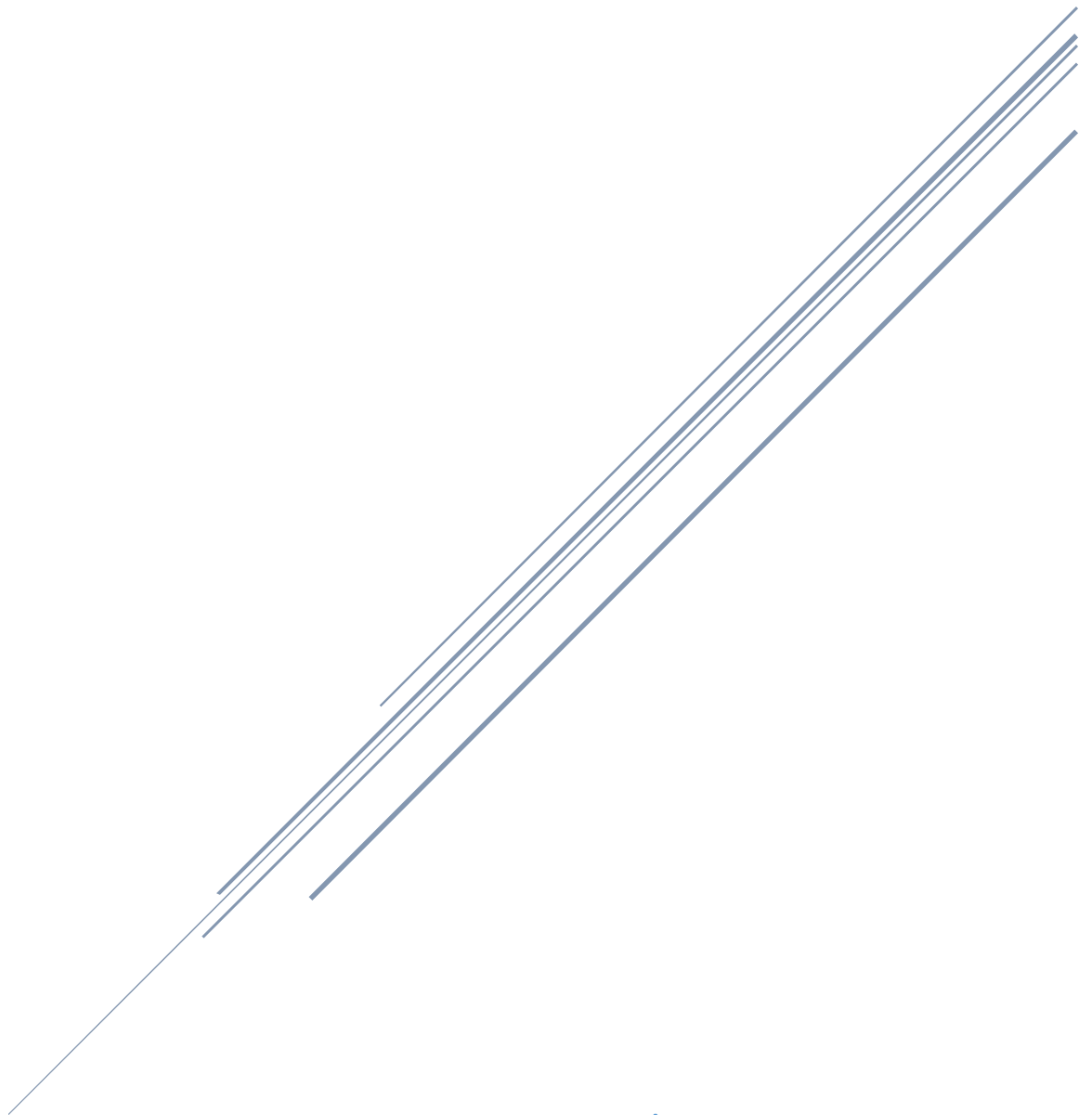


# CODE AND SUMMARY OF CLASS 4

Sahir Ahmed Sheikh

Saturday (2 – 5)



Teachers:

**Muhammad Bilal And Ali Aftab Sheikh**

## Code And Summary Of Class 4 – Saturday (2 – 5) | Quarter 3

Assalamu Alaikum!

