Learn Python Programming From Zero to Hero

https://www.linkedin.com/in/srivastavapranjal/

First Milestone

Introduction to Python, Variables, Operators, Boolean Expression, Conditional Statement, Loops

Second Milestone

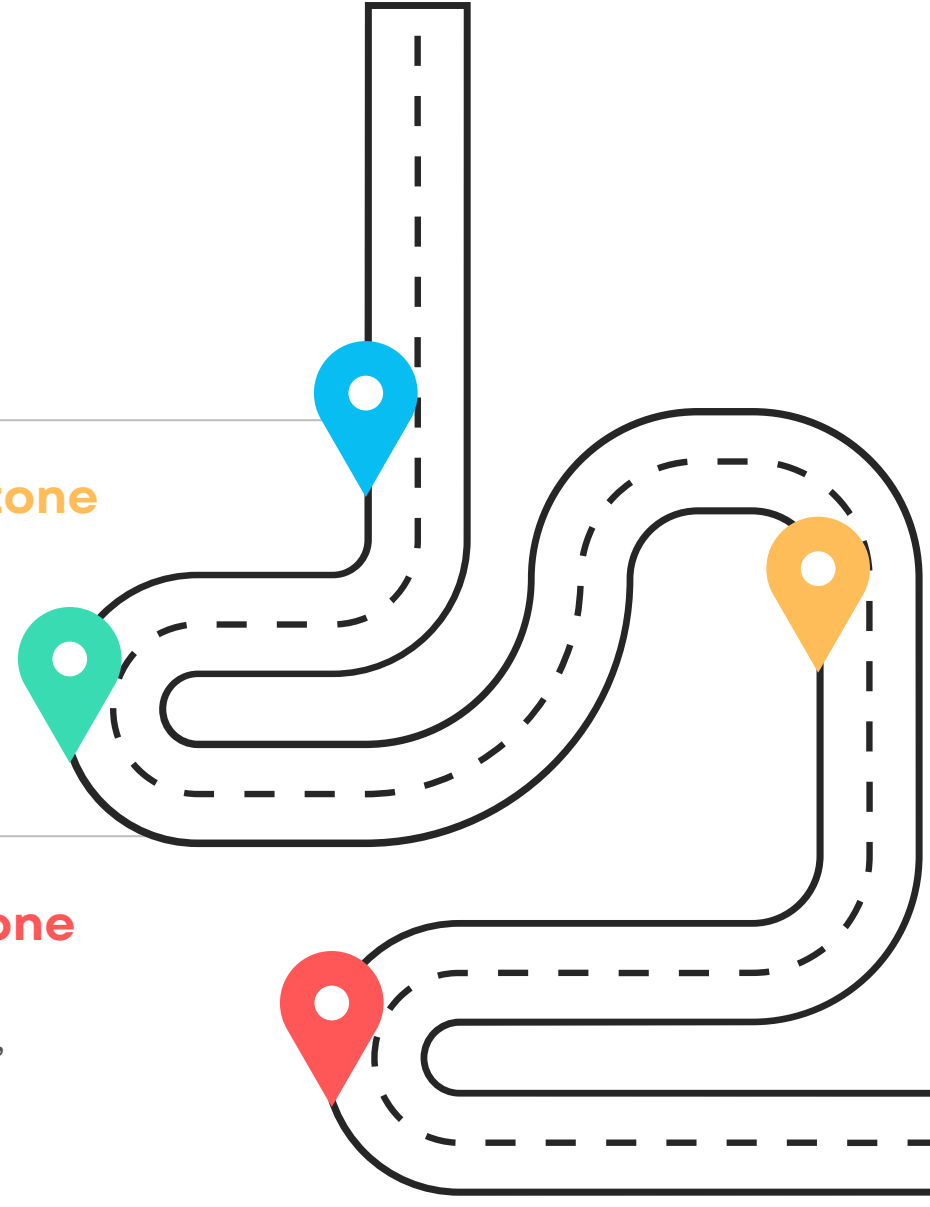
Python Datatypes, List, String, Tuples, Numbers, Sets, Dictionary

