Chapter 8 : Algorithms

Aviral Janveja

1 Introduction

We come back to where we started from, in chapter 1 :

> "**Programming** is the art of instructing computers to perform specific tasks. At the core of a computer program, an **algorithm** serves as a systematic set of steps or instructions designed to solve a computational problem. So, while an algorithm provides the strategy, a program brings that strategy to life in a formal language that computers can comprehend."

Till now we have focused on the programming toolbox, syntax and specific features of the Python language. However, there could be many ways to solve a particular problem in the very same programming language. Hence, we now take that algorithmic or strategic part into consideration. The question here is :

> "How can we relate our choices in the design of an algorithm, to the amount of time it will take to solve a particular problem. For example, how will the program efficiency differ if you choose to use recursion versus iteration."

Why care about program efficiency when the computers are getting faster every year? **Because**, the size of data-sets and problems are increasing as well. For example, being able to search through the billions of web-pages on the internet.

2 Efficiency

In terms of efficiency, there are two aspects to consider : **space efficiency** and **time efficiency**. The amount of storage space required versus the amount of time required to compute the result. There might be a trade-off between the two at times. However, in this chapter we are going to focus on the **time efficiency** of our algorithms.

2.1 How To Measure Time Efficiency