Hope you all are doing well. In today's session, we focused on several key programming concepts and practical implementation. The class involved **checking assignments, installing Python, exploring new code editor (Cursor AI), learning about project management tools (UV), and Loops in python.**

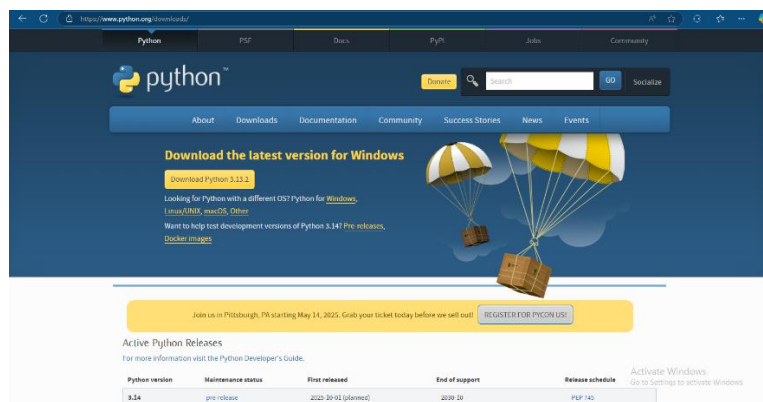
### Class Structure & Key Highlights

#### 1. Assignment Review

- Students were required to submit and verify previous assignments.
- Special CRs (Class Representatives) were selected based on performance.
- Additional assignments were given to hardworking students.
- **I am also included in the CR selection process and will be part of the interview for CR selection.**

#### 2. Installing Python & Setting Up Environment

- Students were guided through installing Python via Google search.
- The simplest way is to search "Download Python" and install it from the first link that appears.



- The installation process involves clicking "Next" multiple times.

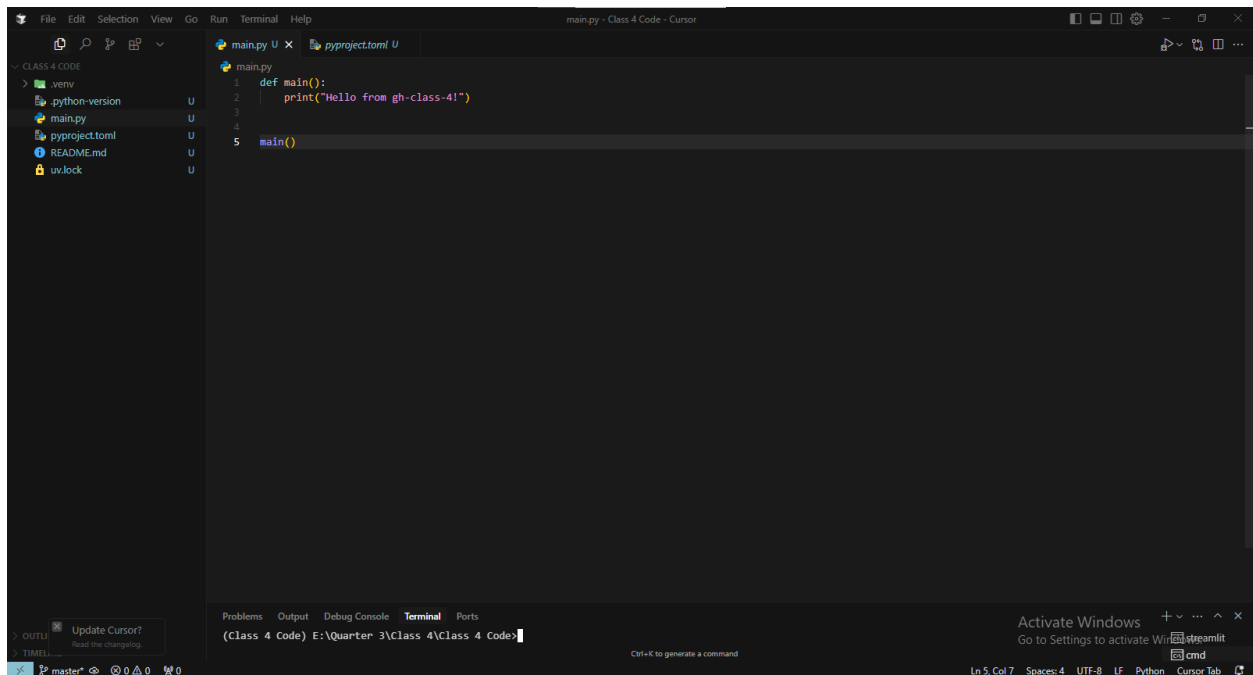
- The most important step is to check the **“Add to Path”** option. If missed, Python won’t work properly, and the environment variable will have to be set manually.
- After installation, confirm Python is installed using the command:

```
python --version
```

