Python Practice Programs

Basics:

1. Print a floating-point number with n decimal places.
2. WAP to read characters, symbols and integers(0 to 9), print their ASCII value.
Follow up: WAP to read integers(0 to 127) and print their ASCII character.
3. Find the simple interest and compound interest.
4. Find the largest number among three numbers.
5. Find the area and perimeter of a triangle, square and rectangle.
6. Check if a given year is a leap year or not.
7. Check if a given number is prime or not.
8. Check if a given number is palindrome or not.
9. WAP to build a simple calculator.
11. Find the nth fibonacci number.
12. WAP to print pascal's triangle.
13. Learn about **random, math, os, sys, datetime, json, pickle** and **re** built-in libraries of python.
14. Print numbers from start to end. Read start and end values from the user.

Strings:

1. Find if two strings are the same.
2. Find the maximum occurring character in a string.
3. Reverse a string.
4. Check if a string is palindrome.
5. Find the duplicate characters of a string.
6. Remove all the duplicate characters from a string.
7. Remove characters from the first string which are present in the second string.
8. Reverse each word of a string.

Eg. Hello World!
olleH !dlroW

9. Find all the substrings of a given string.
10. Find if the given string(s1) is a substring of a string(s2).
11. Find all the subsequences of a given string.
12. Find if the given string(s1) is a subsequence of a string(s2).
13. Check whether the two strings are anagram of each other.
14. Print second most frequent character of a string.
15. Split the string into words without using the split() method.

Python Practice Programs

Array(List), 2d Array(Matrix):

1. Find if two arrays are the same.
2. Sort an array. Learn all these sorting techniques **Bubble Sort, Selection Sort, Insertion Sort, Quick Sort, and Merge Sort.**
3. Find the largest and smallest elements of an array. Also second largest and second smallest element of an array.
4. Find elements occurring an odd number of times in an array.
5. Search for an element in a sorted array. Use binary search technique.
6. WAP to print the array after rotating it towards right by **k** times.
7. Check if the given array is sorted.
8. Remove duplicates from the sorted array.
9. Print all the subarrays of a given array.
10. Find if an array(arr1) is a subarray of another array(arr2).
11. Print all the subsequences of a given array.
12. Find if an array(arr1) is a subsequence of another array(arr2).
13. Read and print a matrix of **m** rows and **n** columns.
14. Find row-wise sum and product of a matrix.
15. Find column-wise sum and product of a matrix.
16. Find the largest and smallest element of a matrix.
17. Count the number of odd and even elements of a matrix.
18. Find the sum and product of diagonal elements of a matrix.
19. Find the transpose of a matrix.
20. Print matrix in reverse order.
21. Add two matrices.
22. Multiply two matrices.
23. Rotate a matrix by 90 degrees clockwise.
24. Count the frequency of each element in a matrix.
25. Print matrix in row-wise and column-wise reverse order.

Object-Oriented Programming and Data Structures:

1. Implement the class circle, rectangle, square and triangle. Include all the required attributes and methods for each class.
2. Implement the class human and inherit it into student, employee, pilot, doctor, scientist. Include all the required attributes and methods for each class.
3. Implement menu driven stack data structure.

Python Practice Programs

4. Implement menu driven queue, doubly queue and circular queue data structure.
5. Implement menu driven linked list, doubly linked list and circular linked list data structure.
6. Implement menu driven binary tree data structure.
7. Implement a Digital Library Management System.
8. Implement a University Management System.
9. Implement a Bank Management System.
10. Implement an Online Ticket Booking System.

Note :

- **➔ Learn function concepts in python on your own, then, solve the above programs.**
- **➔ Avoid built-in functions/methods, slicing.**
- **➔ In OOP and DS implement any one program from 7, 8, 9, 10 bcz these are mini-scale projects which take time. If you are having time, implement according to your interest.**