Python learing guideline for 30 days,			
Conditions	Every day conplete 1 task, and more and more practice		
	Repeat yourself which learn yesterday		
	If conplete a task, please test properly suitable output with various type of input.		
	If every test sucessfully passed, then push in your github.		
	Today which you learned, search in google it's best solution.		
Day 1-5: Introduction to Python Basics		**Data**	**Progress status**
Day-1	Install Python and a text editor or IDE (Integrated Development Environment) like PyCharm or VS Code.		
Day-2	Learn about variables, data types (integers, floats, strings, booleans), and basic operations (+, -, *, /, %).		
Day-3	Dive into string manipulation: slicing, concatenation, methods like `upper()`, `lower()`, `split()`.		
Day-4	Understand control structures: if statements, for and while loops, and the `range()` function.		
Day-5	Practice with basic data structures: lists, tuples, dictionaries. Learn about indexing, slicing, and common operations.		
Day 6-10: Functions and Modules			
Day-6	Define functions, understand function arguments, return values, and scope.		
Day-7	Explore built-in functions and how to create your own modules.		
Day-8	Learn about importing modules, including standard library modules and third-party packages with `pip`.		
Day-9	Practice using functions and modules in various contexts.		
Day-10	Start working on small projects or exercises to reinforce your understanding.		
Day 11-15: Object-Oriented Programming (OOP)			
Day-11	Introduction to OOP concepts: classes, objects, attributes, and methods.		
Day-12	Understand inheritance, encapsulation, and polymorphism.		
Day-13	Dive deeper into OOP with examples and exercises.		
Day-14	Learn about special methods (`__init__`, `__str__`, etc.) and class variables vs instance variables.		
Day-15	Practice implementing classes and solving problems using OOP principles.		
Day 16-20: Intermediate Topics			
Day-16	Introduction to file handling: reading from and writing to files.		
Day-17	Exception handling: try, except, finally blocks for handling errors gracefully.		
Day-18	Explore regular expressions for pattern matching and text processing.		
Day-19	Learn about built-in data structures like sets and frozensets.		
Day-20	Dive into more advanced topics like decorators and context managers.		
Day 21-25: Web Development Basics			
Day-21	Introduction to web development with Flask or Django framework.		
Day-22	Learn about routing in Flask/Django and creating basic web pages.		
Day-23	Explore HTML templates and rendering dynamic content.		
Day-24	Integrate databases (SQLite or PostgreSQL) with Flask/Django for data persistence.		
Day-25	Deploy a simple web application locally or on a platform like Heroku.		
Day 26-30: Final Projects and Review			
Day-26	Work on a larger project that combines the skills you've learned.		
Day-27			
Day-28	Review your code, refactor, and optimize where possible.		
Day-29	Explore additional topics or libraries based on your interests (e.g., data visualization with Matplotlib, NumPy for numerical computing).		
Day-30	Celebrate your progress! Reflect on what you've learned and plan your next steps in Python programming.		
Remember, consistency is key. Spend at least an hour each day practicing coding and reinforcing what you've learned. Good luck!			