Third Milestone

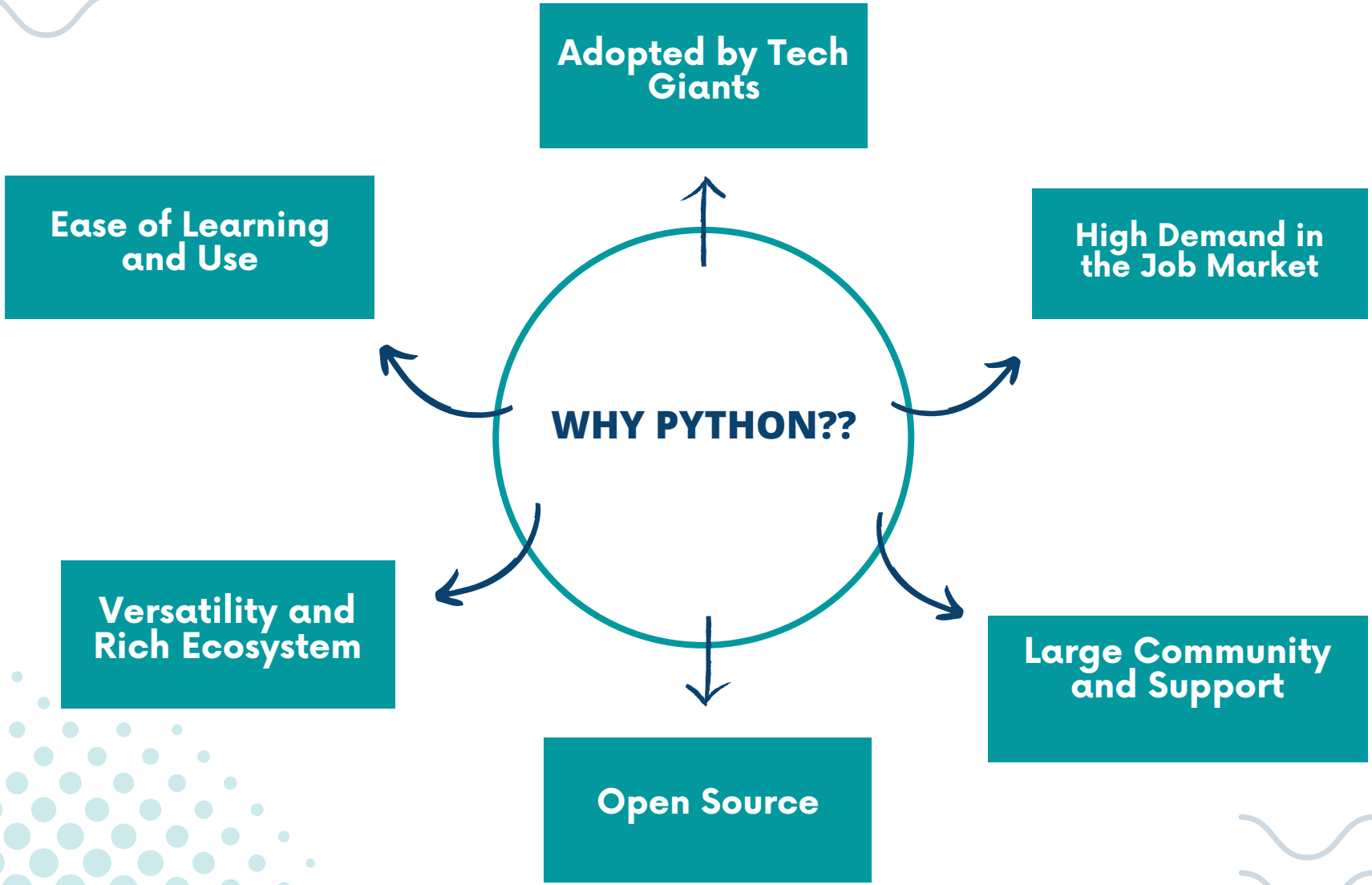
Functions, Object Oriented Programming, Comprehension, Modules, Packages

