

Empower Your Skills, Boost Your Productivity for a Prosperous Pakistan

Pak Angels Essential Generative AI Training Cohort-II

The session will start on 7:00 PM.



THE RISE OF GENERATIVE AI

A GAME - CHANGER



Module 3: Python For Beginners

- ✓ Seeing Google Colab and Hugging Face Interface
- ✓ What is Python? Why should we use it?
- ✓ Master variables, data types & operators in Python
- ✓ Make decisions with if-else statements in Python
- ✓ Automate tasks with loops (for & while) in Python
- ✓ Conquer lists, tuples, sets & dictionaries in Python
- ✓ Get a taste of problem-solving with LeetCode
- ✓ Questions ?



What is Python?

- ✓ **Python** is one of the most popular programming languages in the world.
- ✓ **Versatile:** It's used in web development, **data science**, **artificial intelligence**, scientific computing, automation, and more.
- ✓ **High Demand:** Many top tech companies like Google, Facebook, and Netflix use Python in their technology stacks.
- ✓ Python is **equipped with a rich ecosystem of libraries and frameworks**, such as TensorFlow, PyTorch, and Hugging Face's Transformers, which are essential for developing generative models like GPT-4, BERT, and StyleGAN.
- ✓ **Its simple and readable syntax** allows researchers and developers to easily implement and experiment with complex AI models.



Empower Your Skills, Boost Your Productivity for a Prosperous Pakistan

Thank You!





```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

Hello Word Code



```
#include <iostream>  
  
int main() {  
    std::cout << "Hello World" <<  
    std::endl;  
    return 0;  
}
```





Hello Word Code



```
print("Hello World")
```



```
console.log("Hello World");
```



Hello Word Code



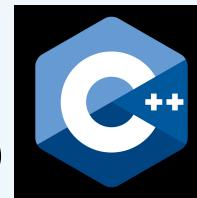
```
print("Hello World")
```



```
console.log("Hello World");
```



```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```



```
#include <iostream>  
  
int main() {  
    std::cout << "Hello World" <<  
    std::endl;  
    return 0;  
}
```

Empower Your Skills, Boost Your Productivity for a Prosperous Pakistan



Practical Demo Python

Muhammad
Talha



July – Sept, 2024 Gen-AI Training © 2024



Content

- ✓ Print Statement
- ✓ Comments
- ✓ Variables and their Data types
- ✓ Operators in Python
- ✓ Make decisions with if-else statements in Python
- ✓ Loops (for & while) in Python
- ✓ Conquer lists, tuples, sets & dictionaries in Python