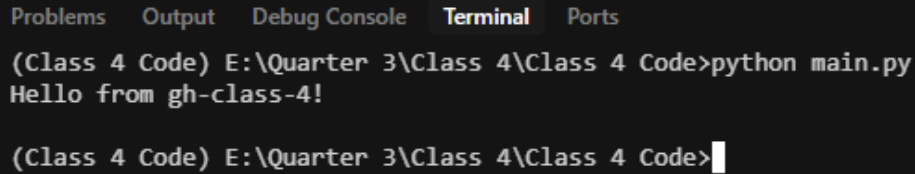
- If the version appears (e.g., Python 3.x.x), the installation was successful. Otherwise, reinstall and ensure "Add to Path" is checked.
- Next, students were guided to open a terminal and type python to enter the Python shell, where they tested a simple command:

```
C:\Users\System Angel>python
Python 3.13.2 (tags/v3.13.2:4f8bb39, Feb  4 2025, 15:23:48) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello world")
Hello world
>>>
```

- Python files use the .py extension, and a simple script main.py was created:



Running it in the terminal using:  
python main.py  
displayed Hello World in the output.



The screenshot shows a terminal window with a dark background. At the top, there are tabs for 'Problems', 'Output', 'Debug Console', 'Terminal', and 'Ports'. The 'Terminal' tab is active. The text in the terminal reads: '(Class 4 Code) E:\Quarter 3\Class 4\Class 4 Code>python main.py' followed by the output 'Hello from gh-class-4!'. Below this, the prompt '(Class 4 Code) E:\Quarter 3\Class 4\Class 4 Code>' is shown with a cursor.

### 3. Exploring Code Editor: Cursor AI

#### ➤ What is Cursor AI?

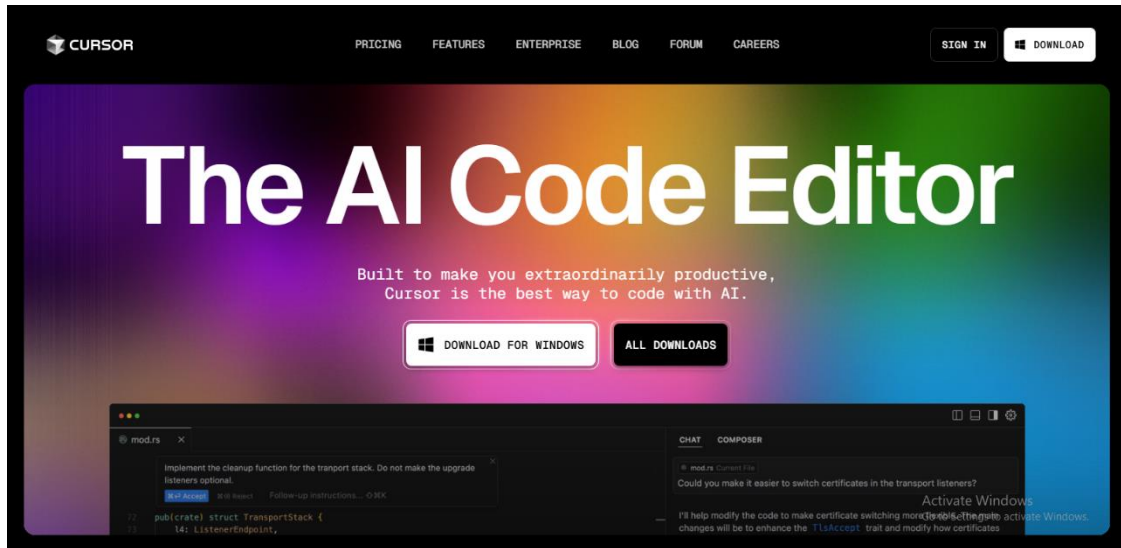
- Cursor AI is an AI-powered code editor built on top of VS Code.
- Since VS Code is open-source, developers have modified it to include AI-powered features, making coding easier and more efficient.
- Cursor AI provides features like code completion, AI-generated suggestions, and automatic debugging.
- It looks and functions almost identically to VS Code but has enhanced AI tools.

