DSC 5101 – Computer Programming in DSAI
Part I - Python Programming

HO 01 - Python Programming Basics

Opim Salim Sitompul

Department of Data Science and Artificial Intelligence
Universitas Sumatera Utara

Co-funded by the Erasmus+ Programme of the European Union

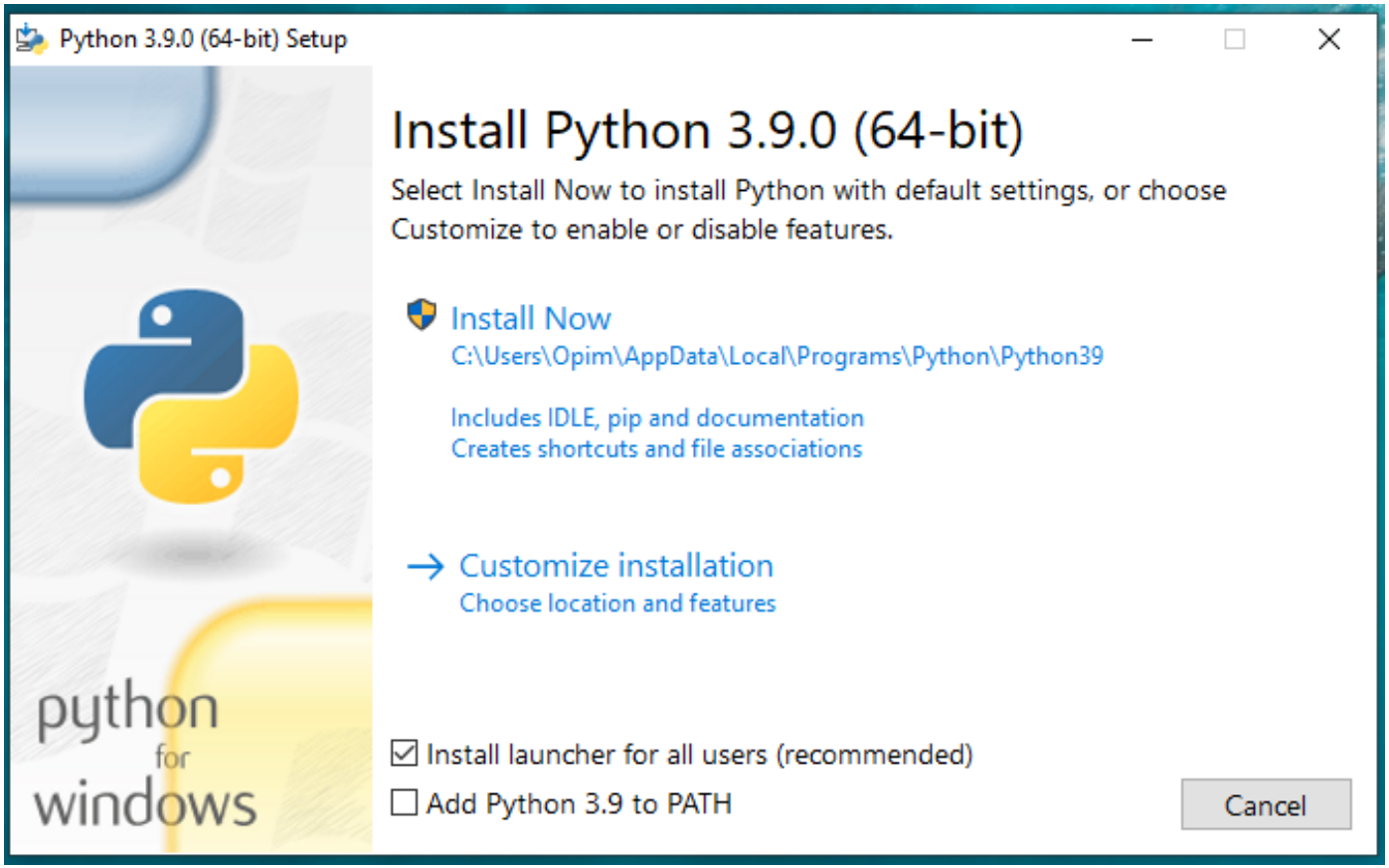
Outline I

Introduction to Python