Variables



Muhammad
Talha

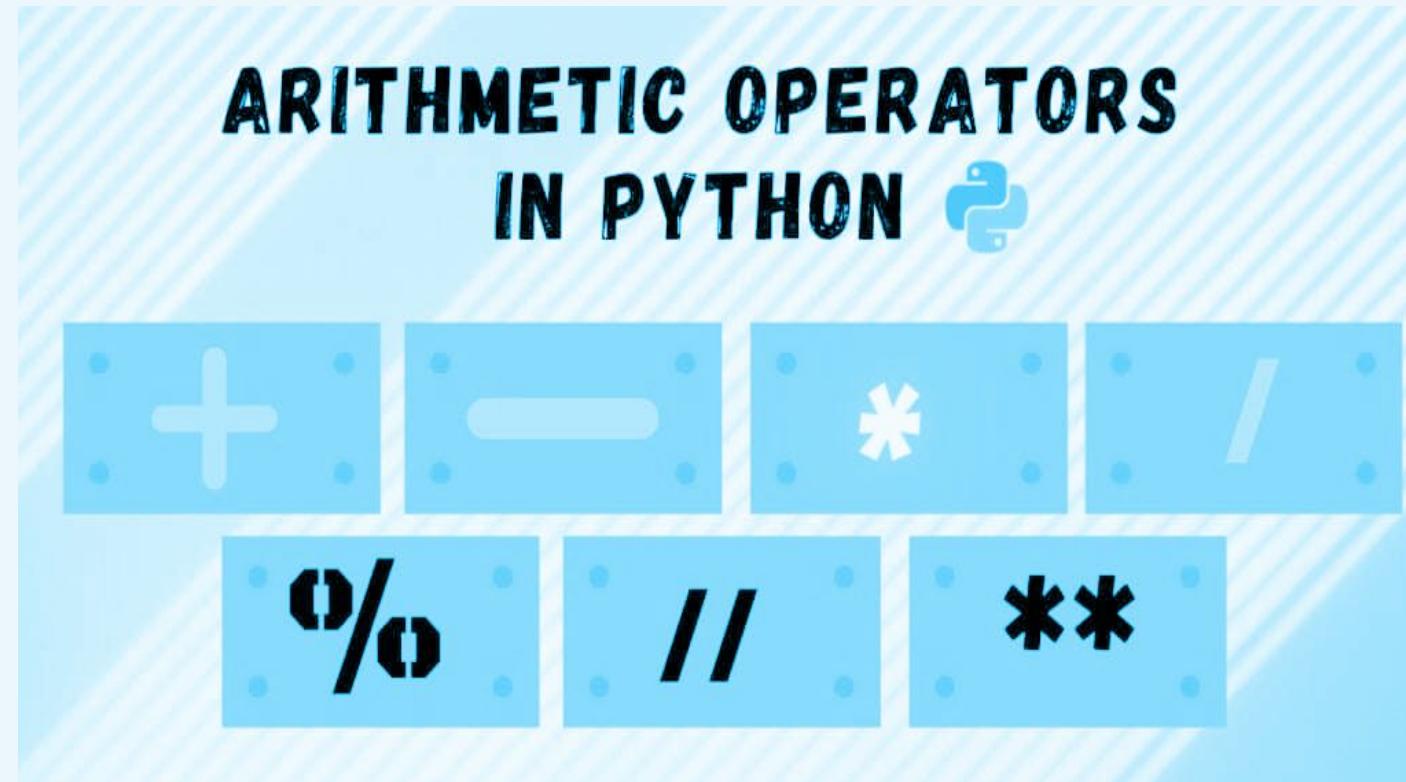


Rules For Naming Variable

- ✓ A variable name **must start with a letter or the underscore character _**
- ✓ There must be **no white space** in a variable name
- ✓ A variable name **cannot start with a number**
- ✓ A variable name can only contain **characters (A-Z, a-z, 0-9, and _)**. No special characters are allowed i.e. @icode , Python\$ are the invalid variables.
- ✓ Variable names are **case-sensitive** (age, Age and AGE are three different variables)
- ✓ **Keywords** cannot be used as variable names.



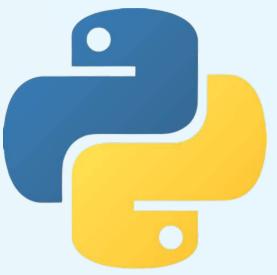
Operators





What Are Logical Operators In Python?

Operator	Description	Example
AND	Returns True if both operands are True	A and B
OR	Returns True if either of the operands are True	A or B
