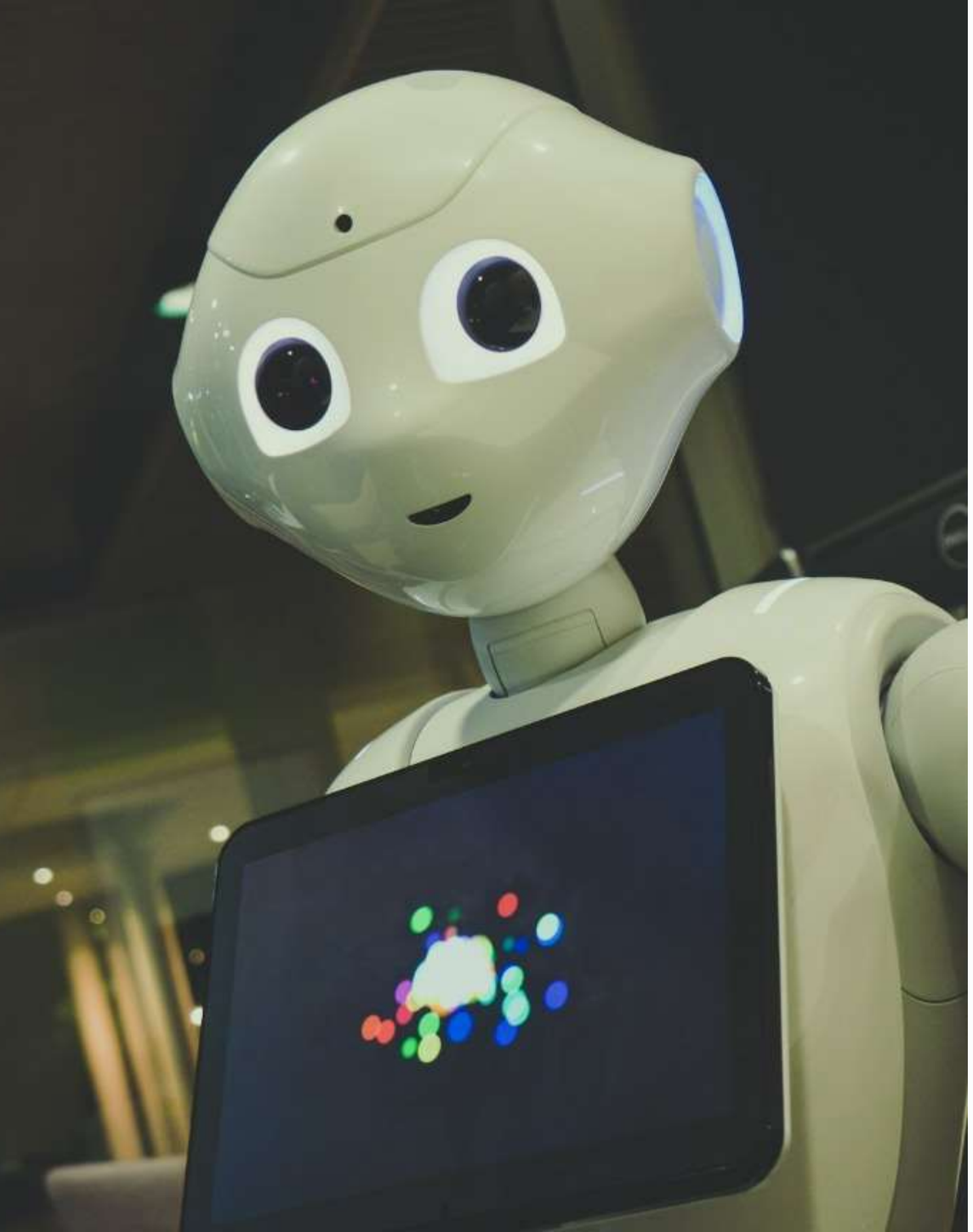
Welcome & Course Overview



"AI and Applications of AI"

Course Highlights

- Comprehensive Python Foundation
- Foundational Math and Stats for AI
- Practical Machine Learning Implementation
- Introduction to Deep Learning

Presenters

- Prince Karna
- Pujan Pant
- Piyush Phuyal

25th May,2025

Python Basics

Today's Agenda

- Why Python for AI?
- Variables & Data Types Deep Dive
- Control Structures for AI Logic
- Functions - Building Blocks of AI
- Python Environment and Setup
- Hands-On Practice
- Kahoot(Quiz)

Why Python for AI?