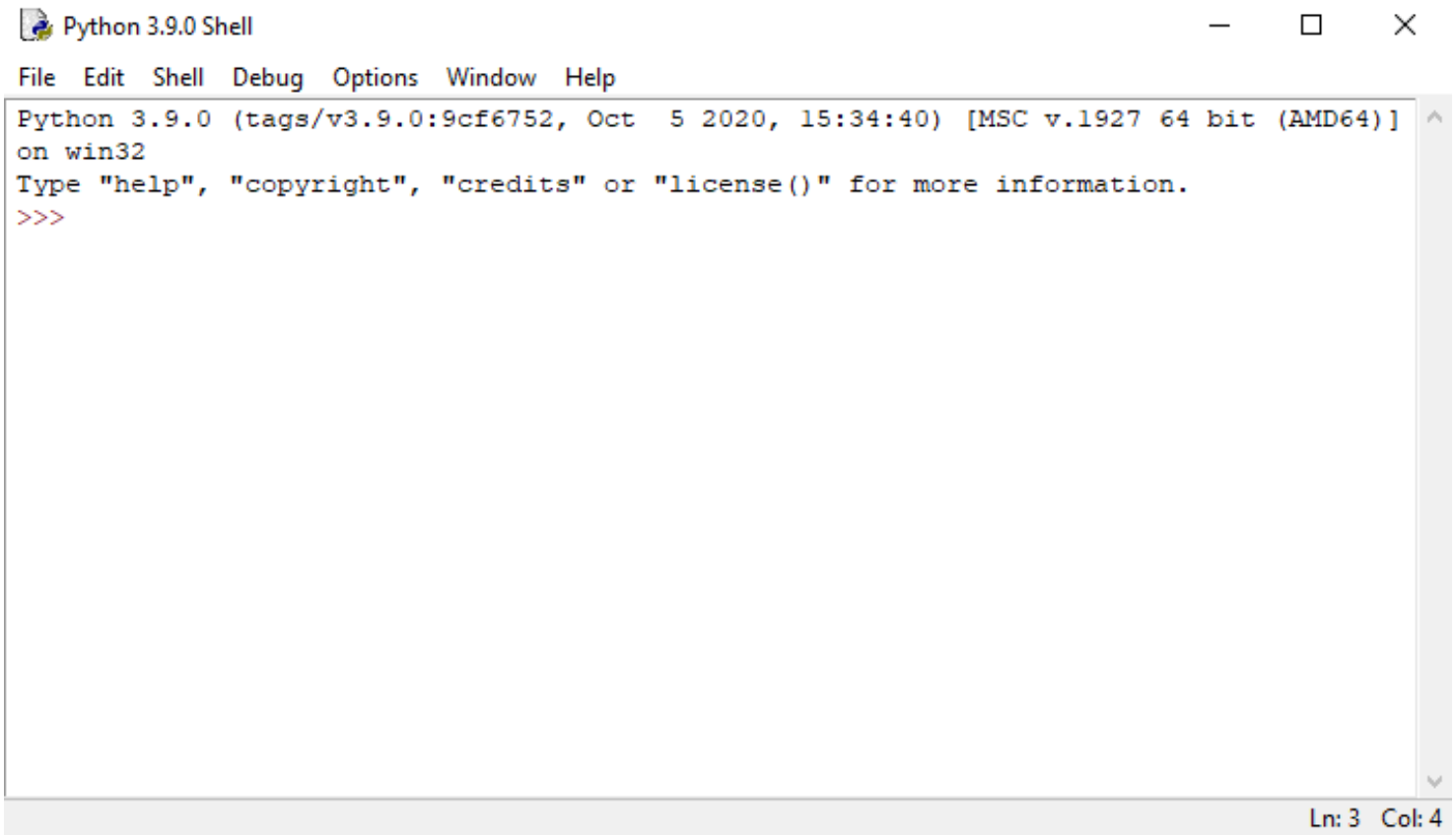
- Python is a first-class tool for scientific computing tasks [1].
- Before we can use Python, we have to consider several options in the installation choices.
  1. Python IDLE
  2. Anaconda
  3. Pycharm
- In this course we will be using Python 3, latest version is Python 3.9.0 (Release Date: Oct. 5, 2020).

