**# Introduction to Computer Science and Programming**

**Course Goals**

Computer Science (CS) is a discipline that blends theoretical foundations with practical applications to solve complex problems using computational methods. This introductory course is designed to:

1. Provide a **solid foundation** in computer science principles.
2. Develop **problem-solving skills** using computational thinking.
3. Teach **programming fundamentals** with a focus on Python.
4. Introduce students to **data types, operators, and variables**.
5. Foster an **understanding of computation** and how it translates into solving real-world problems.

**What is Computation?**

**Computation** is the process of performing calculations or processing information systematically using algorithms. An algorithm is a **finite sequence of well-defined instructions** that transforms input into a desired output.

**Key Concepts of Computation:**

* **Algorithms**: Step-by-step procedures for solving problems.
* **Programming**: Writing instructions that a computer can execute.
* **Abstraction**: Simplifying complex systems by breaking them down into manageable components.
* **Efficiency**: Measuring how well a computational solution performs in terms of speed and resource usage.

**Real-World Example:**

A recipe for baking a cake is an algorithm:

1. Gather ingredients.
2. Mix them in a specified order.
3. Bake at a given temperature.
4. Serve the cake.

In computing, an analogous algorithm could be:

1. Receive user input.
2. Process input using predefined rules.
3. Return the output as a response.

**Introduction to Data Types, Operators, and Variables**

**1. Data Types**

A data type specifies the kind of value a variable can store. In Python, common data types include:

* **Integer (int)**: Whole numbers (e.g., 10, -3).
* **Float (float)**: Decimal numbers (e.g., 3.14, -0.99).
* **String (str)**: Textual data (e.g., 'Hello', 'Python').
* **Boolean (bool)**: Logical values (True or False).
* **List (list)**: Ordered collection of values (e.g., [1, 2, 3]).
* **Tuple (tuple)**: Immutable ordered collection (e.g., (10, 20, 30)).
* **Dictionary (dict)**: Key-value pairs (e.g., {'name': 'Alice', 'age': 25}).

**Example Code:**

age = 25 # Integer

height = 5.9 # Float

name = "Alice" # String

is\_student = True # Boolean

**2. Operators**

Operators allow us to perform computations on data. They can be classified as:

**A. Arithmetic Operators**

* + (Addition): 5 + 3 → 8
* - (Subtraction): 10 - 4 → 6
* \* (Multiplication): 2 \* 6 → 12
* / (Division): 10 / 2 → 5.0
* // (Floor Division): 10 // 3 → 3
* % (Modulus): 10 % 3 → 1
* \*\* (Exponentiation): 2 \*\* 3 → 8

**B. Comparison Operators**

* == (Equal to): 5 == 5 → True
* != (Not equal to): 5 != 3 → True
* > (Greater than): 7 > 3 → True
* < (Less than): 2 < 5 → True

**C. Logical Operators**

* and (Logical AND): True and False → False
* or (Logical OR): True or False → True
* not (Logical NOT): not True → False

**Example Code:**

x = 10

y = 5

print(x + y) # Output: 15

print(x > y) # Output: True

**3. Variables**

A **variable** is a named location in memory that stores data. In Python, variables are dynamically typed, meaning you don’t have to declare their type explicitly.

**Variable Naming Rules:**

✅ Must begin with a letter (A-Z or a-z) or an underscore (\_). ✅ Cannot use reserved keywords (e.g., if, else, print). ✅ Should be descriptive (e.g., num\_students instead of n).

**Example Code:**

name = "Alice"

age = 25

height = 5.4

is\_student = True

print(name, age, height, is\_student)

**Summary of Key Takeaways**

* **Computer Science** involves problem-solving using computational methods.
* **Computation** is the execution of well-defined steps (algorithms) to achieve a goal.
* **Programming** involves using a language like Python to instruct a computer.
* **Data types** define what kind of values a variable can hold (e.g., integers, floats, strings, booleans).
* **Operators** allow performing mathematical and logical computations.
* **Variables** store and manipulate data in a program.

**Exercises**

**1. Understanding Computation**

* Define an algorithm for making a cup of tea.
* Write pseudocode for an algorithm that calculates the square of a number.

**2. Hands-on with Variables and Data Types**

Write a Python program that:

1. Takes user input for name, age, and height.
2. Stores the values in variables.
3. Prints a sentence using these values.

**Bonus Challenge:**

Modify the program to check if the person is old enough to vote (age >= 18).

By completing this lecture, students will have a foundational understanding of computation, data types, operators, and variables, setting the stage for deeper learning in programming and computer science.

**End of Lecture**