#### ➤ Cursor AI Pricing & Free Trial:

- It offers a **14-day free trial**.
- After the trial, it costs **\$20 per month** for AI features.
- If you don't want to pay, you can create a **new account** every two weeks and continue using it for free.

#### ➤ How to Install Cursor AI:

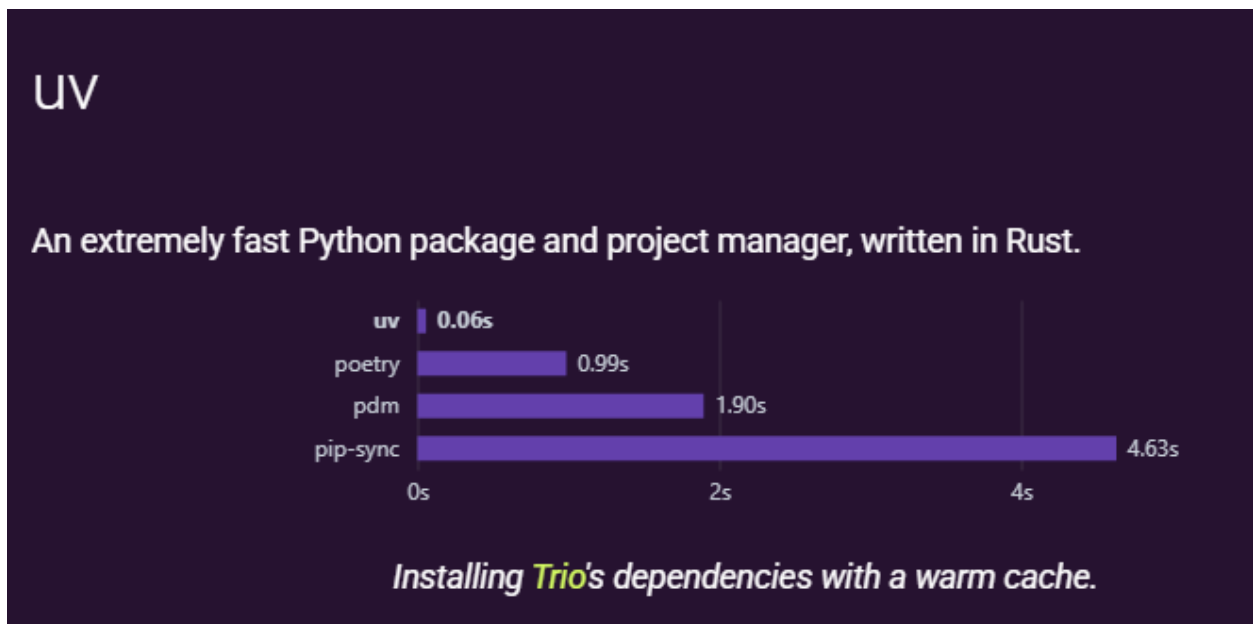
1. Go to **Google** and search for **Cursor AI Code Editor**.
2. Click on the **official website** and navigate to the download page.
3. Download the **Windows, Mac, or Linux** version according to your OS.
4. Install it like any other software.
5. Open Cursor AI, and you will notice that its interface is exactly like **VS Code**.



## 4. Introduction to UV Project Manager

### ➤ What is UV?

- UV is a Python package manager that is **much faster** than pip and Poetry.
- It is built using **Rust**, making it highly optimized for performance.
- Unlike pip, which installs dependencies one by one, UV installs them **in parallel**, making installation much quicker.



➤ **Why Use UV?**

- **Speed:** It installs packages up to **2x faster** than pip.
- **Efficiency:** Uses less memory and reduces dependency conflicts.
- **Modern Features:** Handles project environments automatically.

