Python Introduction

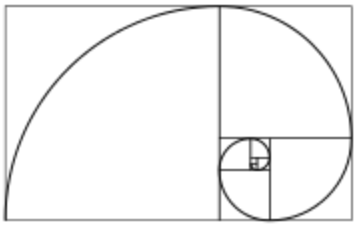
Dr S. S. Chandra

Complete Python programs for the following tasks and demonstrate the working code to your tutor. Initial scripts with basic structure have been provided to help you get started. Feel free to use web search, but remember not to deprive yourself of a learning experience!

1. Write a program that computes the Fibonacci sequence up to a number N and prints it to standard output. The Fibonacci sequence is defined as a recurrence relation that is a sum of the two previous Fibonacci numbers:

   $$F_{N-2} + F_{N-1} = F_N$$

   The Fibonacci sequence for N=10 is 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55.

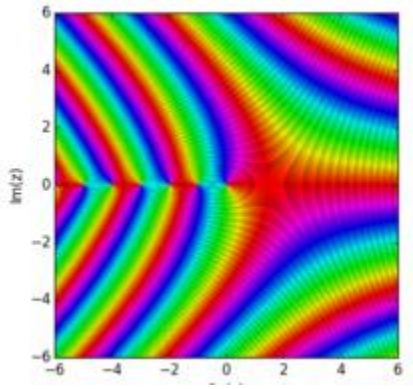
   

   Interesting fact: These numbers produce a spiral and is a common recurring pattern in nature. For example, the seeds in a sunflower are arranged via this sequence!

2. Compute the factorial function up to any given number N. The factorial is defined as

   $$N! = 1 \times 2 \times \cdots \times (N-1) \times N$$

   For example: 5! = 120. Use your program to determine the number of possibilities of the order of the cards when shuffling a deck of cards (N=52).

   Interesting fact: The factorial function was generalised by Euler to create the Gamma function which is now thought to be the basis of the distribution of prime numbers (see Riemann's Hypothesis) and String theory, the current most developed theory for understanding the fundamental forces in nature, particle physics and a quantum theory of gravity.

3. Code up examples of above that use recursion (see appendix) if you haven't already done so.
4. Create a program that encrypts and decrypts a message (for simplicity, a single integer) using RSA encryption. To encrypt a signal/message, you need to compute

   $$c \equiv m^e \bmod n$$

   And to decrypt using

   $$c^d \equiv (m^e)^d \equiv m \bmod n$$

   Where m is the integer representing the message character, e is the public key and d is the private key.

   Interesting fact: This encryption scheme is the basis of nearly all encryption done over the internet.

Appendix

Given a function f, the concept of recursion is to reduce the problem by an increment or portion and to recall the function f inside itself with this reduced portion. The procedure is terminated when one reaches the base case, a case where the problem cannot be reduced any further where the solution is known. At this stage, the result is propagated back up to all the previous calls to get the final result.

For example, we can define the natural numbers $\mathbb{N}$ recursively as the successor function S, i.e. the "what comes after" function. Given that the successor of 0 is 1, i.e. $S(0) = 1$, all other numbers can be defined as

$$S(x + 1) = S(x) + 1$$

Therefore, to compute $S(2)$, we end up with

$$\begin{aligned} S(2) &= S(1) + 1 \\ &= (S(0) + 1) + 1 \\ &= 1 + 1 + 1 \\ &= 3 \end{aligned}$$

This formulation of the natural numbers comes from the Peano axioms and will later be shown to form the basis of the recursive function definition of computation.