Fourth Milestone

File Handling, Exception Handling, OS, DateTime

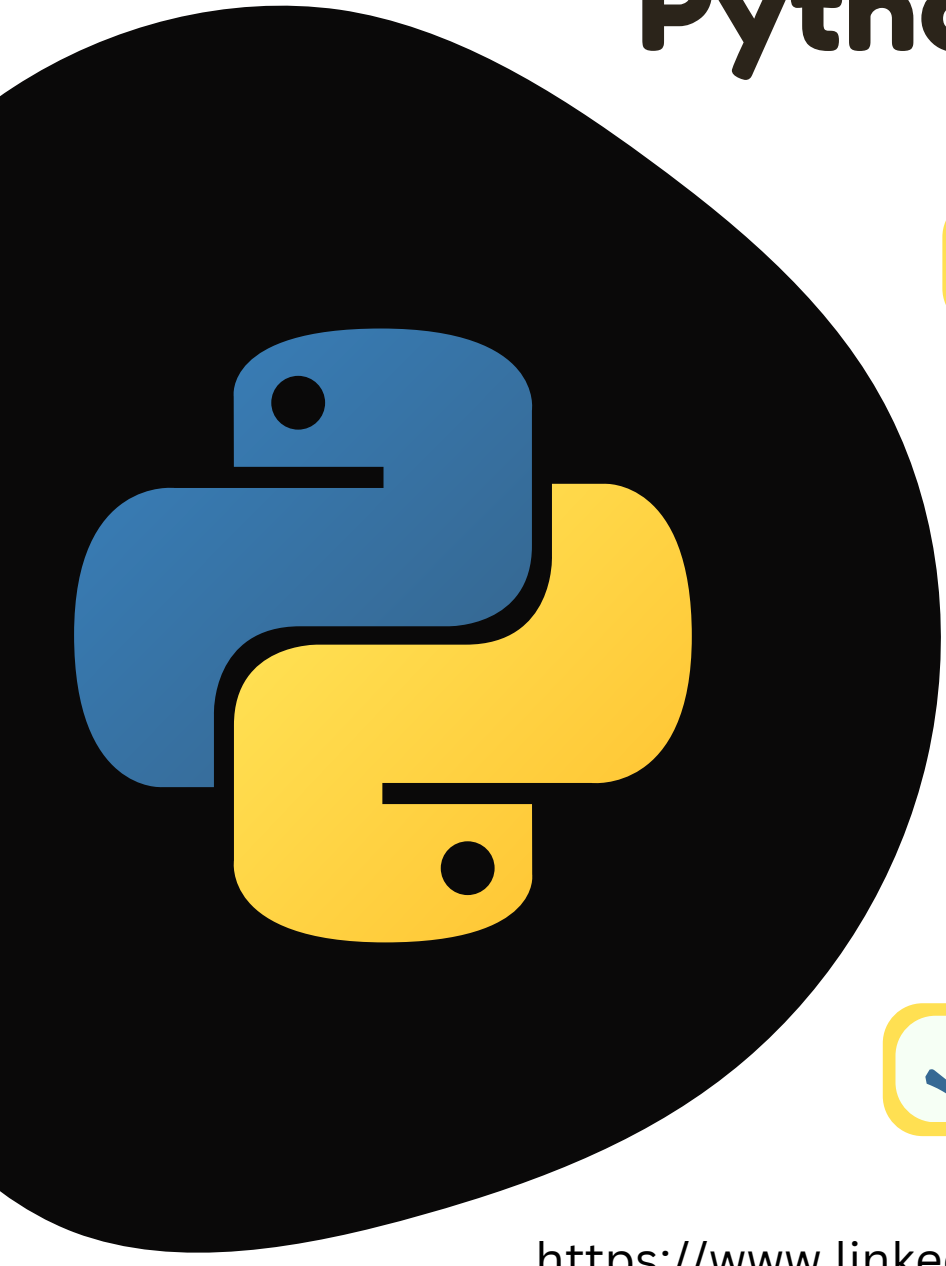


https://www.linkedin.com/in/srivastavapranjal/



https://www.linkedin.com/in/srivastavapranjal/

Python is



- High-Level Language
- Interpreted Language
- OOP Language
- General-Purpose Language
- My fav Programming language

https://www.linkedin.com/in/srivastavapranjal/

Important Links

https://www.python.org/

https://code.visualstudio.com

https://www.jetbrains.com/pycharm/

https://www.linkedin.com/in/srivastavapranjal/

Gets started!