NOT	Retruns True if the operand in False	not A

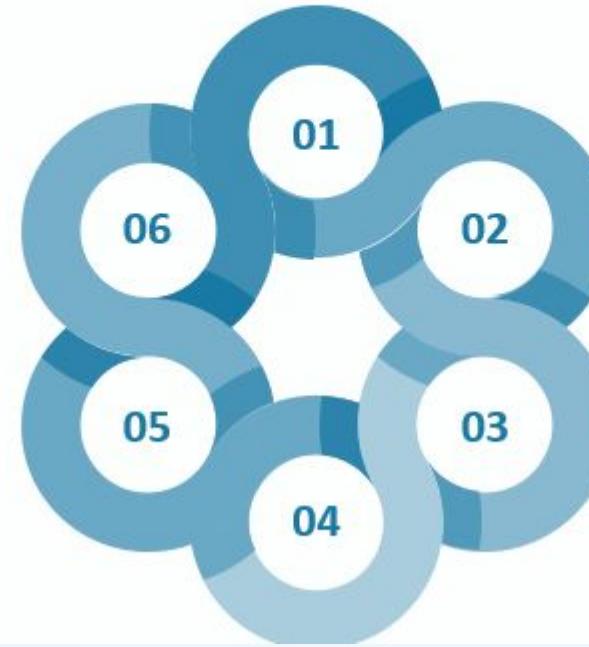


Python Comparison Operators

'<' Less Than

'<=' Less Than or Equal To

'!=' Not Equal To



Equal To '=='

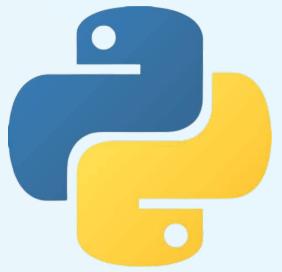
Greater Than or Equal to '>='

Greater Than '>'

educba.com



True



Conditional Statement (If - Elif - Else)

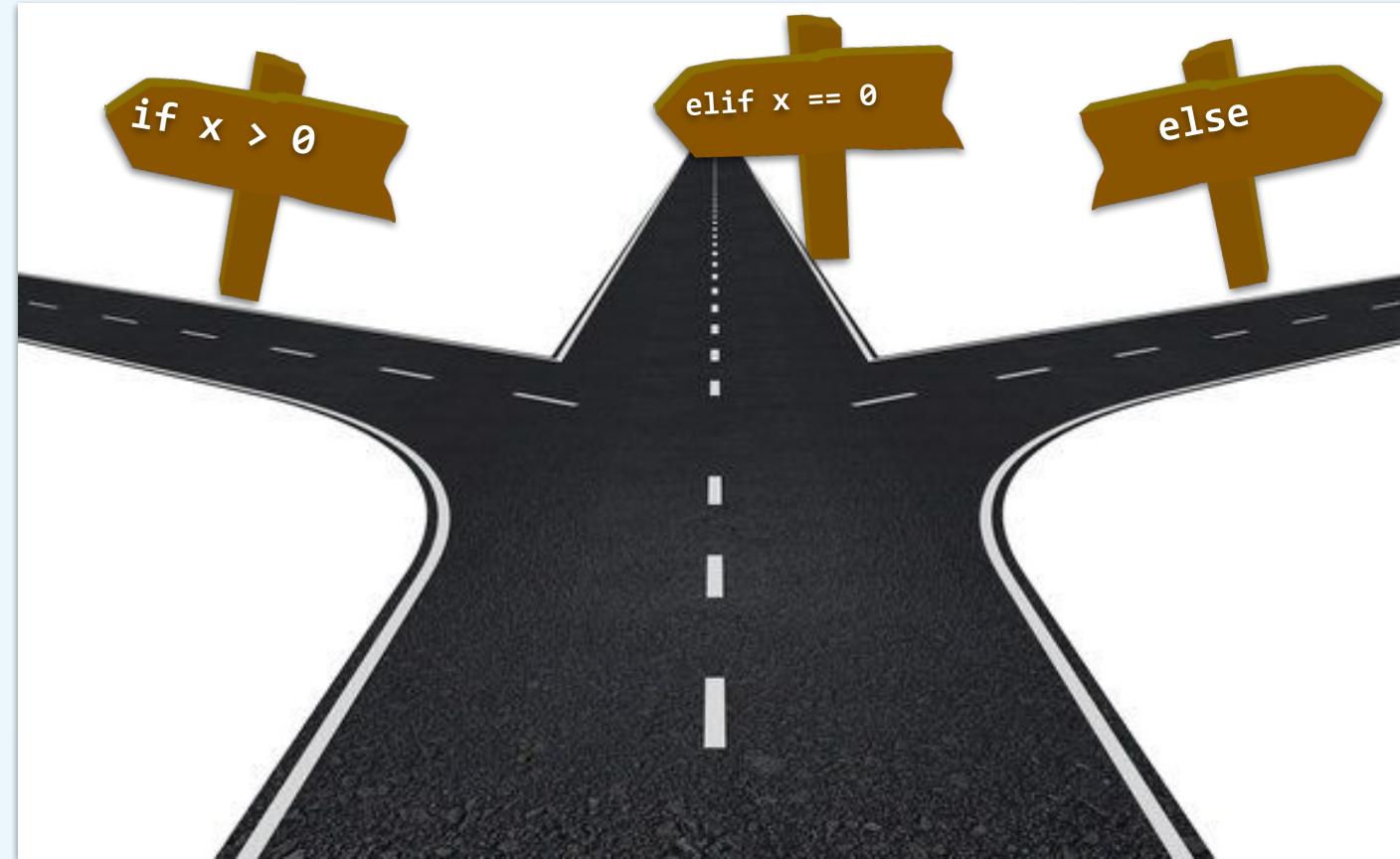
- ✓ If-Statements take a condition and only runs a block of code if the condition evaluates to **True**.
- ✓ Elif-Statements take a condition and only tests it if the prior If-Statement (and all prior Elif-Statements) evaluate to **False**. It is optional.
- ✓ Else-Statements take no condition and runs a block of code if the prior If-Statement (and Elif-Statements, if there are any) evaluate to **False**.