Simple Syntax

Easy to read and write

Rich Libraries

Strong AI and ML tools

Community Support

Active global developers

Industry Standard

Widely adopted in AI

Variables & Data Types Deep Dive -1

What is a Variable?

- A **variable** is like a **label** attached to a value in memory.
- It allows you to **store**, **modify**, and **reuse** data in your program.
- Think of it as a **named box** where you can put different kinds of information.

Variable Naming Rules

- Must begin with a **letter** (A-Z or a-z) or **underscore _ .**
- Can contain **letters, digits, and underscores.**
- **Case-sensitive**: **Name** and **name** are different.
- Avoid using Python **keywords** (like **if**, **class**, **for**).

How Do Variables Work in Python?

Python is **dynamically typed**:

- You **don't need to declare** the type of a variable.
- The interpreter automatically figures it out based on the assigned value.

Good Names:

- **name**, **user_age**, **temp**

Bad Names:

- **2score** (starts with a number)
- **class** (reserved words/keywords)

Example:

```
Message = "Hello"    # This is a string
Count = 15           # This is an integer
```

Why Are Variables Important?

- They make your code **reusable** and **maintainable**.
- You can **store user input**, **track values**, and **pass data** between functions or parts of your program.

Variables & Data Types Deep Dive -2

⍰ What is a Data Type?

- A **data type** defines the **kind of value** a variable can hold.
- Python automatically assigns a data type when you assign a value.
- Different data types are used for storing **numbers**, **text**, **lists**, **true/false**, etc.

🔢 Basic Built-in Data Types

Data Type	Description	Example
int	Integer	age = 25
float	Decimal	price = 99.99
str	String	name = “Piyush”
bool	Boolean	is_cool = True

🎁 Collection Data Types

Data Type	Description	Example
list	Ordered(changable)	marks = [90, 85, 78]
tuple	Ordered(unchangable)	x_axis = (1,0)
dict	Key-value pair	student = {"name": "Bob", "age": 20}

⍰ How Python Identifies Data Type?

- Python uses the **type of value** to understand what the variable is.

```
x = 10          # int
y = "hello"     # str
z = [1, 2, 3]   # list
```

You can check the type using the **type()** function:

```
print(type(x))  # Output: <class 'int'>
```

Question: Why Data Types Matter?

🤔 Why Data Types Matter?

- You can't mix incompatible types without errors.

  For example:

```python
age = 20
print("Age is " + age)  # ❌ Error: Cannot add string and integer
```

✅ Fix:

```python
print("Age is " + str(age))
```

Control Structures for AI Logic-1

🧠 What Are Conditionals?

- Conditionals allow a program to **make decisions**.
- They check whether something is **True or False**, and run code accordingly.
- Think of them as the "**if this, then that**" logic.

🔑 Basic Conditional Keywords

- `if` – Checks a condition.
- `elif` – Checks another condition if previous ones failed.
- `else` – Runs when all previous conditions are false.

✅ Example:

```python
age = 18

if age >= 18:
    print("You are an adult.")
else:
    print("You are a minor.")
```

📌 Syntax Rules

- Conditions go inside **parentheses (optional)**, but indentation is **mandatory**.
- The block of code inside `if`, `elif`, or `else` must be **indented** (usually 4 spaces).

🔄 Flow of Decision-Making

```python
if condition1:
    # Executes if condition1 is True
elif condition2:
    # Executes if condition1 is False and condition2 is True
else:
    # Executes if all above conditions are False
```

🧠 Real-Life Analogy

> **If** it's raining, **then** take an umbrella.
> **Else if** it's cloudy, **then** wear a jacket.
> **Else**, go outside as usual.

This logic can be translated into Python code!

