Python for School Students – Learn to Code Creatively

This document outlines the syllabus for a Python programming course designed for students in grades 6th through 10th. The course aims to introduce fundamental coding concepts and logic building skills, enabling students to create fun, interactive Python programs. The curriculum covers topics from basic syntax and data types to conditionals, loops, functions, data structures, and file handling (optional for older students). Through hands-on activities and mini-projects, students will gain practical experience and develop the ability to read and write small programs independently. Upon completion, participants will receive a certification acknowledging their newly acquired Python programming skills.

Python Programming Skills Pyramid

Course Details

Course Name: Python for School Students – Learn to Code Creatively

Target Audience: Class 6th to 12th Students

Duration: 20 to 30 hours

Mode: Online / Offline

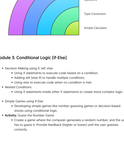
Fee: ₹ 15,000

Module-wise Syllabus

Module 1: Introduction to Computers & Python

- What is Programming?
  - Explanation of what programming is and its importance in today's world
  - Introduction to algorithms and how they translate into code.
- Introduction to Python
  - Overview of Python's history, features, and advantages
  - Real-world applications of Python in various fields (e.g., web development, data science, artificial intelligence)
- Installing Python & IDE (Thonny/VS Code)
  - Step-by-step guide on installing Python on different operating systems.
  - Introduction to Integrated Development Environments (IDEs) like Thonny and VS Code.
  - Setting up the IDE for Python development.
- Basic Syntax & Structure
  - Basic syntax of Python.
  - Creating a simple Python script.
  - Understanding indentation.
- First Statements, Comments & Output
  - Using the print() function to display output.
  - Adding comments to code for clarity, maintenance and documentation.
- **Activity:** Hello World, Covering Curiosities
  - Write a program to print "Hello, World!"
  - Create a program that takes user input (name) and generates a personalized greeting.

Introduction to Python Programming



Module 2: Variables, Data Types & Operators

- Variables and Naming Rules
  - Understanding variables as containers for storing data.
  - Rules for naming variables in Python (e.g., valid characters, case sensitivity).
- Data Types: Numbers, Strings, Booleans
  - Introduction to different data types: integers, floating-point numbers, strings, booleans.
  - Understanding the characteristics and use of each data type.
- Conditional Operators (Comparison & Logical)
  - Comparison operators (==, !=, >, <, >=, <=) for performing calculations.
  - Relational operators (==, !=, >, <, >=, <=) for comparing values.
  - Logical operators (and, or, not) for combining conditions.
- Type Conversion
  - Converting data from one type to another using functions like int(), float() and str().
- Simple Calculator
  - Write a program that takes two numbers as input and performs basic arithmetic operations (+, -, *, /).

Python Programming Concepts

Module 3: Conditional Logic (If-Else)

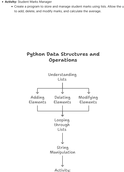
- Decision Making using if, elif, else
  - Introduction to conditional statements (code based on a condition).
  - Adding elif (else if) to handle multiple conditions.
  - Using else to execute code when no condition is met.
- Nested Conditions
  - Using if statements inside other if statements to create more complex logic.
- Simple Games using if-else
  - Developing simple games like number guessing games or decision-based stories using conditional logic.
- **Activity:** Number Guessing Game
  - Create a game where the computer generates a random number, and the user has to guess it. Provide feedback (higher or lower) until the user guesses correctly.

How to use conditional logic in Python?

Module 4: Loops in Python

- while Loop
  - Understanding the while loop and its syntax.
  - Using while loops to repeat a block of code as long as a condition is true.
- for Loop
  - Understanding the for loop and its syntax.
  - Using for loops to iterate over a sequence (e.g., a list or a string).
- for Loop with Range
  - Using the range() function to generate a sequence of numbers for use in for loops.
- Nested Loops
  - Using loops inside other loops to create more complex patterns or iterate over multi-dimensional data.
- **Activity:** Multiplication Table Generator Table
  - Write a program to print various patterns using nested loops (e.g., a triangle or a square).
  - Create a program to print the multiplication table for a given number.

Python Loop Mastery Cycle



Module 5: Functions and Code Reuse

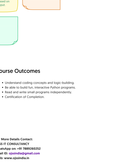
- Defining Functions
  - Introduction to the concept of functions and their importance in code organization.
  - Defining functions using the def keyword.
- Parameters and Return Values
  - Passing parameters to functions to provide input.
  - Returning values from functions using the return statement.
- Calling Functions
  - Calling functions to execute the code they contain.
- Using Built-in Functions
  - Exploring and using various built-in functions in Python (e.g., len(), max(), min(), abs()).
- **Activity:** Math Helper Functions
  - Write a program that implement specific mathematical operations (e.g., calculating the area of a circle, finding the factorial of a number).

Python Functions Mastery

Module 6: Data Structures (Lists, Strings)

- Working with Lists, Add, Delete, Modify
  - Introduction to lists as ordered, mutable collections of items.
  - Adding elements to a list using append(), insert().
  - Deleting elements from a list using remove(), pop().
  - Modifying elements in a list by assigning new values.
- Looping through Lists
  - Iterating over the elements of a list using for loops.
- Text Processing
  - Performing various operations on strings (e.g., concatenation, slicing, finding the length, changing case).
- Dictionaries (Key-Value Pairs)
  - Create a program to store and manage student marks using lists. Allow the user to add, delete, and modify marks, and calculate the average.

Python Data Structures and Operations



Module 7: File Handling (Optional for Class 10+)

- Reading and Writing Text Files
  - Introduction to text files and how to read/write using the open() function.
  - Reading data from a file using read(), readline(), and readlines().
  - Writing data to a file using write() and writelines().
- CSV Files
  - Introduction to CSV (Comma Separated Values) files.
  - Reading and writing CSV files using the csv module.
- Text Based Games
  - Storing progress data in text or CSV files for later use.
- **Activity:** Save Quiz Scores to File
  - Modify the quiz game from Module 3 to save the user's scores to a file.

Python File Handling and CSV Operations

Module 8: Fun with Python Projects

- Mini Projects
  - Rock Paper Scissors Game
    - Develop a classic Rock-Paper-Scissors game where the user plays against the computer.
  - Dice Simulator
    - Create a program that simulates rolling a dice and displays the result.
  - Quiz Game
    - Build a quiz game with multiple-choice questions and scoring.
  - Birthday Reminder
    - Develop a program that reminds the user of upcoming birthdays.
  - Story Generator
    - Create a program that generates random stories based on user input.

Python project ideas

Course Outcomes

- Understand coding concepts and logic building
- Be able to build fun, interactive Python programs.
- Read and write small programs independently
- Certification of Completion

For More Details Contact:
Colin C Colaço,Coder2
WhatsApp no: +91 7889392352
Email: colin@coder2.com
Web: www.coder2.com