Module 2: Deep Dive into Python

Input Output Functions:

Input Functions:

- *input() function is used to take user input.*
- *It returns a string, so type conversion may be necessary for numerical input.*

```
name = input("Enter your name: ")

age = int(input("Enter your age: "))
```

Output Functions:

- *print() function is used for standard output.*
- *You can format output using f-strings or the format() method.*

```
print(f"Name: {name}, Age: {age}")
```

Loops, Lists, Dictionaries, Tuples:

Loops:

- *for loop is used for iterating over a sequence (list, tuple, string, etc.).*
- *while loop continues as long as a certain condition is true.*

for Loop:

The for loop is used for iterating over a sequence (list, tuple, string, etc.). It iterates through each element in the sequence.

```
for i in range(5):

    print(i)
```

for Statement:

The for loop is initiated with the for keyword.

range(5):

range(5) generates a sequence of numbers from 0 to 4 (5 is exclusive). The loop iterates over these values.

print(i):

In each iteration, the value of i is printed.

while Loop:

The while loop continues executing a block of code as long as a specified condition is true.

```
while condition:

    # code block
# Using while loop to print numbers until a certain condition is met
counter = 0
while counter < 5:
   print(counter)
   counter += 1
```

while Statement:

The while loop is initiated with the while keyword.

counter < 5:

while counter < 5 sets up a condition for the loop to continue as long as counter is less than 5.

print(counter):

Inside the loop, the current value of counter is printed.

counter += 1:

counter is incremented in each iteration to eventually exit the loop.

Lists:

- Ordered, mutable collection.
- Common operations: indexing, slicing, appending, and removing.

```
my_list = [1, 2, 3, 4, 5]

my_list.append(6)
```

Characteristics:

Ordered:

Lists maintain the order of elements based on their index, starting from 0 for the first element.

Mutable:

Lists are mutable, meaning you can modify their elements by changing, adding, or removing items.

```
# Example List
my_list = [1, 2, 3, 4, 5]

# Indexing
```

```
first_element = my_list[0]  # 1

# Slicing
subset = my_list[1:4]  # [2, 3, 4]

# Appending
my_list.append(6)  # [1, 2, 3, 4, 5, 6]

# Removing
my_list.remove(3)  # [1, 2, 4, 5, 6]
```

Dictionaries:

- Unordered, mutable collection of key-value pairs.
- Access values using keys.

```
my_dict = {"name": "John", "age": 25}

print(my_dict["name"])
```

```
# Example Dictionary
my_dict = {"name": "John", "age": 25}

# Accessing Values
name_value = my_dict["name"]  # "John"

# Adding a New Key-Value Pair
my_dict["city"] = "New York"  # {"name": "John", "age": 25, "city": "New York"}

# Updating a Value
my_dict["age"] = 26  # {"name": "John", "age": 26, "city": "New York"}

# Removing a Key-Value Pair
del my_dict["city"]  # {"name": "John", "age": 26}
```

Tuples:

- Ordered, immutable collection.
- Useful for representing fixed collections of items.

Immutable:

Tuples are immutable, meaning once they are created, their elements cannot be changed or modified. You cannot add, remove, or update elements in a tuple.

```
my_tuple = (1, 2, 3)
```

```
# Example Tuple
my_tuple = (1, 2, 3)

# Accessing Elements
first_element = my_tuple[0]  # 1

# Slicing
subset = my_tuple[1:3]  # (2, 3)
```

File Handler:

Reading from Files:

- open() function is used to open a file.
- read(), readline(), or readlines() methods are used for reading file content.

```
with open("filename.txt", "r") as file:
    content = file.read()
print(content)
```

with Statement:

The with statement is used to ensure that the file is properly closed after reading its content. It is a recommended way of working with files in Python as it automatically takes care of closing the file, even if an error occurs.

open() Function:

open("filename.txt", "r") opens the file named "filename.txt" in read mode ("r").

file.read():

The read() method reads the entire content of the file as a single string and stores it in the variable content.

File Closing:

As the file is opened using the with statement, it is automatically closed when the block is exited.

Writing to Files:

- open() function with mode "w" is used for writing.
- write() method is used for writing to the file.

```
with open("output.txt", "w") as file:
    file.write("Hello, World!")
```

with Statement:

The with statement is used to ensure that the file is properly closed after writing to it. It automatically takes care of closing the file, even if an error occurs.

open() Function:

open("output.txt", "w") opens the file named "output.txt" in write mode ("w").

file.write():

The write() method is used to write the specified content ("Hello, World!") to the file.

File Closing:

As the file is opened using the with statement, it is automatically closed when the block is exited.