CS2008: Numerical Computing

Muhammad Almas Khan

School of Computing
Department of Computer Sciences

January 22, 2025

Session Overview

- Types in Python and Type Conversion
- Mutable and Immutable Types
- Anonymous Functions
- Classes and Object-Oriented Concepts
- Exception Handling
- Activities to Solidify Learning

Understanding Types in Python

Key Data Types:

- **Primitive:** ‘int‘, ‘float‘, ‘str‘, ‘bool‘
- **Collections:** ‘list‘, ‘tuple‘, ‘set‘, ‘dict‘

Type Conversion Example:

```
# Converting between types
x = 42      # int
pi = str(3.14) # float to string
y = int("10") # string to int
```

Methods of List and Tuple

Container	Method	Description
List	append()	Adds an item to the end of the list.
	extend()	Adds all elements of an iterable to the list.
	remove()	Removes the first occurrence of a value.
	pop()	Removes and returns the item at the given index.
Tuple	count()	Returns the number of occurrences of a value.
	index()	Returns the index of the first occurrence of a value.

Methods of Set and Dictionary

Container	Method	Description
Set	add()	Adds an item to the set.
	remove()	Removes an item from the set (raises error if not found).
	discard()	Removes an item if it exists, does nothing if not.
	pop()	Removes and returns an arbitrary item from the set.
Dictionary	get()	Returns the value for a given key.
	keys()	Returns all keys in the dictionary.
	values()	Returns all values in the dictionary.
	items()	Returns all key-value pairs.
	pop()	Removes and returns a key-value pair.

For detailed documentation, visit: https://docs.python.org/3/tutorial/datastructures.html

Let's Implement It in Jupyter Notebook Now!

Jupyter Notebook!

Mutable vs Immutable Types

Definition:

- **Mutable:** Objects that can be changed after creation (e.g., ‘list‘, ‘dict‘, ‘set‘).
- **Immutable:** Objects that cannot be changed after creation (e.g., ‘int‘, ‘float‘, ‘str‘, ‘tuple‘).

Example: Mutable vs Immutable:

```
# Mutable
my_list = [1, 2, 3]
my_list[0] = 10 # Changes the first element

# Immutable
my_tuple = (1, 2, 3)
# my_tuple[0] = 10 # Raises an error
```

Are you ready?

Jupyter Notebook!

Activity 1: Exploring Types

Task: Identify Data Types

```
# Example:
data = [42, 3.14, "Python", True]
for item in data:
    print(f"{item} is of type {type(item)}")
```

Your Turn:

- Create a list with mixed data types.
- Write a loop to print each item and its type.

Activity 2: Mutable vs Immutable

Task: Explore Mutability

- Create a list and modify one of its elements.
- Create a tuple and try modifying one of its elements.
- Observe the behavior and errors (if any).

Hint: Use indexing to modify elements of the list or tuple.

Anonymous Functions (Lambda)

What is a Lambda Function?

- A one-liner function defined with ‘lambda‘.
- Useful for simple operations.

Example:

```
# Lambda to calculate square
square = lambda x: x**2
print(square(5)) # Output: 25
```

Activity 3: Using Lambdas

Task: Create and Use Lambda Functions

- Write a lambda function to multiply two numbers.
- Use it to calculate the product of 7 and 3.
- Extend it to find cubes using ‘lambda‘.

Hint: Use ‘lambda a, b: a * b‘ for multiplication.

Understanding Classes

What are Classes?

- A blueprint for creating objects.
- Encapsulates data (attributes) and methods (functions).

Example:

```
class Animal:
    def __init__(self, name):
        self.name = name

    def speak(self):
        print(f"{self.name} makes a sound.")

# Create an object
dog = Animal("Dog")
dog.speak() # Output: Dog makes a sound
```

Activity 4: Creating Classes

Task: Define a Class

- Create a ‘Person‘ class with attributes ‘name‘ and ‘age‘.
- Add a method ‘introduce‘ to print ‘My name is ¡name¿, and I am ¡age¿ years old.‘
- Create an object and call the method.

Handling Exceptions in Python

Why Handle Exceptions?

- Prevents program crashes.
- Handles unexpected errors gracefully.

Example:

```
try:
    result = 10 / 0
except ZeroDivisionError as e:
    print("Error:", e)
finally:
    print("Execution complete.")
```

Activity 5: Try-Except Block

Task: Handle User Input Errors

- Write a program to take user input for an integer.
- Handle ‘ValueError‘ if the input is not a valid integer.
- Print a message indicating success or failure.

Summary

- Key data types and type conversion.
- Mutable vs immutable types.
- Lambda functions for simple tasks.
- Classes and object-oriented concepts.
- Exception handling for error management.

Next Step: Practice and apply the concepts.
Check out the cheatsheet I posted on GCR for more details.