Computer Programming Lab 5

2024/03/26 Tim Chen

Contents

Module

Module

Definition:

- Files containing Python code.
- They can define functions, classes, variables.

Purpose:

- Organize Python code into reusable units.
- Enhance code readability and maintainability.
- Encapsulate related functionalities.

Module (cont.)

Example:

```python
# lab05_module.py
def greeting(name: str):
  print(f"Hello, {name}")
```

Module (cont.)

Using the module.

```
# lab05.py
import lab05_module

if __name__ == '__main__':
  lab05_module.greeting('Alice')
```

Module (cont.)

Using the module with `from` keyword.

```
# lab05.py
from lab05_module import greeting

if __name__ == '__main__':
  greeting('Alice')
```

JSON

JSON

```
{
  "input": [
    "This is a sentence.", "This is another sentence."
  ],
  "author": "GNITOAHC",
  "scripts": {
    "dev": "next dev",
    "build": "next build"
  }
}
```

JSON Module

JSON Module

```
json_raw = """
{
  "input": [
    "This is a sentence.", "This is another sentence." ],
  "author": "GNITOAHC",
}
"""
```

JSON Module (cont.)

```
import json

loaded_json  = json.loads(json_raw)
author = loaded_json["author"] # Get the value inside JSON
print(author) # Output: GNITOAHC
```

Notes: Python dictionaries can be directly converted to JSON format and vice versa.

Python `isinstance()`

Python `isinstance()`

Definition:

- The `isinstance()` function is used to check if an object is an instance of a specified class or any of its subclasses.

Purpose:

- Verify the type of an object dynamically.
- Enables conditional logic based on the type of an object.

Python isinstance() (cont.)

Example:

```
# Syntax: isinstance(object, classinfo)
x = 5
print(isinstance(x, int))   # Output: True
print(isinstance(x, str))   # Output: False
print(isinstance(x, (str, int, float))) # Output: True
```

Python `split()`

Python split()

Definition:

- Split a string into a list of substrings based on a specified delimiter.
- Delimiter can be a character, a string, or None.

Purpose:

- Tokenize strings.
- Extract meaningful data from text.
- Parse input data.

Python split() (cont.)

Example:

```
x = "This is a sentence."
print(x.split()) # ['This', 'is', 'a', 'sentence.']
```

Key points:

- The split() function is used to split a string into a list of substrings based on a delimiter.
- It returns a list of substrings.

Typing & Union

Typing

```python
x: int = 10

if isinstance(x, int):
  print("x is int")
elif isinstance(x, str):
  print("x is str")
```

Union

Definition:

- The `Union` type allows specifying that a variable or parameter can accept values of multiple types.
- It is used when a function or method can accept different types of input.

Union (cont.)

```python
from typing import Union

def process_data(data: Union[list, tuple, dict]) -> None:
  if isinstance(data, list):
    print("Processing list...")
  elif isinstance(data, tuple):
    print("Processing tuple...")
  elif isinstance(data, dict):
    print("Processing dictionary...")
```

Union (cont.)

Recap the JSON:

```
{
  "input": [
    "This is a sentence.", "This is another sentence.",
    [ "This is nested sentence",
      [ "Double nested sentence." ] ],
  ],
}
```

Assignment 4

Assignment 4

Instruction:

- Write a **recursive function** to analyze a nested list of text, **counting the total number of words** and the **occurrence of a specific word**. The function should return these metrics as a tuple.

Assignment 4 (cont.)

Input Format

- A line as JSON format and a line of target word.

Output Format:

- The total number of words in the text
- The number of occurrences of the target word

Assignment 4 (cont.)

Input sample:

```
{ "input": [ "A sentence.", ["A nested list."]] }
nested
```

Output sample:

```
Total Words: 5, 'nested' Occurrences: 1
```

Assignment 4 (cont.)

Notes:

- The data you're going to process is the value of "input". Remember to extract it after loading.
- Match the words using `split()` function. You **do not** need to remove the leading or trailing punctuation marks. e.g. target word "list" won't match "list."
- Nested list will occur in the test case.
- The return type should be `def func() -> tuple[int, int]: ...`

Assignment 4 (cont.)

Tips:

- Use `nested_text: Union[str, list] = json.loads ...`
- Define a function called:

```
analyze_text(text: Union[str, list], target_word)
```

- Format the output with:
  (Remember to put in the corresponding value)

```
print(f"Total Words: {}, '{}' Occurrences: {}")
```

Assignment 4 (cont.)

Tips:

- You can add your `nest_text` value as the following to debug.

```
nested_text: Union[str, list] = [
  "This is a sentence.",
  ["This is a nested list", "with two sentences."],
  "This is another sentence containing the word.",
]
```

- Output: `Total Words: 19, 'sentence' Occurrences: 1`

Appendix

Appendix

TA Hour:

- Thur. 16:30 ~ 17:30 大仁 102

Contact Me:

- GitHub - https://github.com/gnitoahc/
- Email - chaotingchen10@gmail.com
- Website - https://chaoting.xyz/

Thank you