Still confused? Don't worry! We'll practice on these next slides! :)



Lets Run through an Example

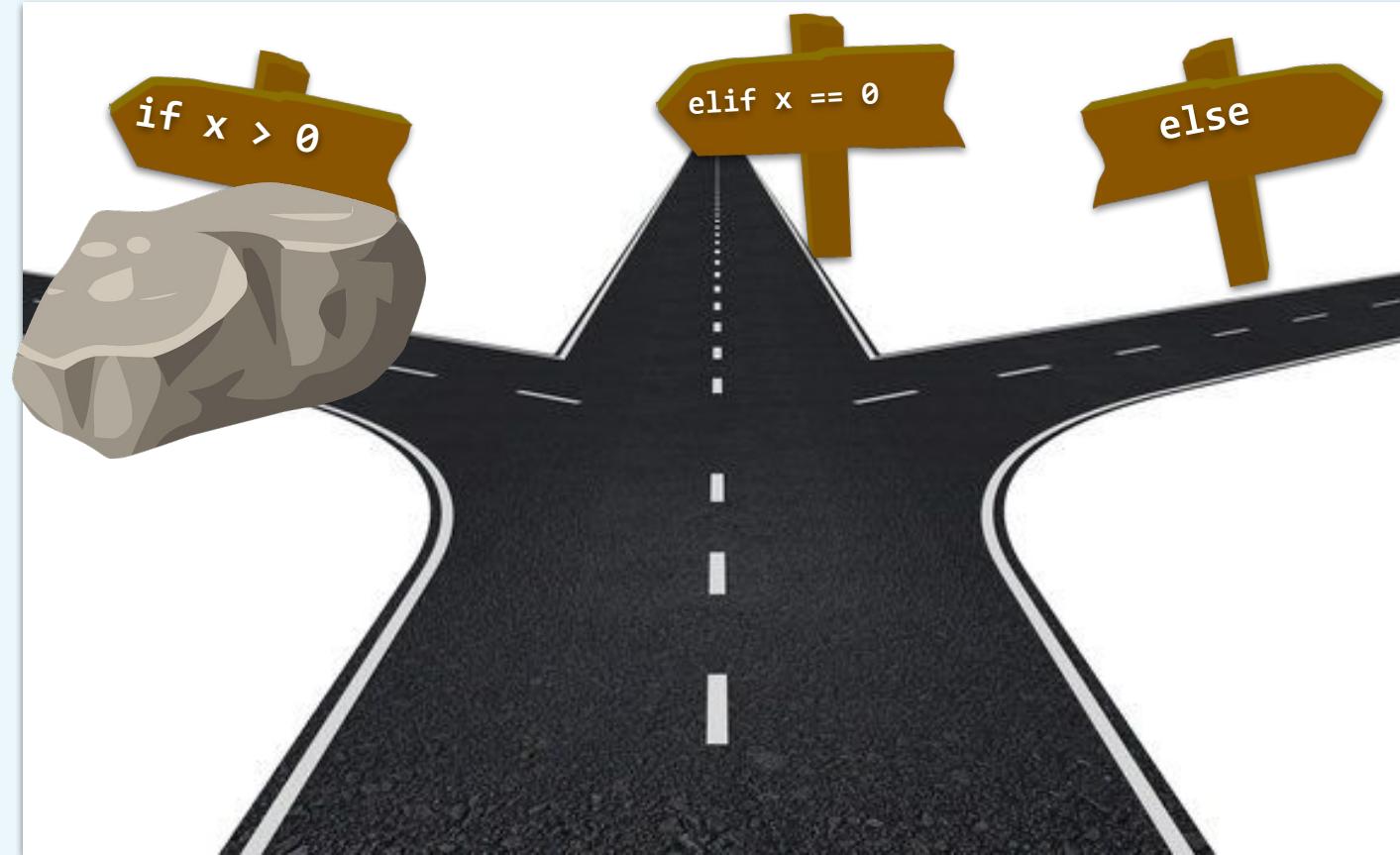
Suppose $x = -8$





Lets Run through an Example

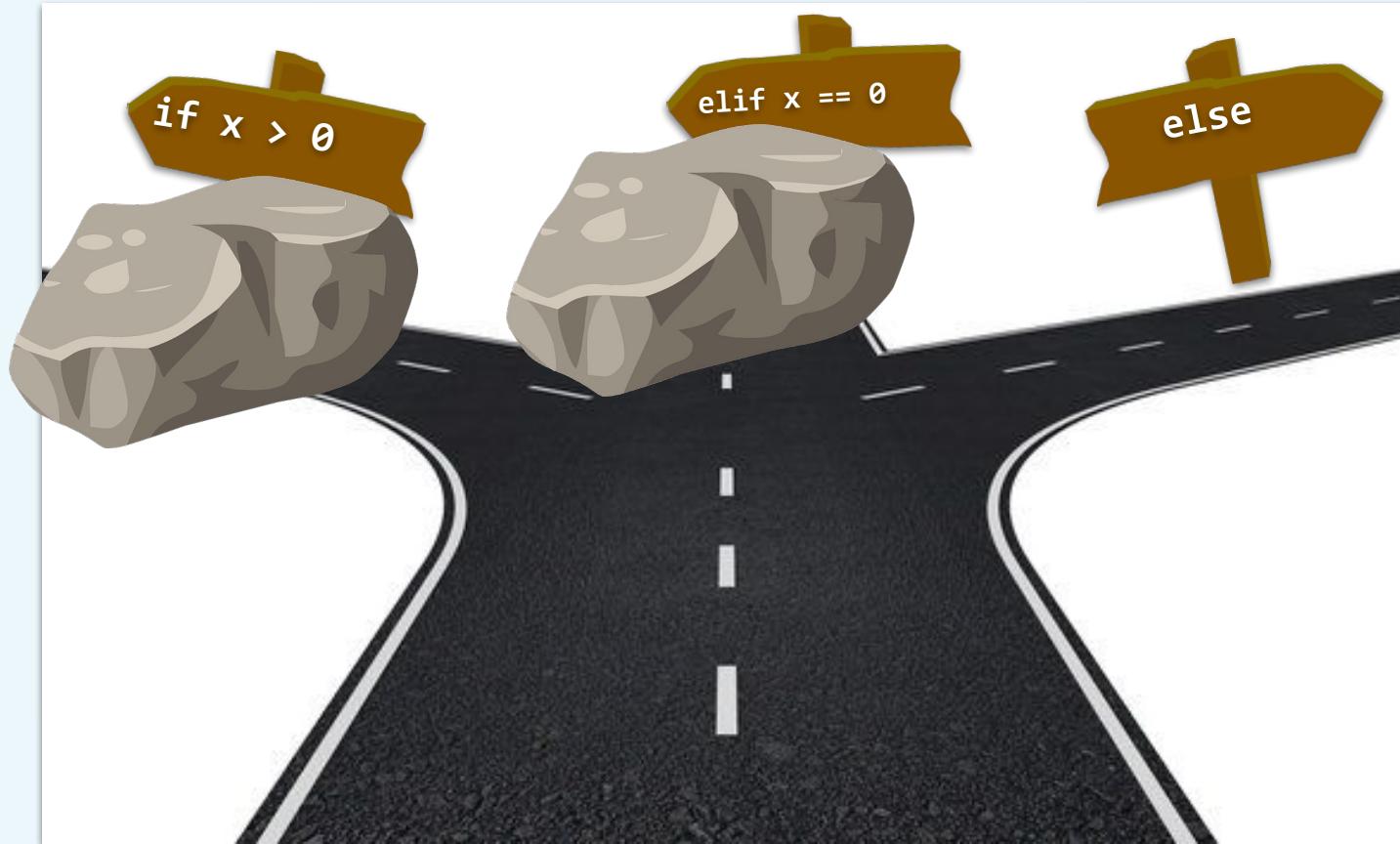
Suppose $x = -8$





Lets Run through an Example

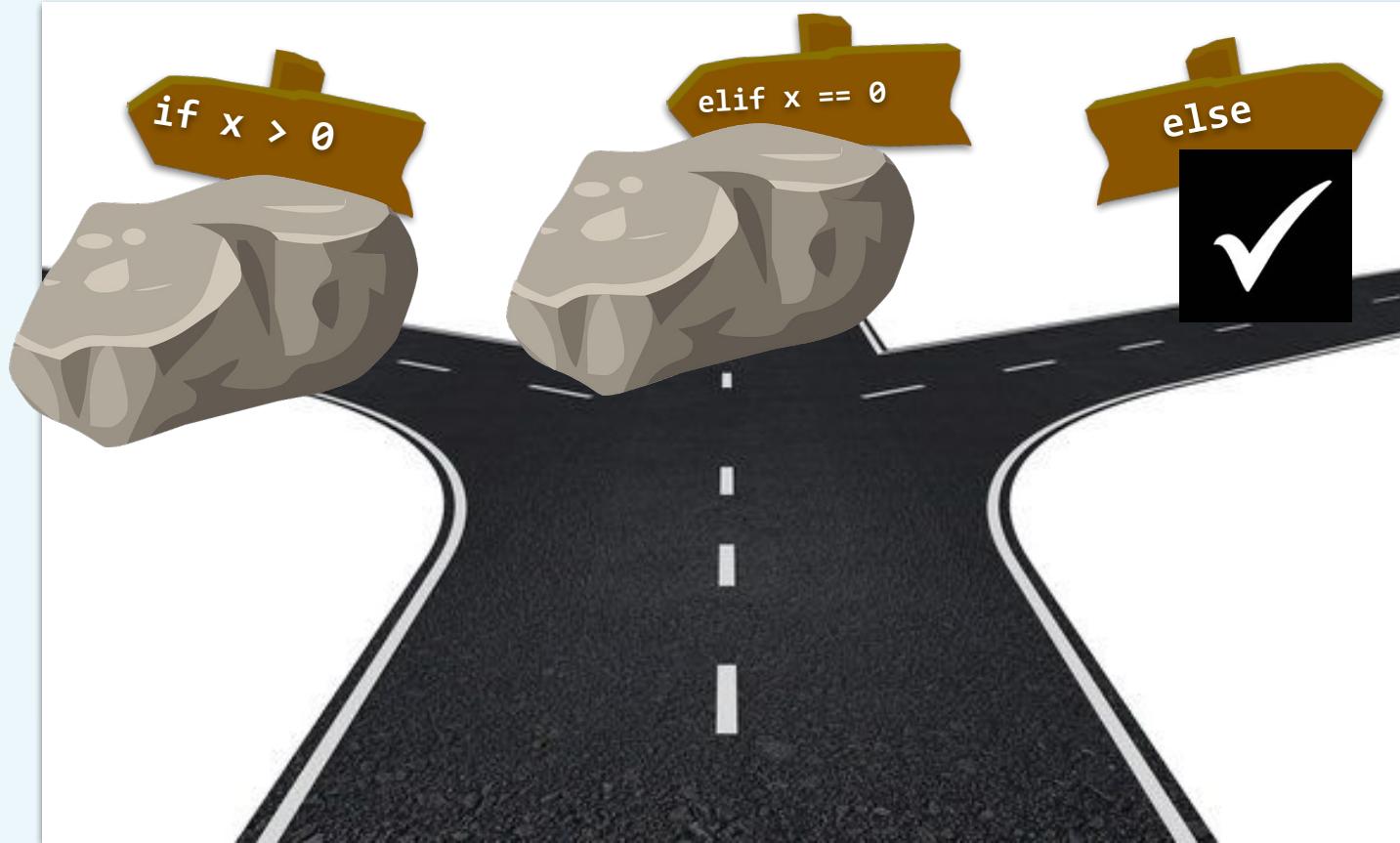
Suppose $x = -8$





Lets Run through an Example

Suppose $x = -8$

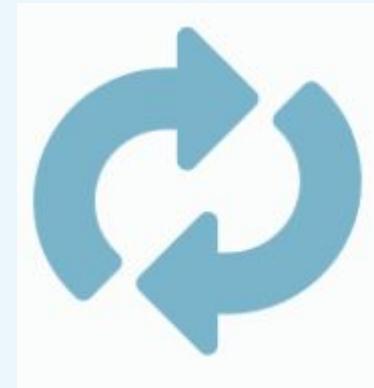




Loop Structures

- ✓ Used for Repetition of Task

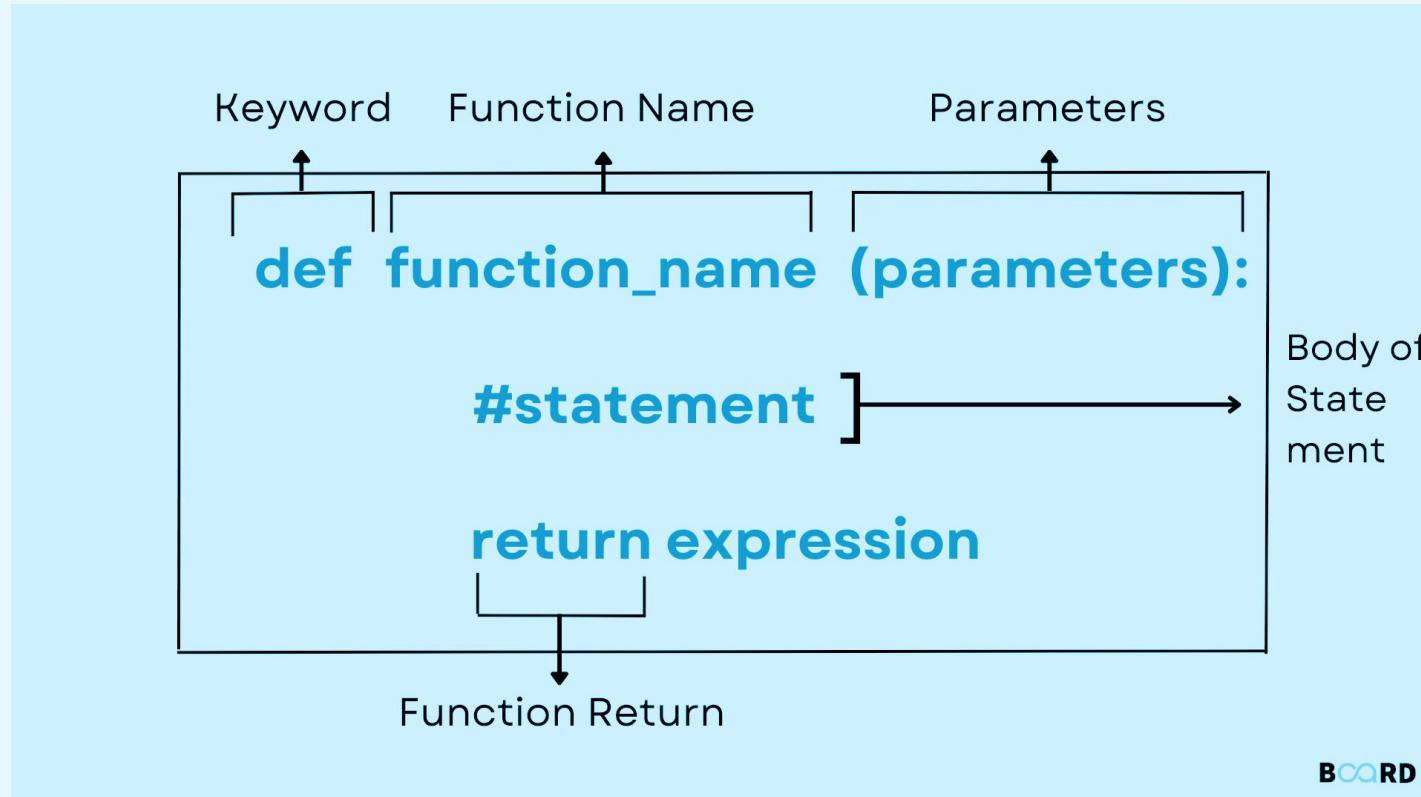
**For Loop
While Loop**

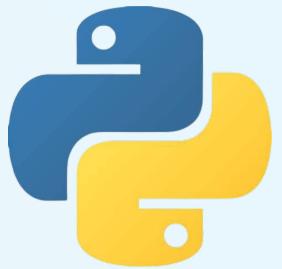




Functions

- ✓ Built-in Methods like max(), min(), math.sqrt()
- ✓ User Defined Methods





List []

It is used to store multiple Items in a single variable

List → ['Python', 'Swift', 'C++']

Index → 0 1 2

Neg Index -3 -2 -1



Tuple ()

A tuple is a collection **similar to List**.

The primary difference is that we **cannot modify a tuple** once it is created.

Tuple → ('Python', 'Swift', 'C++')

Index → 0 1 2

Neg Index -3 -2 -1



Set { }

A set is a **collection of unique data**, meaning that elements within a set cannot be duplicated.

Cannot Use Indexes to fetch items.

```
Set = { "stores", "useful", "things" }
```

✓ Unchangeable ✓ Unordered



Dictionary { }

A Dictionary is a **key-value** pair (consisting of a key and a value)
Keys are always unique.

```
country_capitals = {  
    'Germany' : 'Berlin',   ←----- element 1  
    'Canada' : 'Ottawa',   ←----- element 2  
    'England' : 'London'   ←----- element 3  
}  
          ↕           ↕  
        key         value
```



Let's Create a Simple Calculator



Let's Create a Simple Calculator

A screenshot of the Visual Studio Code (VS Code) interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and a search bar labeled "VS Code Front". The left sidebar features the Explorer, Search, and Problems sections, with the Problems section showing 6 items. The main workspace shows a file named "main.py" with the following code:

```
1