https://www.linkedin.com/in/srivastavapranjal/

Python Variables

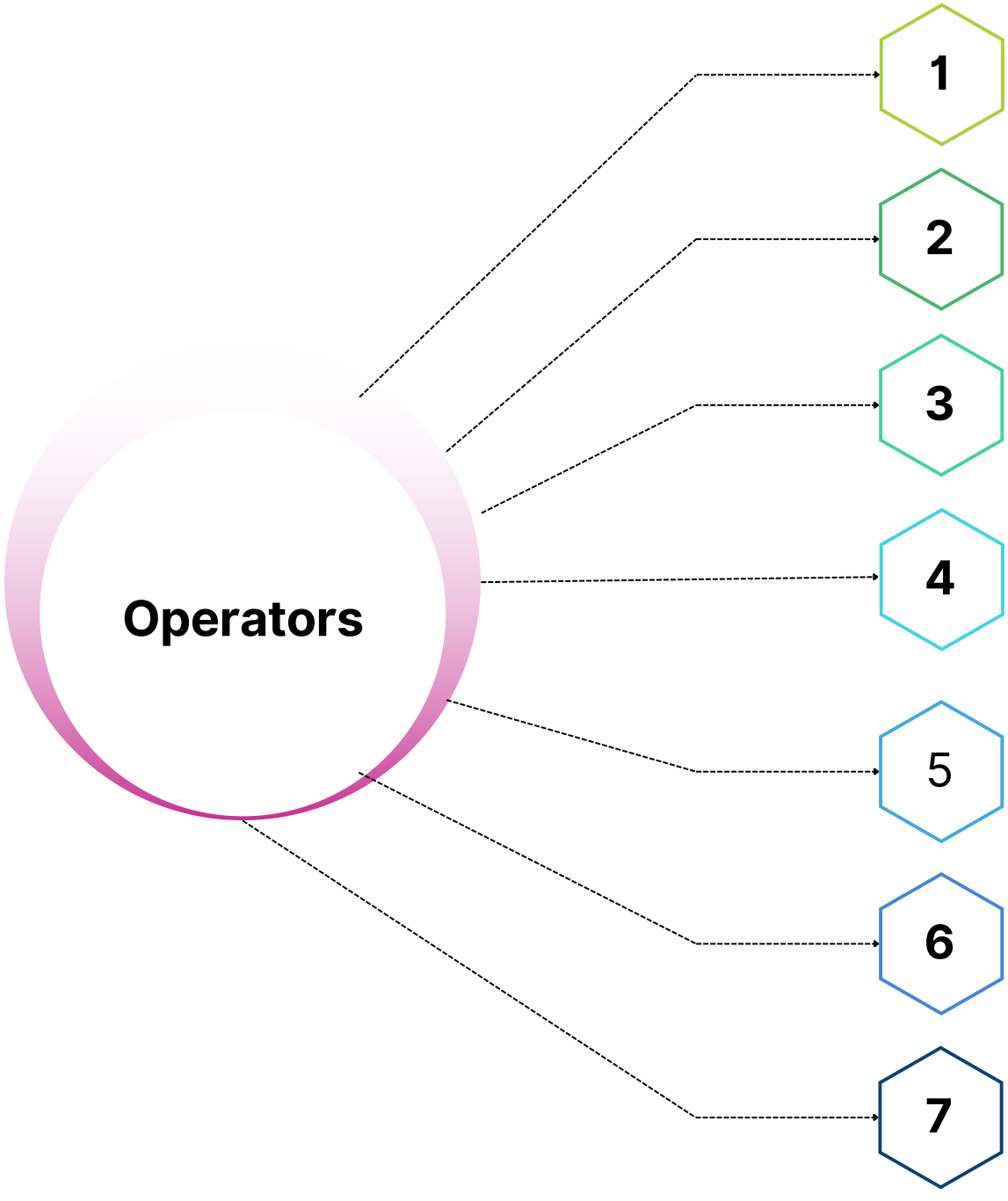
- Variables in Python are used to store data or values.
- They are created when you assign a value to a name using the assignment operator "=".
- Python is dynamically typed, so variables can hold values of any data type without explicit declaration.
- The data type of a variable is determined based on the value assigned to it.
- Variables can be reassigned to new values, changing their content during the program's execution.

