is a graph showing the number of observations within each given interval

```
import matplotlib.pyplot as plt
import numpy as np

x = np.random.normal(170, 10, 250)

plt.hist(x)
plt.show()
```

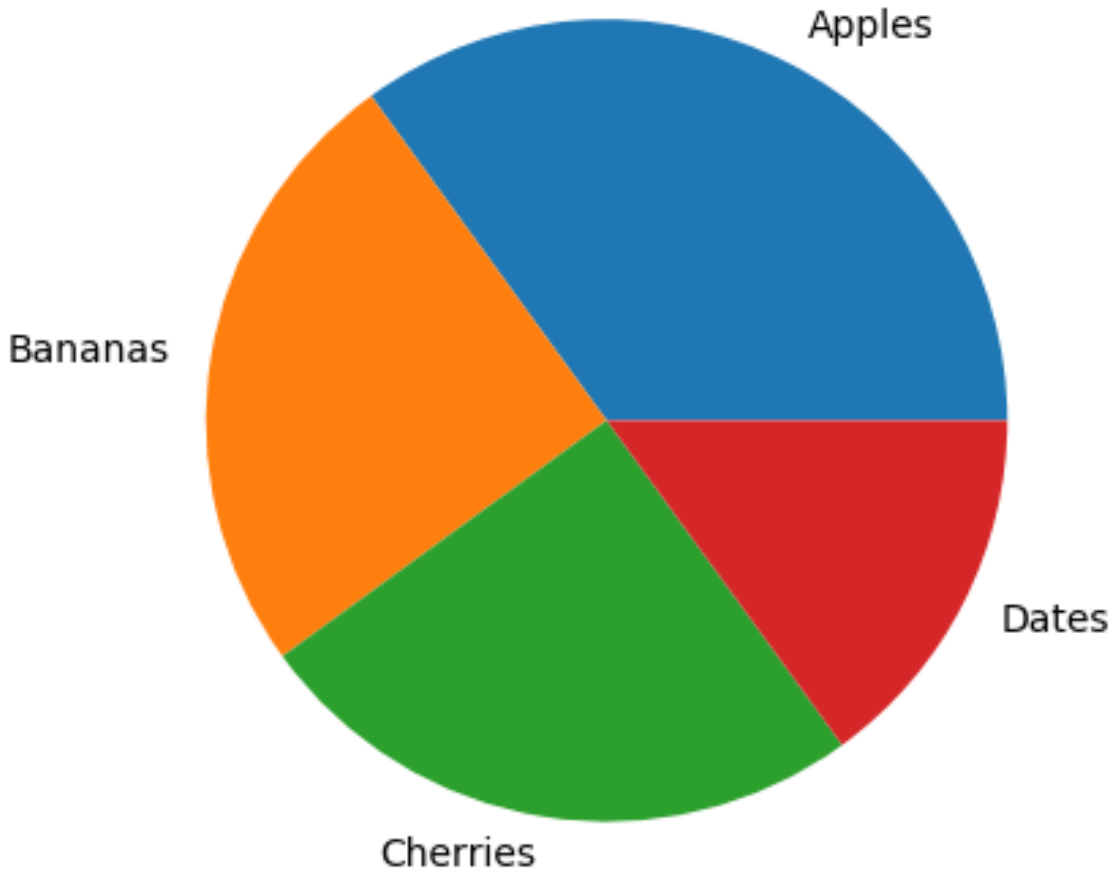


Pie chart

```
import matplotlib.pyplot as plt
import numpy as np

y = np.array([35, 25, 25, 15])
mylabels = ["Apples", "Bananas", "Cherries",
"Dates"]

plt.pie(y, lables = mylabels)
plt.show()
```



HomeWorks 3b, 3c:
Write a function for each sum.
Write a class with methods are the two functions
Summary to a report (pdf)

3. Write a `while` loop fragment that calculates the following values:

Practice

b) Sum of the first n odd numbers: $1 + 3 + 5 + \ldots + 2n - 1$

c) Sum of a series of numbers entered by the user until the value 999 is entered. *Note*: 999 should not be part of the sum.

Home work

Practice

Summary

- Check homeworks
- Function
- Global variables
- Class
- Plots
- Practices