➤ **Installation Process:**

- Open **PowerShell** (recommended over CMD).
- Run the following command:

```
powershell -ExecutionPolicy ByPass -c "irm  
https://astral.sh/uv/install.ps1 | iex"
```

- Once installed, verify using:

```
uv --version
```

➤ **Setting Up a Project with UV:**

- To initialize a new Python project with UV, use:

```
uv init .
```

- This creates:
  - main.py: A Python script file.
  - project.toml: Stores dependencies and metadata (similar to package.json in Node.js).
  - Readme-md

➤ **Installing Packages with UV:**

- Instead of using pip install, UV provides a faster alternative:

```
uv install streamlit
```

- This installs streamlit within the project environment without affecting the global system.

Step-1 : UV DOCS : <https://docs.astral.sh/uv/>

Step-2 : Go to installation tab.

Step-3 : Copy powershell uv installation command  
`powershell -ExecutionPolicy Bypass -c "irm https://astral.sh/uv/install.ps1 | iex"`

Step-4 : Open Powershell and run the copied command

Step-5 : Initialize project : `uv init .`

Step-6 : Add Package : `uv add <package-name>`

Step-7 : `uv venv .venv\Scripts\activate`

## 5. Working with Loops in Python

### ➤ What are Loops?

- Loops allow us to execute a block of code **multiple times** without repeating the code manually.
- Python provides two primary loops:
  1. **While Loop** (based on a condition)
  2. **For Loop** (used for iterating over sequences)

### ➤ While Loop

- The **while loop** continues to execute as long as the condition remains True.
- Example:

```
count = 1  
✓ while count <= 3:  
    print("Kacha papita paka papita")  
    count += 1
```

- **Key Points:**

1. The loop runs **as long as the condition is True**.
2. The variable count is incremented inside the loop to avoid an **infinite loop**.
3. If we forget to increment count, the loop will **never stop**, leading to an infinite loop.

4. To manually stop an infinite loop, press **CTRL + C** in the terminal.

### ➤ For Loop

- The **for loop** is used for **iterating over a sequence** (list, tuple, dictionary, etc.).
- Example:

```
iftar_items: list = ["Khujoor", "Samosay", "Pakoray", "Rooh Afza", "Chana chaat" ]

for items in iftar_items:
    print(items)
```

- **Key Points:**

1. The loop automatically assigns each item in the list to the variable item.
2. The loop stops when all elements have been iterated over.

### ➤ Summation Using Loops:

we created a function to sum multiple numbers

```
def items(*n):
    sum = 0
    for item in n:
        sum += item
    print(sum)

items(2, 2, 5, 6, 7, 2, 2)
```

