# Python learning plan

Week1: Learning Fundamentals

|  |  |
| --- | --- |
| Day 1 | print, variable, input, conditionals (if-else), List, strings, tuples, for loop, while loop, function, import Modules |
| Day 2 | Simple Coding Problems: Swap two variables, Convert Celsius to Fahrenheit, Sum of all digits in a number, Check a prime number etc. |
| Day 3 | Medium Coding Problems: Reverse a string (Check palindrome), Calculate GCD, Merge two sorted Array, Number guessing game, Calculate the age, etc. |
| Day 4 | **Data Structures:**Stack, Queue, Dictionary, Tuples, Tree, Linked List.   * [Sort values in Stack](https://www.geeksforgeeks.org/sort-a-stack-using-recursion/) * [Implement Two Stacks using one Array](https://www.geeksforgeeks.org/implement-two-stacks-in-an-array/) * [Detect Loop in a linked list](https://www.geeksforgeeks.org/detect-loop-in-a-linked-list/) * [Rotate a linked list](https://www.geeksforgeeks.org/rotate-a-linked-list/) * [Merge two sorted linked list](https://www.geeksforgeeks.org/merge-two-sorted-linked-lists/) * [Minimum value in Binary Search Tree](https://www.geeksforgeeks.org/find-the-minimum-element-in-a-binary-search-tree/) * [Height of a Binary Tree](https://www.geeksforgeeks.org/write-a-c-program-to-find-the-maximum-depth-or-height-of-a-tree/) * [kth maximum value in Binary Search Tree](https://www.geeksforgeeks.org/kth-largest-element-in-bst-when-modification-to-bst-is-not-allowed/) |
| Day 5 | **OOP:**Object, Class, Method and constructor, OOP- Inheritance |
| Day 6 | **Algorithm:**Search (Linear and Binary search), Sort (Bubble sort, Selection Sort), Recursive function (factorial, Fibonacci series), Time Complexity (Linear, Quadratic, and Constant). |
| Day 7 | Revise on week 1 learning. For example: Medium level Python Problem solving, OOP, Data Structure and Algorithm. Since these may seem alien to you. You need to invest more time here.  and try projects from:  <https://docs.python.org/3/tutorial/>  <https://pythonistaplanet.com/python-project-ideas/#Python_Tkinter_Project_Ideas> |

Week-2: Starting Software Development (Building Project)

|  |  |
| --- | --- |
| Day 8 | Start using VS code python extension and learn github, start adding your work done so work onto github  https://www.datacamp.com/learn-python-with-anaconda |
| Day 9 | First Project: Simple Calculator: Get familiar with Tkinter. [Create a simple calculator](https://www.simplifiedpython.net/python-calculator/). |
| Day 10 | Continuing on your project where you are trying to create Simple Calculator. It may take more time for creating your first project as it is your first time learning Python. |
| Day 11 | Revise on week 1 learning. For example: Medium level Python Problem solving, OOP, Data Structure and Algorithm. Since these may seem alien to you. You need to invest more time here.   * [Find duplicates in an array](https://www.geeksforgeeks.org/find-duplicates-in-on-time-and-constant-extra-space/) * [Array Rotation](https://www.geeksforgeeks.org/array-rotation/) * [Inversion of array](https://www.geeksforgeeks.org/counting-inversions/) * [Maximum of all subarrays of size k](https://www.geeksforgeeks.org/sliding-window-maximum-maximum-of-all-subarrays-of-size-k/) * [Trapping rainwater](https://www.geeksforgeeks.org/trapping-rain-water/) * [Kadane’s algorithm](https://www.geeksforgeeks.org/largest-sum-contiguous-subarray/) * [First non-repeating character in a stream](https://www.geeksforgeeks.org/find-first-non-repeating-character-stream-characters/) * [Rotate a linked list](https://www.geeksforgeeks.org/rotate-a-linked-list/) * [Remove loop in a linked list](https://www.geeksforgeeks.org/detect-and-remove-loop-in-a-linked-list/) |
| Day 12 | * [Searching in the linked list](https://www.geeksforgeeks.org/search-an-element-in-a-linked-list-iterative-and-recursive/) * [Find a middle element in a linked list](https://www.geeksforgeeks.org/write-a-c-function-to-print-the-middle-of-the-linked-list/) * [Find the Intersection point of two linked lists](https://www.geeksforgeeks.org/write-a-function-to-get-the-intersection-point-of-two-linked-lists/) * Sample Stack and Queue Problems * [Get minimum element from the stack](https://www.geeksforgeeks.org/design-a-stack-that-supports-getmin-in-o1-time-and-o1-extra-space/) * [Queue using Two Stacks](https://www.geeksforgeeks.org/queue-using-stacks/) * [Stack using Two Queues](https://www.geeksforgeeks.org/implement-stack-using-queue/) |
| Day 13 | * Implement [Breadth-First Search](https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/) and [Depth First Search](https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/) * [Find k smallest and largest elements in a list](https://www.geeksforgeeks.org/k-largestor-smallest-elements-in-an-array/) * [Check for BST](https://www.geeksforgeeks.org/a-program-to-check-if-a-binary-tree-is-bst-or-not/) * [Vertical traversal of binary tree](https://www.geeksforgeeks.org/print-binary-tree-vertical-order/) * [Boundary traversal of binary tree](https://www.geeksforgeeks.org/boundary-traversal-of-binary-tree/) * [Delete a node from BST](https://www.geeksforgeeks.org/binary-search-tree-set-2-delete/) * Detect Cycle in a [directed](https://www.geeksforgeeks.org/detect-cycle-in-a-graph/)and [undirected](https://www.geeksforgeeks.org/detect-cycle-undirected-graph/)graph * [Find the number of islands](https://www.geeksforgeeks.org/find-number-of-islands/) * [Word Boggle](https://practice.geeksforgeeks.org/problems/word-boggle/0) * [Implementing Dijkstra](https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/)   Found from: https://www.geeksforgeeks.org/100-days-of-code-a-complete-guide-for-beginners-and-experienced/ |
| Day 14 | Learn Server and hosting [to host your project](https://devcenter.heroku.com/articles/getting-started-with-python). Create a [Heroku](https://www.heroku.com/) setup and deploy the app you built. |

[**https://www.dataquest.io/blog/python-projects-for-beginners/**](https://www.dataquest.io/blog/python-projects-for-beginners/)

[**https://app.dataquest.io/m/139/introduction-to-k-nearest-neighbors/1/introduction**](https://app.dataquest.io/m/139/introduction-to-k-nearest-neighbors/1/introduction)

**https://www.datacamp.com/learn-python-with-anaconda**