Data Types in Python

Lab Objective:

By the end of this lab, students will:

1. Understand the different **data types** in Python, including integers, floats, strings, booleans, lists, tuples, sets, and dictionaries.
2. Learn how to perform operations on different data types.
3. Explore how to check and convert data types.
4. Work with complex data types like lists, tuples, and dictionaries.

Part 1: Basic Data Types in Python

Objective:

- Understand and work with basic data types such as integers, floats, strings, and booleans.

Tasks:

1. **Integers and Floats:**
    - Declare variables of integer and float types, perform arithmetic operations, and print the results.

    Code Example:

    ```
    # Declare variables
    x = 10  # Integer
    y = 3.14  # Float

    # Perform arithmetic operations
    sum_result = x + y
    prod_result = x * y

    print("Sum:", sum_result)
    print("Product:", prod_result)
    ```

2. **Strings:**
    - Declare string variables and perform string operations like concatenation and slicing.

Code Example:

```
name = "Python"
greeting = "Hello, " + name  # String concatenation
print(greeting)

# String slicing
print(name[0:3])  # Output: Pyt (slicing from index 0 to 2)
```

3. **Booleans:**
    - Use booleans to evaluate conditions.

Code Example:

```
is_python_fun = True
is_java_fun = False

print("Is Python fun?", is_python_fun)
print("Is Java fun?", is_java_fun)
```

Exercise 1:

- Declare two integer variables and one float variable. Perform addition, subtraction, multiplication, and division, then print the results along with their types.

Part 2: Complex Data Types

Objective:

- Learn about complex data types like lists, tuples, sets, and dictionaries.

Tasks:

1. **Lists:**
    - Create a list, append new items, access elements by index, and modify elements.

Code Example:

```
# Create a list
fruits = ["apple", "banana", "cherry"]

# Access elements by index
print(fruits[0])  # Output: apple

# Append a new item
fruits.append("orange")
print(fruits)
```

```
# Modify an element
fruits[1] = "blueberry"
print(fruits)
```

2. **Tuples:**
    - Create a tuple and demonstrate that it is immutable by trying to modify its elements.

Code Example:

```
# Create a tuple
coordinates = (10, 20, 30)

# Access elements
print(coordinates[1])  # Output: 20

# Trying to modify a tuple will raise an error
# coordinates[1] = 25  # This will cause an error
```

3. **Sets:**
    - Create a set, add new items, and demonstrate the uniqueness property of sets.

Code Example:

```
# Create a set
colors = {"red", "green", "blue"}

# Add a new item
colors.add("yellow")

# Demonstrate uniqueness
colors.add("red")  # 'red' is already in the set, so it won't be added
again
print(colors)
```

4. **Dictionaries:**
    - Create a dictionary, access values by key, and modify entries.

Code Example:

```
# Create a dictionary
student = {"name": "John", "age": 22, "major": "Computer Science"}

# Access values by key
print(student["name"])

# Modify an entry
student["age"] = 23
print(student)
```

Exercise 2:

- Create a list of your favorite movies, append a new movie to the list, and then modify one of the existing movie titles.
- Create a dictionary with details about a book (title, author, year published). Modify the year published and print the updated dictionary.

Part 3: Checking and Converting Data Types

Objective:

- Learn how to check the data type of a variable and how to convert between different types.

Tasks:

1. **Checking Data Types:**
   - Use the `type()` function to check the type of a variable.

   Code Example:

   ```
   age = 25
   height = 5.9
   name = "Alice"

   # Check data types
   print(type(age))  # Output: <class 'int'>
   print(type(height))  # Output: <class 'float'>
   print(type(name))  # Output: <class 'str'>
   ```

2. **Type Conversion:**
   - Convert variables from one type to another (e.g., integer to float, string to integer).

   Code Example:

   ```
   num_str = "100"
   num_int = int(num_str)  # Convert string to integer
   print(num_int, type(num_int))  # Output: 100 <class 'int'>

   # Convert integer to float
   num_float = float(num_int)
   print(num_float, type(num_float))  # Output: 100.0 <class 'float'>
   ```

Exercise 3:

- Declare a variable with an integer value. Convert it into a float and a string, then print the value and its type after each conversion.

Part 4: Data Type Operations

Objective:

- Explore operations that can be performed on different data types, such as list concatenation, dictionary key-value access, and set operations.

Tasks:

1. **List Operations:**
    - Concatenate two lists and use list slicing.

    Code Example:

    ```
    list1 = [1, 2, 3]
    list2 = [4, 5, 6]

    # Concatenate lists
    combined_list = list1 + list2
    print("Combined List:", combined_list)

    # List slicing
    sliced_list = combined_list[1:4]
    print("Sliced List:", sliced_list)
    ```

2. **Set Operations:**
    - Perform union and intersection on sets.

    Code Example:

    ```
    set1 = {1, 2, 3}
    set2 = {3, 4, 5}

    # Union of two sets
    union_set = set1.union(set2)
    print("Union:", union_set)

    # Intersection of two sets
    intersection_set = set1.intersection(set2)
    print("Intersection:", intersection_set)
    ```

3. **Dictionary Operations:**
    - Add a new key-value pair to a dictionary and delete an existing entry.

Code Example:

```
student = {"name": "John", "age": 22}

# Add a new key-value pair
student["major"] = "Computer Science"
print(student)

# Delete a key-value pair
del student["age"]
print(student)
```

Exercise 4:

- Create two sets of your favorite fruits. Perform a union and intersection on the sets and print the results.
- Create a dictionary representing a person's contact details (name, phone, email). Add a new field for address and delete the phone number.

Assignment:

1. **Problem 1:**
    - Write a Python program to declare variables of different data types (integer, float, string, boolean). Print the values and their types. Then, convert the integer to float and string, and print the converted values.
2. **Problem 2:**
    - Create a Python program that defines a list of numbers. Perform the following:
        - Append a new number to the list.
        - Sort the list in ascending order.
        - Remove an element from the list.
        - Print the final list.
3. **Problem 3:**
    - Write a program that creates a tuple of your favorite foods. Access the first and last item in the tuple and print them. Demonstrate that tuples are immutable by trying to modify an element.
4. **Problem 4:**
    - Create a dictionary with the details of a car (brand, model, year). Update the year, add a new key for color, and print the updated dictionary. Delete the model entry from the dictionary and print the final dictionary.
5. **Problem 5:**
    - Write a Python script that creates a set of unique numbers. Add new numbers to the set and demonstrate that sets do not allow duplicates. Perform and print the union and intersection of two sets.