So welcome back. Today we are going to discuss and learn that what are comments, why we use comments in our code. We have already discussed some of this in previous class, but we will continue this lecture from comment , moving forward we will talk about what are the types of comments. There are two types, single line comments and multi-line comments.

2nd we talk about what are variables in Python And what is the naming convention of variables in python obviously. Then we will perform some exercise of how to name a variable so that we get minimum of the syntax errors in our code.

After that we will define what scope is and what is scope of variables. There are two scope of variables. The number one is local variable and the number two is global variable.

What is Comment?

A comment in Python is text that you don't want to display or execute. In simple terms, it's a way to hide information within your code.

Benefits of comments:

Comments have several benefits, and one of the main ones is that they help explain what specific sections of your code do. For example, after learning Python and working on developing a game, like a snake game, you might be assigned a project where you have to create the snake game in Python.

Once you're done with the project, I may conduct a viva or test to ask you about the code you wrote. To help explain your code during such assessments, you can use comments to describe the functionality of different sections. For instance, in a snake game, you might have a feature where, after the snake eats three small pieces of food, a larger piece appears at a random location within the window. You can add a comment in your code saying something like, "This part of the code creates a large food item after the snake eats three smaller items."

A second benefit of using comments is that they allow you to credit the original creator of the code, especially if you've downloaded or copied the code from the internet. The original author can include their name, the date the code was created, or any other relevant information in a comment. This ensures that if someone else uses or modifies the code in the future, they know who the original author is and have access to any additional details provided.

Another benefit of comments is that you can use them to hide warnings in your code. Warnings are different from errors. Errors stop your code from running and need to be fixed,

but warnings are just alerts that don't stop the code from working. You can use comments to hide these warnings so they don't show up when you run the code, keeping the output clean.

Moving on, there are two types of comments: **single-line comments** and **multi-line comments**.

1. **Single-line comment**: This is used to comment out a single line of code. It allows you to hide or ignore just one line during execution.
2. **Multi-line comment**: This is used when you want to comment out multiple lines or a block of text. It can hide several lines at once, like a paragraph.

For a better understanding and examples, I recommend checking out slides 4 and 5 of lecture 4, where you'll find examples that show how to use both single-line and multi-line comments.

On slide 6, I explain that when using multi-line comments in the shell, we might see error. This happens because, in the shell, you can only execute one line of code at a time. For example, you write one line, press enter, and it gets executed.

However, for multi-line code or comments, you need to create a file and write the script in that file. You then save the file with the multi-line code or comments, and finally, you run the script. After running it, you can check the output in the shell. This method allows you to handle multiple lines effectively.

What are Variables?

Variable: It is just like box where we store things. In programming Box will be variables and things will be data stored in the memory of computer. Data must be stored in the memory before it is processed.

In **programming, a variable** is a value that can change, depending on conditions or on information passed to the program.

In python, the programmer does not need to declare the variable type explicitly, we just need to assign the value to the variable.

For examples view slide 7 and 8 of lecture 4

How to name Variables?

It is important to follow some rules in order to name variables because if we do not follow rules IDE will not understand. Below are some rules must follow while naming variables

1. The first character of a variable name must be alphabet (a-z,A-Z) or underscore (_)

2. The characters allowed within the name are underscore, digits, upper-case letters, lowe-case letters.

3. Special symbols such as %^&*!!@# are not allowed

4. Blank space or comma are not allowed

5. Variables are case sensitive.

6. Variable name can’t start from number

View slide 10 for examples.

Activity: Name the variables and ask in class which ones are valid or not valid

Scope of Variables:

Scope is like a permission or boundary that defines where you can do certain things. For example, imagine we're in a classroom. Inside the classroom, you are allowed to dance, run, smile, laugh—basically anything you want to do. But once you step outside the classroom, you're no longer allowed to do those things, like dancing or laughing.

So, inside the class is your **scope**—the place where you're allowed to act freely. Outside of it, those permissions no longer apply. Similarly, in programming, scope defines where certain variables or functions can be used.

There are two types of variable scopes: **local** and **global**.

1. **Local Variable**: Think of it like living in Pakistan without a passport or visa. You can travel within the country using your CNIC, but you can't leave the country. This is like a local variable—it's defined and can only be used within a specific block of code, like a function. Once you're outside of that block, you can't access the local variable.
2. **Global Variable**: Now imagine you have a passport and visa. You can travel from Pakistan to America, then to Turkey, Malaysia—anywhere in the world, you're free to move. This is like a global variable. It's defined at the top of your program and can be accessed from anywhere in the code. Any function or block of code can use it, just like you can travel anywhere on the globe.

In short, a **global variable** is available throughout the entire program, while a **local variable** is only accessible within the block or function where it's defined.

File Edit Format Run Options Window Help

```
# Global variable
x = 10

def my_function():
    # Local variable
    y = 5
    print("Inside the function:")
    print("Global variable x:", x)  # Accessing global variable
    print("Local variable y:", y)    # Accessing local variable

my_function()  # Call the function

# Accessing global variable outside the function
print("Outside the function:")
print("Global variable x:", x)

#Trying to access local variable y outside the function will cause an error
# Uncommenting the line below will result in a NameError
# print("Local variable y:", y)  # This will raise an error
```

Explanation: The global variable x is defined with a value of 10 and can be accessed both inside and outside the function. Inside the function my_function(), a local variable y is defined with a value of 5. Local variables, like y, can only be accessed within the function where they are defined. The function prints both the global variable x and the local variable y when called. After the function is executed, the program successfully prints the global variable x again outside the function. However, if you try to access the local variable y outside the function (as shown in the commented-out line), it will raise an error because local variables are not accessible outside their scope.

WARNING: IF YOU DON'T UNDERSTAND IT. IT'S OKAY. WE WILL DISCUSS THIS LATER IN LECTURES. THEN YOU WILL GET BETETR UNDERSTANDING OF IT.

This is the end for today's lecture

Allah Hafiz