2 print("Hello World! Welcome to Esstential Generative AI Class")
```

The bottom right corner of the code editor has a status bar with "Ln 2, Col 64" and "Spaces: 4". Below the code editor is the Terminal panel, which displays the command "\$ python main.py" followed by the output "Hello World! Welcome to Esstential Generative AI Class". The bottom status bar also includes "UTF-8", "CRLF", "Python", "Select Interpreter", "CODEGPT", "Continue", and "Prettier".



The image shows a screenshot of the Visual Studio Code (VS Code) interface. The interface is divided into three main sections, each highlighted with a large, semi-transparent circular overlay containing a number:

- 1 Folders & Files**: This section is located on the left side of the screen, containing the Explorer and File Explorer panes. The Explorer pane shows a folder structure with 'VS CODE FRONT' expanded, revealing a file named 'main.py'. The File Explorer pane shows the same 'main.py' file.
- 2 Coding Area**: This section is located in the center of the screen, containing the main code editor area. It displays a Python script with two lines of code:

```
1 print("Hello World! Welcome to Esstential Generative AI Class")
```

A large blue VS Code logo is positioned in the top right corner of this area.
- 3 Output Area**: This section is located at the bottom of the screen, containing the Terminal, Output, Debug Console, and Problems panes. The Terminal pane shows command-line output from running 'main.py':

```
STONE@DESKTOP-E5A80N6 MINGW64 ~/Desktop/Programming/VS Code Front
$ python main.py
Hello World! Welcome to Esstential Generative AI Class

STONE@DESKTOP-E5A80N6 MINGW64 ~/Desktop/Programming/VS Code Front
$
```

The status bar at the bottom of the screen shows the current file is 'main.py', with line 2, column 64, and the file is saved in UTF-8 format. It also indicates the use of Python as the language, a selected interpreter, and options for Continue and Prettier.

Folders and File Area:

- **Purpose:** This is where you can see and navigate your files and folders. It's essentially your project's directory structure, allowing you to open and manage files.