Control Structures for AI Logic-2

🔍 Comparison Operators (used in conditions)

Operator	Meaning	Example
`==`	Equal to	`x == 5`
`!=`	Not equal to	`x != 5`
`>`	Greater than	`x > 5`
`<`	Less than	`x < 5`
`>=`	Greater or equal	`x >= 5`
`<=`	Less or equal	`x <= 5`

Short Break(10 min)

Loops for Data Processing

🧠 What Are Loops?

- Loops allow you to **repeat a block of code** multiple times.
- Useful when you want to **automate repetitive tasks**.
- Python mainly uses two types of loops: `for` and `while`.

🔁 1. `for` Loop – Iterate Over a Sequence

- Used to loop through a list, string, range, etc.
- It runs once **for each item** in the sequence.

✅ Example:

```python
fruits = ["apple", "banana", "mango"]
for fruit in fruits:
    print(fruit)
```

🔄 2. `while` Loop – Repeat Until Condition Fails

- Runs as long as the condition is **True**.
- Be careful: it can create an **infinite loop** if the condition never becomes False.

✅ Example:

```python
count = 0
while count < 3:
    print("Hello")
    count += 1
```

🔚 Loop Control Statements

Statement	Purpose
`break`	Exit the loop immediately
`continue`	Skip the rest of the current loop and move to the next iteration
`pass`	Do nothing (placeholder)

🧠 Real-Life Analogy

> **For loop:** Like reading names from a guest list — you go one by one.
> **While loop:** Like watering plants until the soil is wet — you repeat until a condition is met.

Functions - Building Blocks of AI-1

🧠 What Is a Function?

- A **function** is a named block of code designed to perform a **specific task**.
- You can **call** it multiple times instead of rewriting the same code.
- Think of it like a **machine**: you give input, it processes something, and may give output.

⚙️ Why Use Functions?

- Makes code **shorter, clearer, and organized**.
- Helps with **code reusability** — write once, use many times.
- Allows you to **break down complex problems** into smaller parts.

🔧 How to Define a Function in Python

```python
def function_name():
    # block of code
    print("This is a function")
```

✅ Calling a function:

```python
function_name()
```

🧠 Real-World Analogy

Imagine a coffee machine:

- "You press a button (call a function)"
- "It prepares coffee (runs the code)"
- "You get coffee (output)"

🚀 Use Case Examples

- A function to calculate area
- A function to send a welcome message
- A function to check login status

Functions - Building Blocks of AI-2

📥 Parameters (Inputs to Functions)

- Functions can accept data using **parameters**.

```python
def greet(name):
    print("Hello, " + name)
```

✅ Calling with a value:

```python
greet("Alice")  # Output: Hello, Alice
```

📤 Return Values

- Functions can return a value using the `return` keyword.

```python
def square(num):
    return num * num
```

✅ Capture the result:

```python
result = square(4)  # result = 16
```

🧩 Types of Functions (By Behavior)

Type	Description	Example
No input, no output	Just performs a task	`def show_time():`
Input, no output	Takes data, doesn't return	`def greet(name):`
Input and output	Takes and returns data	`def add(a, b):`

Main Break(30 min)

Python Environment & Setup

For the next few sessions, we will be using OneCompiler for our coding exercises. Google Colab will be used in the coming days. You can directly access OneCompiler by typing 'OneCompiler' into your web browser and selecting 'Python'.

Why OneCompiler?

- No installation required
- Pre-installed libraries
- Works on any device
- Instant feedback
- Save and share code
- Perfect for learning

Perfect for today's session! We can start coding immediately and focus on learning Python for AI.

1 Visit OneCompiler

Go to **onecompiler.com/python** in your web browser

2 Start Coding Immediately

You'll see a **code editor** ready to use - no setup needed!

```
# Try this simple code:
print("Hello, AI World!")
name = input("What's your name? ")
print(f"Welcome to AI course, !")
```

3 Run Your Code

- Click the **"Run"** button (or press Ctrl+Enter)
- See output in the **console** below

QnA

And

Feedback

Next: Data Structures Deep Dive