https://www.linkedin.com/in/srivastavapranjal/

Rules for Python Variables

- A Python variable name must start with a letter or the underscore character.
- A Python variable name cannot start with a number.
- A Python variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _).
- Variable in Python names are case-sensitive (name, Name, and NAME are three different variables).
- The reserved words(keywords) in Python cannot be used to name the variable in Python.

https://www.linkedin.com/in/srivastavapranjal/



ARITHMETIC OPERATORS

Addition (+), Subtraction (-), Multiplication (*), Division (/), Floor Division (//), Modulo (%), Exponentiation (**)

COMPARISON OPERATORS

Equal to (==), Not equal to (!=), Greater than (>), Less than (<), Greater than or equal to (>=), Less than or equal to (<=)

LOGICAL OPERATORS

Logical AND (and), Logical OR (or), Logical NOT (not)

ASSIGNMENT OPERATORS

Assign (=), Add and assign (+=), Subtract and assign (-=), Multiply and assign (*=), Divide and assign (/=), Modulo and assign (%=), Floor divide and assign (//=), Exponentiation and assign (**=)

IDENTITY OPERATORS

is, is not

MEMBERSHIP OPERATORS

in, not in

BITWISE OPERATORS

Bitwise AND (&), Bitwise OR (|), Bitwise XOR (^), Bitwise NOT (~), Left Shift (<<), Right Shift (>>)

Conditional Statements

- Conditional statements in Python are used to control the flow of a program based on certain conditions.
- They allow you to execute different blocks of code depending on whether a condition is true or false.
- Python provides two main conditional statements: "if" and "else." Additionally, you can use "elif" (short for "else if") to handle multiple conditions.

Python Loops

- Loops are used to repeatedly execute a block of code until a specific condition is met.
- They help automate repetitive tasks and simplify code that needs to iterate over a sequence or perform actions based on certain conditions
- The "for" loop is used to iterate over a sequence, such as a list, tuple, string, or other iterable objects. It executes a block of code for each element in the sequence.
- The "while" loop continues to execute a block of code as long as a specified condition remains true.

https://www.linkedin.com/in/srivastavapranjal/

First Milestone

- Introduction to Python
- Advantages of Python over other programming languages
- Python Variables
- Operators and Boolean expressions
- Conditional statements- If, else, and elif statements, Nested-if
- Loops- while and for loops, nested loops, break, pass and continue keywords

Python Datatypes

- Data types are used to categorize and represent different types of data.
- Python is a dynamically typed language, meaning the data type of a variable is determined at runtime based on the value assigned to it.

https://www.linkedin.com/in/srivastavapranjal/

Different types of Python Datatypes

Numeric Types:

- int: Integer data type represents whole numbers, positive or negative, without any fractional parts.
- float: Floating-point data type represents numbers with decimal points or exponential notation.

Sequence Types:

- str: String data type represents a sequence of characters enclosed within single (' '), double (" "), or triple (''' ''') quotes.
- list: List data type represents an ordered collection of elements, and it is mutable (can be changed).
- tuple: Tuple data type is similar to a list but is immutable (cannot be changed once created).

https://www.linkedin.com/in/srivastavapranjal/

Boolean Type:

- bool: Boolean data type represents True or False values. It is used for logical operations and conditional expressions.

