DATA SCIENCE COURSE TUTORIAL # 11

3.4 Variables and Data Types in Python

1: Variables

In Python, a variable is a name that refers to a value. You can think of a variable as a container that holds data. To create a variable, you simply assign a value to it using the equals sign (`=`).

```python
# Example of variable assignment
x = 5
y = "Hello, World!"
```

2: Rules for Naming Variables

When naming variables in Python, there are some important rules to follow:

1. **Valid Characters**: Variable names can only contain letters (a-z, A-Z), numbers (0-9), and underscores (_). They cannot start with a number.
2. **Case Sensitivity**: Variable names are case-sensitive. For example, `myVar` and `myvar` are two different variables.
3. **No Spaces**: Variable names cannot contain spaces. Use underscores (_) to separate words.
4. **Reserved Words**: Avoid using Python's reserved words (keywords) as variable names (e.g., `if`, `else`, `while`, `def`, etc.).

3: Data Types (Actual Built-in Types)

Python has some fundamental built-in data types that directly represent values.

- **Numeric Types**
  - `int`: Integer values without a decimal point. These can be positive, negative, or zero.
    Example: `5`, `-10`, `0`
  - `float`: Floating-point numbers, which represent decimal values. They can also be written in scientific notation.
    Example: `3.14`, `-0.001`, `2e3` (which means 2000.0)
  - `complex`: Complex numbers that have a real part and an imaginary part. The imaginary part is represented with `j`.
    Example: `2 + 3j`, `-1j`
- **Boolean Type**
  - `bool`: Represents logical values. It has only two possible values: `True` and `False`. Internally, `True` is equivalent to `1` and `False` is equivalent to `0`.
    Example: `is_active = True`, `5 > 3` (returns `True`)

- **Text Type**
    - `str`: Represents a sequence of characters (string). Strings can be written in single quotes (`' '`), double quotes (`" "`), or triple quotes (`''' '''` or `""" """`).
      Example: `"Hello"`, `'Python'`, `"""This is a string"""`

You can check the data type of a variable using the `type()` function:

```
# Example of checking data types
x = 5
y = "Hello"
print(type(x))  # Output: <class 'int'>
print(type(y))  # Output: <class 'str'>
```

4: Dynamic Typing

Python is dynamically typed, meaning you don't have to explicitly declare the data type of a variable when you create it. The interpreter infers the type based on the value assigned to the variable. This allows for more flexibility in your code but can also lead to runtime errors if you're not careful.

```
# Example of dynamic typing
x = 5          # x is of type int
x = "Hello"    # x is now of type str
```

5: Assigning Multiple Variables

You can assign values to multiple variables in a single line using commas:

```
# Example of assigning multiple variables
a, b, c = 1, 2, 3
```

6: Reassigning Variables

You can reassign a variable to a new value at any time:

```
# Example of reassigning variables
x = 5
x = 10
```