💎 Output: 26

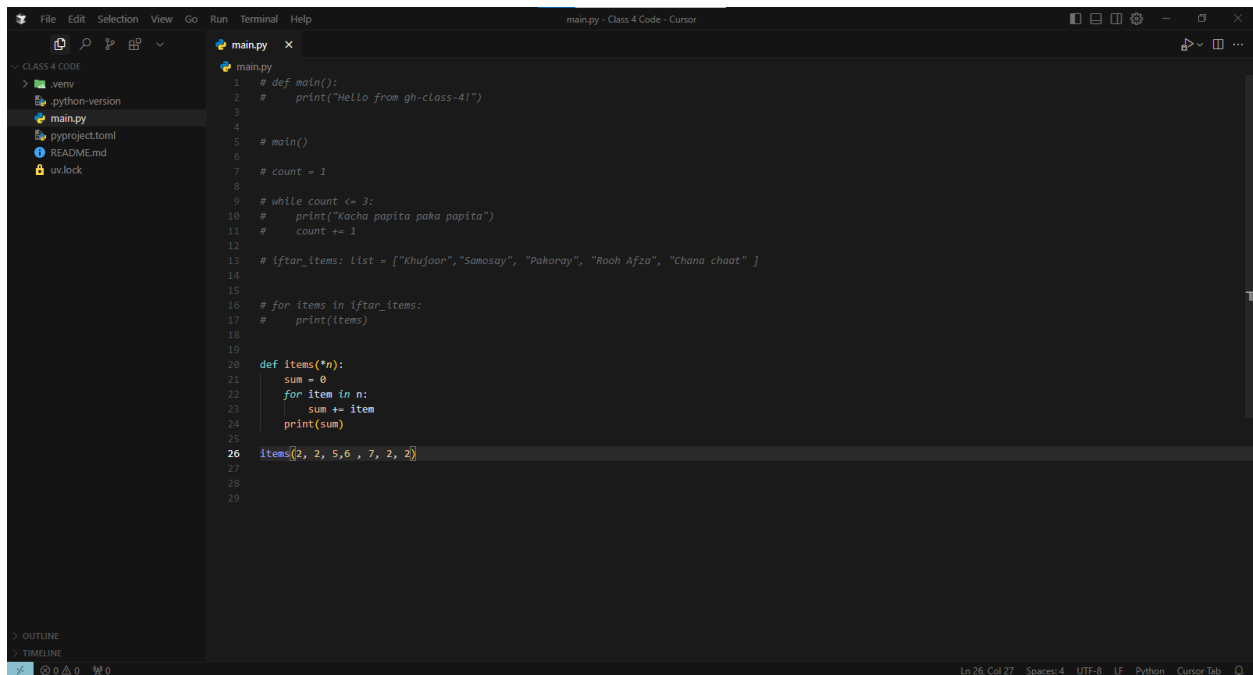


## Class Resources & Assignments

### Today's Class Code:

The complete code for today's class has been attached as an image below, allowing you to view it in its entirety. Additionally, I have provided a GitHub link where you can access the complete code as well.

Repo Link: [Class Code](#)

A screenshot of a code editor window titled 'main.py - Class 4 Code - Cursor'. The editor shows a Python file named 'main.py' with the following code:

```
1 # def main():
2 #     print("Hello from gh-class-4!")
3
4
5 # main()
6
7 # count = 1
8
9 # while count <= 3:
10 #     print("Kacha papita paka papita")
11 #     count += 1
12
13 # iftar_items: List = ["Khujaor", "Samosay", "Pakoray", "Raah Afza", "Chana chaat" ]
14
15
16 # for items in iftar_items:
17 #     print(items)
18
19
20 def items(*n):
21     sum = 0
22     for item in n:
23         sum += item
24     print(sum)
25
26 items(2, 2, 5, 6, 7, 2, 2)
```

The code includes comments and a function 'items' that calculates the sum of a list of numbers. The editor interface includes a sidebar with a file explorer showing 'CLASS 4 CODE' and 'main.py', and a bottom status bar indicating 'Ln 26, Col 27, Spaces: 4, UTF-8, LF, Python, Cursor Tab'.

### 🎯 Assignments

#### ✓ **Assignment 1:** Personal Library Manager

Work on the Personal Library Manager project from the repository:

[Personal Library Manager - GitHub Repo](#)

#### ✓ **Assignment 2:** Complete Previous Assignments

Ensure all previous assignments are completed and submitted on LinkedIn and the class submission form.

### ✓ **Assignment 3:** Problem-Solving Challenges

#### ⚙️ Problem 1: Reverse a String

Write a function that takes a string as input and returns the reversed string.

Example:

⚙️ Input: "hello"

⚙️ Output: "olleh"

💡 Hint: Use Python's slicing feature.

#### ⚙️ Problem 2: Count Vowels in a String

Write a function that counts the number of vowels (a, e, i, o, u) in a string (case-insensitive).

Example:

⚙️ Input: "Apple"

⚙️ Output: 2

💡 Hint: Use a loop and check if each character is in a set of vowels.

#### ⚙️ Problem 3: Sum of Digits

Write a function that takes a non-negative integer and returns the sum of its digits.

Example:

⚙️ Input: 1234

⚙️ Output: 10

💡 Hint: Convert the number to a string and iterate over each digit or use modulus and division.

 **Deadline:** Before the next class

 **Submission:**

Upload all assignments on LinkedIn

Submit all class assignments using this form: [Assignment Submission Form](#)

Stay consistent and keep learning!

# Thank You for Reading!

Hope you understood Class 4 well.

"There are no secrets to success. It is the result of preparation, hardwork, and learning from failure." – *Colin Powell*