Tools in this course

- Anaconda Python distribution
  - Python
  - Jupyter Notebook

Jupyter overview

- Notebook: code + text
- Server-client app to edit and run notebooks
- Via a web browser
- Without internet access

Comments essentials

- Put hints
- Describe code
- Readable and easy
- Not executed
- #Comment
- Single line comment
- Multiple line comment

Indentation essentials

- White spaces
- Recognize blocks
- 4 whitespaces inserted by default
- Inappropriate indentation: IndentationError

Variables foundations

- Memory location to store different data
- No command for variable declaration
- No type placed
- Assignment statement
- variableName = value
- language = “Python”

Variable Name

- Contain letters or _
- Can’t start with numbers
- No spaces
- Can’t use Python keywords
- Case sensitive
- Descriptive and readable
- camelCaseMethod

Inputs foundations

- Function: block of related statement
- input(arg): ask user for input
- Arg: value passed to function
- The input(): converts input to string

List foundations

- Sequence of values
- Ordered sequence: zero based index
- Mutable items
- Allow duplicates
- [value1, value2, value3]

Tuple foundations

- Sequence of values
- Immutable items but you can update the tuple
- Ordered items: zero based index
- Duplicates allowed
- (value1, value2, value3)

Dictionary foundations

- Collection of values
- Item → {key: value}
- No duplicates
- {“language”: “Python”, “version”: 3.9.4}

Set foundations

- Unordered collection
- Immutable item
- Remove duplicates
- No indexing

Operators essentials

- Arithmetic: +, -, *, /, %, //, **
- Comparison: >, <, >=, <=, ==, !=
- Logical: and, or, not
- Bitwise: &, |, ~, ^, >>, <<
- Assignment: =, +=, -=, *=, /=, %=, //=, **=
- Identity: is, is not
- Membership: in, not in

Conditional statements

- Make decisions based on condition
- Decide the direction of flow
- The if conditional statement → if condition:
- The elif statement → elif condition:
- The else statement → else:
- Indented body

The while loop

- Execute block repeatedly
- Based on test expression
- Indented body
- while testExp:
- Control the loop → break, continue, pass

The for loop

- Execute block repeatedly
- Iterate over sequences
- Indented body
- for var in seq:
- Control the loop: → break, continue, pass

Built-in Functions

- Pre-designed function from Python Library
- Call the function and Insert arguments
- functionName(arg)
- Examples: abs(), bool(), callable(),complex(), dict(), dir(), enumerate(), float(), format(), frozenset(), id(), input(), int(), iter(), len(), list(), max(), min(), next(), open(), set(), str(), sum(), tuple(), type(), zip()

User-Defined Functions

- Group of related statements
- Organize, reuse code, and save time
- The def keyword followed by the name
- Indented body(4 whitespaces by default)
- Parameters and Arguments
- Calling a function → functionName(arg)
- Return statement

Anonymous Lambda

- Function without name
- Lambda keyword
- Any number of arguments
- Only a single expression
- lambda arguments: expression
- Return function objects

Python OOP

- Class: Data Type and Blueprint for objects
- Object: an Instance of class
- Class has Indented body
- The class keyword

Python Comprehensions

- Short way to create sequence
- Create sequence from sequence

Python with Files

- Text files vs Binary files
- open('filename.ext', 'mode')
- Reading, Writing and Deleting files

Python io Module

- Manage files related to inputs and outputs operations
- Implement an in-memory file like object

Python os Module

- Provides functions for interacting with the operating system
- Handling directories

Python shutil Module

- Move files to different directories

send2trash Module

- Send files to the trash or Recycle Bin
- Delete files or folders

zipfile Library

- Compress and extract files and folders

NumPy Array

- Used data processing libraries
- Create n-dimensional array
- Similar to lists
- NumPy is faster than lists while working with vectors
- Slower than lists when you add items to the end
- Homogeneous

Pandas Library

- Manipulate numerical data and time series
- Built on NumPy
- Import and analyze data easily
- Fast, high performance and high productivity

Matplotlib Library

- Visualization library for 2D plots
- Built on NumPy
- Several Plots like line, bar, scatter, histogram and more

Seaborn Library

- Visualization library for statistical graphics plotting
- Built on matplotlib
- Oriented APIs to explore and understand data
- Seaborn plots: relational, categorical, distribution, regression, matrix and multi-plot grids