Preparing Python using IDLE

- Python IDLE dapat di downloaad pada URL berikut: https://www.python.org/downloads/release/python-390/
- Namafile: python-3.9.0-amd64.exe



Preparing Python using IDLE



Preparing Python Anaconda

- To install Python and the suites of libraries for scientific computing, we can choose one of two Anaconda distributions:
  1. Anaconda (https://www.anaconda.com/products/individual)
  2. Miniconda (https://docs.conda.io/en/latest/miniconda.html)

Preparing Python Anaconda

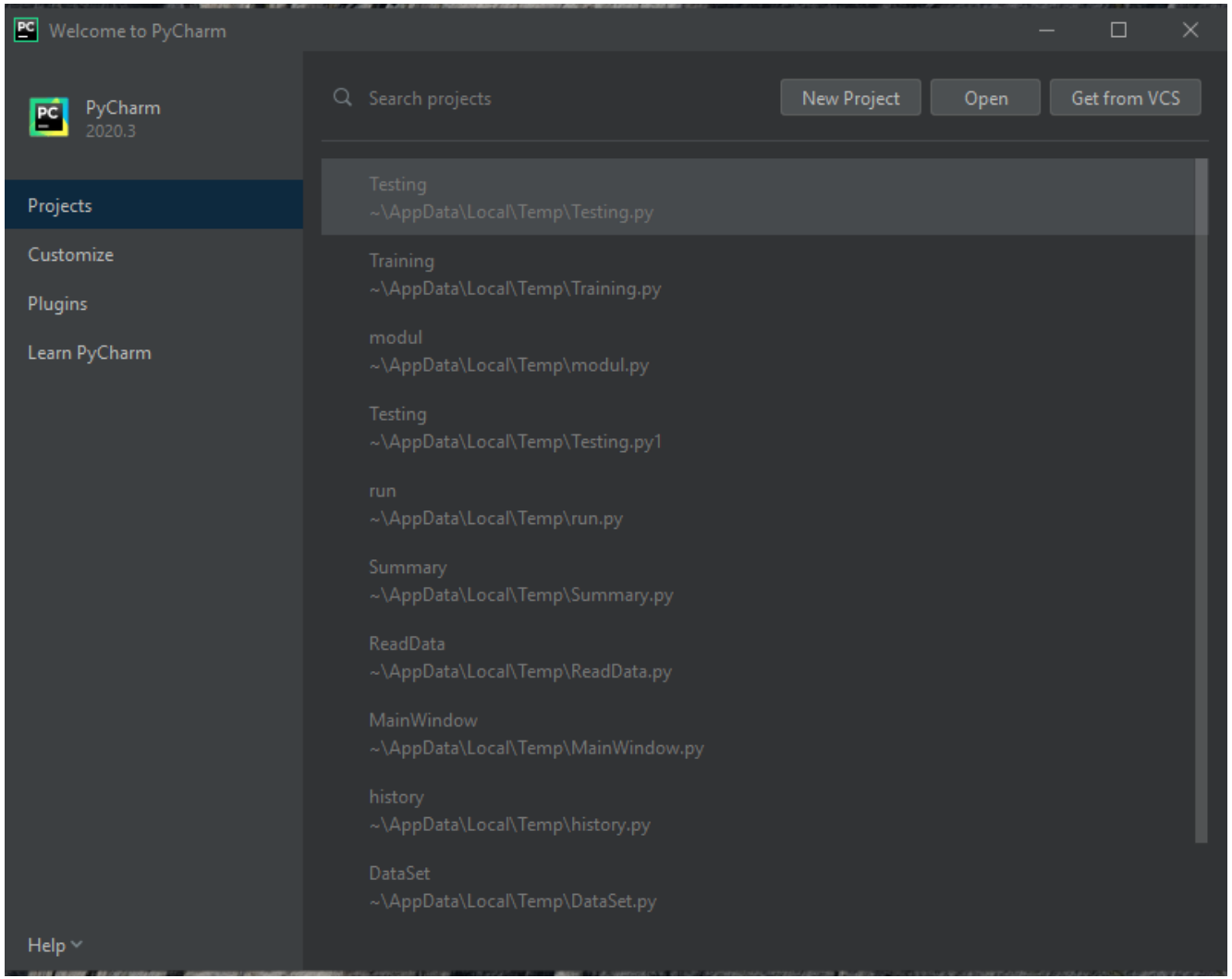
- Choose Anaconda, if you
  - Are new to conda or Python.
  - Like the convenience of having Python and over 1,500 scientific packages automatically installed at once.
  - Have the time and disk space—a few minutes and 3 GB.
  - Do not want to individually install each of the packages you want to use.

Preparing Python Anaconda

- Choose Miniconda, if you
  - Do not mind installing each of the packages you want to use individually.
  - Do not have time or disk space to install over 1,500 packages at once.
  - Want fast access to Python and the conda commands and you wish to sort out the other programs later.