1

Folders & Files

2

Coding Area

Coding Area:

- **Purpose:** This is where you write and edit your code.
- You can open multiple files in tabs and view your code in a clean, focused environment.

3

Output Area

Output Area:

- **Purpose:** This area shows additional information and tools like the Terminal (for running command-line commands), Output (for displaying output from tasks and extensions), Debug Console (for debugging output), and Problems (for listing issues in your code). It provides feedback and interaction for various tasks.



Create a Simple Calculator using Python and Streamlit for the Graphical Interface

Simple Calculator

Enter first number

 - +

Enter second number

 - +

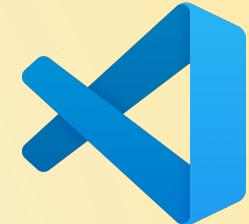
Select an operation

 ▼

Result: 0.55



Create a Simple Calculator



Simple Calculator

Enter first number

110.00

- +

Enter second number

200.00

- +

Select an operation

Division

▼

Result: 0.55



Streamlit
Streamlit (cal-pro.streamlit.app)

```
# Input fields for numbers
num1 = st.number_input("Enter first number")
num2 = st.number_input("Enter second number")

# Dropdown menu for operation selection
operation = st.selectbox("Choose an operation", ("Add", "Subtract", "Multiply", "Divide"))

# Perform the calculation based on the selected operation
if operation == "Add":
    result = num1 + num2
elif operation == "Subtract":
    result = num1 - num2
elif operation == "Multiply":
    result = num1 * num2
elif operation == "Divide":
    if num2 != 0:
        result = num1 / num2
    else:
        result = "Error! Division by zero."

# Display the result
st.write("Result:", result)
```

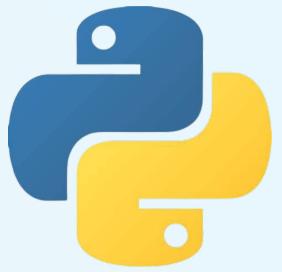


2010-080-01	2010-080-01
2010-080-02	2010-080-02
2010-080-03	2010-080-03
2010-080-04	2010-080-04
2010-080-05	2010-080-05

Empower Your Skills, Boost Your Productivity for a Prosperous Pakistan



Practical Demo GitHub



Development Environment Setup

What is GitHub?