Set Types:

- set: Set data type represents an unordered collection of unique elements. Duplicate elements are automatically removed.
- frozenset: Frozenset data type is similar to a set but is immutable.

Mapping Type:

- dict: Dictionary data type represents a collection of key-value pairs. Each key is associated with a value, and it is mutable.

None Type:

- None: The None data type represents the absence of a value. It is often used to indicate that a variable has not been assigned a value.

https://www.linkedin.com/in/srivastavapranjal/

Second Milestone

- Python Datatypes
- Lists- Creating, Indexing, Slicing, and List methods
- Tuples
- Strings- String concatenation, Indexing, Slicing, and String methods
- Sets and set operations
- Python dictionary- Creating, Accessing, Modifying, and Dictionary methods
- Random modules

https://www.linkedin.com/in/srivastavapranjal/

Python Functions

- Functions are blocks of organized, reusable code that perform a specific task or a set of tasks.
- They allow you to break down a program into smaller, more manageable pieces, making your code modular, easier to read, and maintainable.
- Functions can take input arguments (parameters) and return output values.

https://www.linkedin.com/in/srivastavapranjal/

Object Oriented Programming

- Object-Oriented Programming (OOP) is a programming paradigm that focuses on organizing code into objects, each representing a real-world entity or concept.
- Python is an object-oriented programming language that supports the creation and use of classes and objects.
- OOP provides several concepts, including encapsulation, inheritance, polymorphism, and abstraction, which allow for more organized, maintainable, and reusable code.

https://www.linkedin.com/in/srivastavapranjal/

Classes and Objects

- A class is a blueprint or template that defines the structure and behavior of objects. It serves as a blueprint for creating objects with specific properties and methods.

- An object is an instance of a class. It is a tangible entity that holds data and can perform actions based on the class's defined behavior.

Encapsulation

- Encapsulation refers to the bundling of data and methods that operate on that data within a single unit (i.e., the class).

- The internal state of the object is hidden from the outside, and access to it is controlled through methods.

Inheritance

- Inheritance allows a class (subclass) to inherit the attributes and behaviors of another class (superclass).

- Subclasses can extend or override the functionality of the superclass, promoting code reuse and supporting the "is-a" relationship.

Polymorphism

- Polymorphism allows objects of different classes to be treated as objects of a common superclass.

- It enables code to work with objects at a higher level of abstraction, promoting flexibility and extensibility.

Abstraction

- Abstraction refers to the concept of hiding unnecessary details and exposing only essential features to the user.

- It allows you to build complex systems without needing to understand the internal implementation of each class.

https://www.linkedin.com/in/srivastavapranjal/

Constructor and Destructor

- The constructor (__init__ method) is a special method that initializes the object's attributes when an instance is created.

- The destructor (__del__ method) is a special method that is called when an object is about to be destroyed and cleaned up.

Lambda Functions

- A lambda function is a small anonymous function.
- It is a way to create simple, one-line functions without explicitly defining a function using the def keyword.
- Lambda functions are typically used when you need a simple function for a short period and don't want to define a full-fledged function using def.

Expressions:

lambda arguments: expression

https://www.linkedin.com/in/srivastavapranjal/

Python Modules

- A module is a file containing Python definitions and statements. It allows you to organize and reuse code by breaking it into separate files.
- A module can contain functions, classes, variables, and other Python objects that can be imported into other Python scripts.
- Modules help in keeping code clean and maintainable by promoting code reusability.
- They allow you to divide large programs into smaller, manageable pieces, making it easier to work collaboratively and manage complex projects effectively.

https://www.linkedin.com/in/srivastavapranjal/

Python Packages

- A package is a collection of modules organized in a directory hierarchy.
- A package is like a directory that contains multiple Python module files, and it allows you to organize related modules together for better code organization and reusability.
- Packages provide a way to create a hierarchical structure for your Python project.

