Lesson 6: Working With Text, Files, and Python Modules

Marcin Kurzyna

Lecture Goals

This lecture introduces:

- Advanced string operations in Python.
- Reading from and writing to files.
- Python's module system and common built-in libraries.
- Creating your own simple module.

1 Working With Text

Strings are one of the most important data types in Python. Python provides many built-in methods for manipulating text.

Common String Operations

- Changing case: `upper()`, `lower()`.
- Checking content: `startswith()`, `endswith()`, `isdigit()`.
- Splitting and joining: `split()` and `' '.join(list)`.
- Stripping whitespace: `strip()`.

Example Code

```
text = "  Hello, Python World!  "

print(text.upper())
print(text.strip())
words = text.split()
print(words)
joined = "-".join(words)
print(joined)
```

2 Working With Files

Python allows you to easily read and write files using the `open()` function.

Opening Files

- `"r"` – read-only
- `"w"` – write (overwrites file)
- `"a"` – append

Writing to a File

```
with open("output.txt", "w") as f:
    f.write("Hello from Python!\n")
    f.write("This is a file test.")
```

Reading From a File

```
with open("output.txt", "r") as f:
    content = f.read()

print("File contains:")
print(content)
```

Reading Line by Line

```
with open("output.txt", "r") as f:
    for line in f:
        print("Line:", line.strip())
```

3 Modules and Libraries

Modules allow you to organize code into reusable files. Python provides many built-in modules you can import using `import`.

Import Examples

- `import math`
- `import random`
- `from datetime import date`

Commonly Used Modules

- **math**: mathematical functions.
- **random**: generating random numbers.
- **datetime**: working with dates and times.

Example: Using math and random

```
import math
import random

print(math.sqrt(25))
print(random.randint(1, 10))
```

4 Creating Your Own Module

A module is simply a `.py` file that contains Python code.

Example Structure

```
# mymodule.py
def greet(name):
    return f"Hello, {name}!"

def add(a, b):
    return a + b
```

Using Your Module

```
import mymodule

print(mymodule.greet("Alice"))
print(mymodule.add(3, 7))
```

Summary

In this lecture, we covered:

- How to manipulate strings using Python methods.
- How to read and write files using the `with open()` syntax.
- How to import and use Python's built-in modules.
- How to create your own module and import it.

5 Exercises

1. Write a program that reads a text file and counts how many lines, words, and characters it contains.
2. Write a program that asks the user for a sentence and saves it to a file called `user_input.txt`.
3. Create a module `texttools.py` with two functions:
   - `count_vowels(s)` – returns the number of vowels in a string.
   - `reverse(s)` – returns the reversed string.
4. In a separate script, import `texttools` and test both functions.
5. Using `random`, write a program that generates a password of 12 random characters.
6. Write a program that reads all lines of a file and prints them in reverse order.
7. Use the `datetime` module to print the current date in the format `YYYY-MM-DD`.
8. Challenge: Write a script that copies the contents of one file to another **without using** `shutil` or external libraries.