Preparing Python PyCharm

Preparing Python PyCharm



Python Variable Names Convention

- There are some naming conventions when developing a Python program:
  - use all lowercase,
  - use descriptive names rather than variable names such as *a* or *b* (except for indexing purposes such as the use of variables *i* and *j* in looping constructs).
  - seperate individual words by underscores as necessary to improve readability.

Python Variable Names Convention

- Examples:
  - my_name, your_name, user_name, account_name
  - count, total_number_of_users, percentage_passed, pass_rate
  - where_we_live, house_number,
  - is_okay, is_correct, status_flag

Comments in Code

- Comments to code are to help anyone reading the code to understand what the code does,
  - what its intent was,
  - any design decisions the programmer made, etc,
- A comment is indicated by the '#' character.
- Anything following that character to the end of the line will be ignored by the interpreter.

Comments in Code

- Example:
  - # Get an input
    number = input("Please give an integer:")
    # Check whether the input given is an integer
    input_ok = check_input(number)

Python String

- A string is a series, or sequence, of any combination of characters in order.
- In general a character is represented using ASCII (American Standard Code for Information Interchange) code.
- In ASCII, characters are divided into three categories:
  - Non-printing characters, within the range of 0 to 31 of ASCII codes.
  - Printing characters, within the range of 32 to 127 of ASCII codes.
  - Special (graphics) characters, within the range of 128 to 255 of ASCII codes.

Python String

- In addition to standard ASCII codes, Python also recognize other character formats, such as the Unicode (Universal Coded) Character Set Standard.
- Example: UTF-8 (Unicode Transformation Format – 8-bit).
  - A variable-width character encoding used for electronic communication.
  - UTF-8 consists of 1,112,064 valid character code points in Unicode using one to four one- byte (8-bit) code units encoding.
- The following Python code prints $\Lambda\tau\eta\epsilon\nu\alpha$
  print('\u039b\u03c4\u03b7\u03b5\u03bd\u03b1')

Representing String

- String can be represented by single quotes or double quotes to define the start and end of a string by the same quote, but could not be mixed.
- Example:

  ”Today is Python’s day”

Type of String

- As a dynamically typed language, each data has a certain type.
- An operation on data could only be performed within the same type.
- Example:

  today = ‘Monday’
  five_day_later = today + 5
- Will produce the following error:
  “TypeError: can only concatenate str (not "int") to str”

Type of String

- In order to assess type of a certain data, we could use a built-in function *type*.
- Applying type to variable name *today*, will give the type of the string.

  print(type(today))

  <class 'str'>
- Saying that the variable *today* has a type of 'class <str>'. While the type of constant *5* has a type of:

  print(type(5))

  <class 'int'>
- which is not the same type.

Operation on String

- The class type <str>, has many attributes that could be accessed in order to further get information oh that string.
  - String Concatination:
    today = 'Monday'
    five_day_later = today + ' + ' + str(5)
    print(five_day_later) → **Monday + 5**
  - Length of string:
    print(len(five_day_later)) → **10**
  - Accessing characters:
    print(five_day_later[0]) → **M**
    print(five_day_later[-1]) → **5**

Operation on String

- Accessing subset of characters
  print(five_day_later[0:6]) → **Monday**
- Repeating characters
  print(five_day_later[0:6]*2) → **MondayMonday**
- Splitting string
  print(five_day_later.split('+')) → **['Monday ', ' 5']**
- Counting string
  print(five_day_later.cont('+')) → **1**
- Replacing string
  print(five_day_later.replace('+', '-')) → **Monday - 5**
- Finding string
  print(five_day_later.find('+')) → **7**
- Comparing string
  print(five_day_later == five_day_later) → **True**

Operation on String

- Other methods in <str> class.

Function	Task
startswith()	Check if string starts with certain character
endswith()	Check if string ends with certain character
istitle()	Check if string is in title format
isupper()	Check if upper case
islower()	Check if lower case
isalpha()	Check if alphabet
upper()	Convert to upper case
lower()	Convert to lower case
title()	Convert to title
swapcase()	Revert case
strip()	remove leading trailing spaces

References I

[1] VanderPlas J., Python Data Science Handbook: Essential Tools for Working with Data, O'Reilly, USA, 2016.

[2] Hunt, J., A Beginners Guide to Python 3 Programming, Springer Nature Switzerland AG, Switzerland, 2019.

[3] Hunt, J., Advance Guide to Python 3 Programming, Springer Nature Switzerland AG, Switzerland, 2019.