https://www.linkedin.com/in/srivastavapranjal/

```
my_package/
    __init__.py
    module1.py
    module2.py
    subpackage1/
        __init__.py
        module3.py
    subpackage2/
        __init__.py
        module4.py
```

List Comprehension

- List comprehension is a concise and expressive way to create new lists in Python.
- It allows you to generate a new list by applying an expression to each element of an existing iterable (e.g., list, tuple, string) while filtering the elements based on specified conditions.
- List comprehension provides a more readable and efficient alternative to traditional for loops when creating lists.

new_list = [expression for item in iterable if condition]

https://www.linkedin.com/in/srivastavapranjal/

Third Milestone

- Functions in Python
- Object-Oriented Programming (OOP) in Python
- Lambda keyword, List Comprehension
- Modules and Packages

File Handling

- File handling in Python allows you to work with files on your computer's file system. It enables you to read data from files, write data to files, and perform various file-related operations.
- Python provides built-in functions and methods to perform file handling tasks easily.
- There are three main file handling modes:

1. Read mode ("r"): Used to read data from a file. The file must exist, or an error will occur.
2. Write mode ("w"): Used to write data to a file. If the file exists, it will be truncated (emptied). If it doesn't exist, a new file will be created.
3. Append mode ("a"): Used to append data to the end of an existing file. If the file doesn't exist, a new file will be created.

https://www.linkedin.com/in/srivastavapranjal/

File Handling

- open(): Opens a file and returns a file object that can be used for reading, writing, or both.
- close(): Closes the file and releases the associated system resources.
- read(): Reads the entire contents of a file as a string.
- readline(): Reads a single line from the file.
- readlines(): Reads all lines from the file and returns them as a list.
- write(): Writes a string to the file.
- writelines(): Writes a list of strings to the file.
- seek(): Sets the file's current position to the given offset.
- tell(): Returns the current position of the file.

https://www.linkedin.com/in/srivastavapranjal/

Exception Handling

- Exception handling in Python allows you to gracefully handle runtime errors or exceptional situations that may occur during program execution.

- Instead of letting the program crash, you can catch and handle these exceptions, allowing the program to continue running or take appropriate actions when something unexpected occurs.

https://www.linkedin.com/in/srivastavapranjal/

Exception Handling

- The main construct used for exception handling in Python is the try-except block. The **try** block contains the code that might raise an exception, and the **except** block contains the code to handle the exception.
- The **else** block, if present, is executed when no exceptions occur within the **try** block. It is useful for code that should run only when there are no exceptions.
- The **finally** block, if present, is executed regardless of whether an exception occurred or not. It is commonly used for cleanup operations, such as closing files or releasing resources.

https://www.linkedin.com/in/srivastavapranjal/

Regular Expressions

- []: A set of characters, Example:"[a-m]"
- \: Signals a special sequence, Example: "\d"
- .: Any character (except newline character), Example: "he..o"
- ^: Starts with, Example: "^hello"
- $: Ends with, Example: "planet$"
- *: Zero or more occurrences, Example: "he.*o"
- +: One or more occurrences, Example: "he.+o"
- ?: Zero or one occurrences, Example: "he.?o"
- {}: Exactly the specified number of occurrences, Example: "he.{2}o"
- |: Either or, Example: "falls|stays"

https://www.linkedin.com/in/srivastavapranjal/

Regular Expressions

- [arn]: Returns a match where one of the specified characters (a, r, or n) is present
- [a-n]: Returns a match for any lower case character, alphabetically between a and n
- [^arn]: Returns a match for any character EXCEPT a, r, and n
- [0123]: Returns a match where any of the specified digits (0, 1, 2, or 3) are present
- [0-9]: Returns a match for any digit between 0 and 9
- [0-5][0-9]: Returns a match for any two-digit numbers from 00 and 59
- [a-zA-Z]: Returns a match for any character alphabetically between a and z, lower case OR upper case

https://www.linkedin.com/in/srivastavapranjal/

Fourth Milestone

- File handling in python
- Files input/output functions
- Exception Handling
- OS, DateTime modules
- Regular expressions


Thank
you!