- GitHub is a platform for hosting and sharing code using Git, a version control system.

What is Version Control?

- Version control helps manage changes to your codebase, allowing multiple people to work on a project simultaneously and track changes over time.

<https://code.visualstudio.com/download>

<https://github.com/>

<https://git-scm.com/downloads>



Importance of Version Control

Collaboration:

- Multiple developers can work on the same project without conflicts.

History and Tracking:

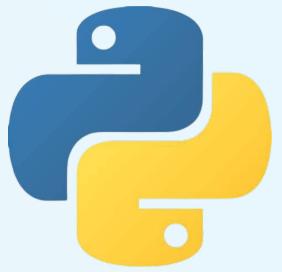
- **Track every change** made to the codebase and revert to previous versions if needed.

Backup and Recovery:

- Keeps a backup of your project on a remote server, **preventing data loss**.

Branching and Merging:

- Experiment with new features or fix bugs in isolated branches before merging them into the main codebase.



Step 1: Sign in to GitHub

Step 2: Create a New Repository

1. Once logged in, click on the "+" icon at the top right of the page and select "**New repository**" from the dropdown menu.

2. Alternatively, you can go to your profile page and click on the "**Repositories**" tab, then click the green "**New**" button.



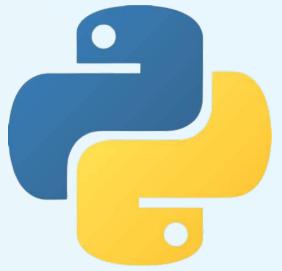
Step 3: Configure Your New Repository

- 1. Repository Name:** Enter a name for your repository. This name should be unique within your GitHub account.
- 2. Description (optional):**
- 3. Public/Private:**
- 4. Initialize this repository with a README:** This file usually contains information about your project.
- 5. Add .gitignore (optional):** Select a .gitignore template that matches the type of project you're creating. This file tells Git which files (or patterns) it should ignore.
- 6. Choose a license (optional):** If you want to, you can select a license for your project. This defines how others can use your project.



Step 4: Create Repository

1.click the green "Create repository" button.



Step 5: Push Code to the Repository (using Git)

1. Clone the Repository: Open a terminal (or Git Bash on Windows) and run the following commands:



Clone the repository to your local machine

git clone

<https://github.com/your-username/your-repository.git>

Change directory to your repository

cd your-repository

Add your files to the repository

git add .

Commit the changes

git commit -m "Initial commit"

Push the changes to GitHub

git push origin main

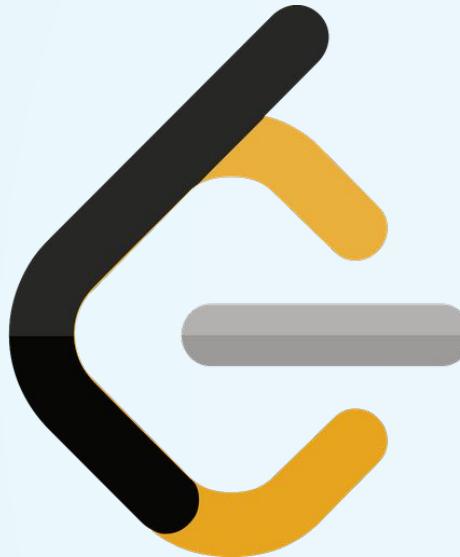
Replace `your-username` and `your-repository` with your GitHub username and the repository name you chose.



And that's it!

You've successfully created a GitHub repository and
pushed your first code to it.

Empower Your Skills, Boost Your Productivity for a Prosperous Pakistan



LeetCode



Benefits of Using LeetCode

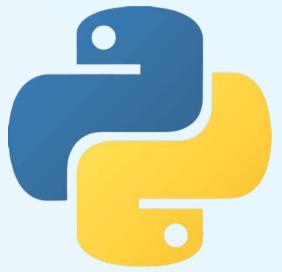
Title: Benefits of LeetCode for Career Development

1. Skill Enhancement

1. Strengthens problem-solving skills
2. Deepens understanding of algorithms and data structures

2. Interview Preparation

1. Real-world interview questions from top tech companies



Let's take a little Taste of **Problem Solving**

- Contains Duplicate
- Palindrome Number

<https://leetcode.com/>



Let's take a little Taste of Problem Solving

Contains Duplicate

<https://leetcode.com/>



LeetCode

217. Contains Duplicate

Easy

Topics

Companies

Given an integer array `nums`, return `true` if any value appears **at least twice** in the array, and return `false` if every element is distinct.



Contains Duplicate

`nums = [1, 2, 3, 1]`



LeetCode



Example 1:

Input: nums = [1,2,3,1]
Output: true

Example 2:

Input: nums = [1,2,3,4]
Output: false

Example 3:

Input: nums = [1,1,1,3,3,4,3,2,4,2]
Output: true



LeetCode

Contains Duplicate

Approach 1: Brute Force

nums = [1, 2, 3, 1]

The brute force approach compares each element with every other element in the array to check for duplicates. If any duplicates are found, it returns `true`. This approach is straightforward but has a time complexity of $O(n^2)$, making it less efficient for large arrays.



LeetCode

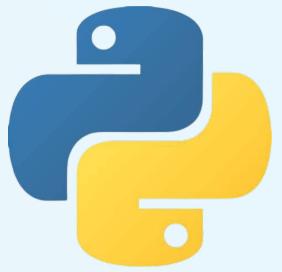
```
class Solution:

    def containsDuplicate(self, nums: List[int]) -> bool:
        n = len(nums)
        for i in range(n - 1):
            for j in range(i + 1, n):
                if nums[i] == nums[j]:
                    return True
        return False
```

nums = [1, 2, 3, 1] ↓↓



LeetCode



Contains Duplicate

Approach 2: Sorting

nums = [1, 2, 3, 1] ↓ ↓

The sorting approach sorts the array in ascending order and then checks for adjacent elements that are the same. If any duplicates are found, it returns `true`. Sorting helps in bringing duplicates together, simplifying the check. However, sorting has a time complexity of $O(n \log n)$.



LeetCode

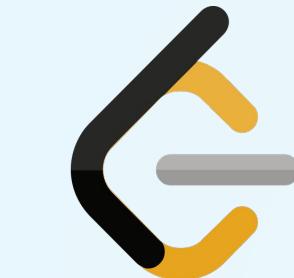


Contains Duplicate

Approach 2: Sorting

```
class Solution:
    def containsDuplicate(self, nums: List[int]) -> bool:
        nums.sort()
        n = len(nums)
        for i in range(1, n):
            if nums[i] == nums[i - 1]:
                return True
        return False
```

nums = [1, 2, 3, 1] ↓ ↓



LeetCode



Contains Duplicate

Approach 3: Hash Set

nums = [1, 2, 3, 1] ↓ ↓

The hash set approach uses a hash set data structure to store encountered elements. It iterates through the array, checking if an element is already in the set. If so, it returns `true`. Otherwise, it adds the element to the set. This approach has a time complexity of $O(n)$ and provides an efficient way to check for duplicates.



LeetCode



Contains Duplicate

Approach 3: Hash Set

nums = [1, 2, 3, 1] ↓ ↓

```
class Solution:

    def containsDuplicate(self, nums: List[int]) -> bool:
        seen = set()
        for num in nums:
            if num in seen:
                return True
            seen.add(num)
        return False
```



LeetCode

Contains Duplicate

Approach 4: Hash Map

nums = [1, 2, 3, 1] ↓ ↓

The hash map approach is similar to the hash set approach but also keeps track of the count of occurrences for each element. It uses a hash map to store the elements as keys and their counts as values. If a duplicate element is encountered (count greater than or equal to 1), it returns **true**. This approach provides more information than just the presence of duplicates and has a time complexity of $O(n)$.



LeetCode



Contains Duplicate

Approach 4: Hash Map

nums = [1, 2, 3, 1] ↓ ↓

In this approach, we iterate through the array and store each element as a key in a hash map. The value associated with each key represents the count of occurrences of that element. If we encounter an element that already exists in the hash map with a count greater than or equal to 1, we return `true`, indicating that a duplicate has been found. Otherwise, we update the count of that element in the hash map. If we complete the iteration without finding any duplicates, we return `false`.



LeetCode



Contains Duplicate

Approach 4: Hash Map

```
class Solution:
    def containsDuplicate(self, nums: List[int]) -> bool:
        seen = {}
        for num in nums:
            if num in seen and seen[num] >= 1:
                return True
            seen[num] = seen.get(num, 0) + 1
        return False
```

nums = [1, 2, 3, 1] ↓↓



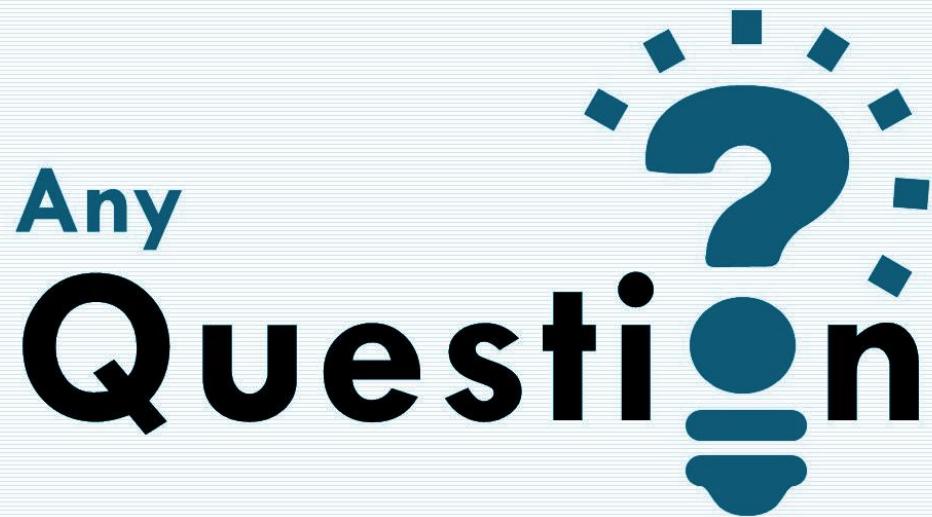
THANK YOU!

Regards,
Zulfiqar Ali Mir &
Muhammad Talha

LinkedIn:

<https://www.linkedin.com/in/zulfiqar-ali-mir/>

<https://www.linkedin.com/in/muhammmad-talha/>



LinkedIn:

<https://www.linkedin.com/in/zulfiqar-ali-mir/>

<https://www.linkedin.com/in/muhammad-talha/>

Regards,
Zulfiqar Ali Mir &
Muhammad Talha