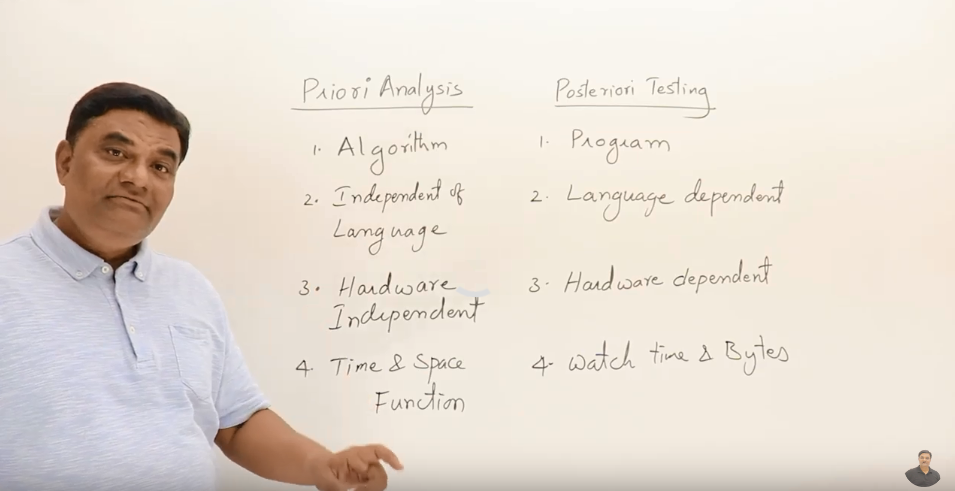
# [Priori Analysis and Posteriri Testing](https://www.youtube.com/watch?v=-JTq1BFBwmo&list=PLDN4rrl48XKpZkf03iYFl-O29szjTrs_O&index=2)

**Priori analysis** is done over the algorithm.

* We analyse the algorithm & we study the algorithm in greater details.

**Posteriori testing** is done over the program.



# Characteristics of Algorithm

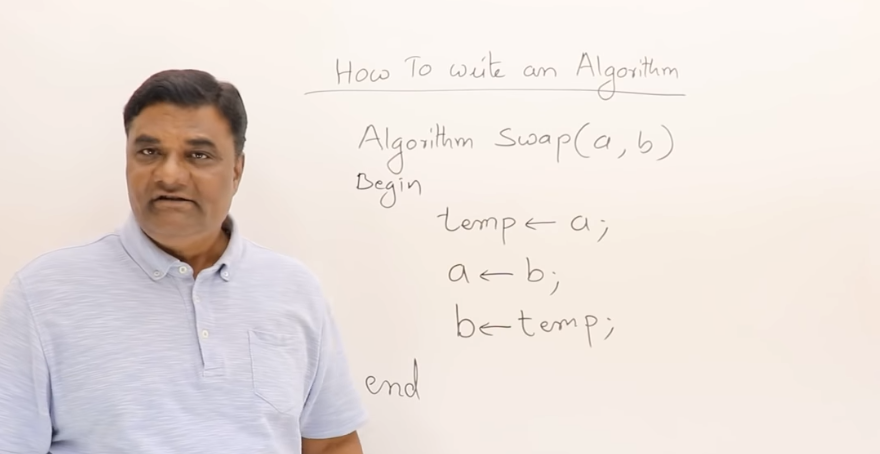
1. **input** - algorithm can take 0 or more input
2. **output** - algorithm must generate an output at least 1 output
3. **definiteness** - every statement you write should be definite and clear. Namely, you can’t write a statement which cannot be understood or which cannot be solved for instance: square root of a negative number which is unknown
4. **finiteness** - algorithm is like a function, function has a limited set of steps, it **stops** and return the result. Likewise algorithm must terminate at some point and should stop. In contrast to for instance SQL servers that run forever, algorithms must stop. for instance:
   1. if you cook something there is a finite and at some point the dish is ready.
   2. if you are experimenting any experience, you stop at some point and the solution is ready.
5. effictiveness - you shouldn’t write unnecessary statements. For instance:
   1. you are cutting some vegetables, but they are not in the recipe and not used in dish.
   2. you are boiling some chemical ingredient but it isn’t a part of the solution.

# How to write and analyze algorithm

**Writing an algorithm**

You can use any syntax to write an algorithm, below is a sample of an algorithm for instance: swap algorithm.

Note: swap means exchange



**Analyzing an algorithm**

Criterias for analyzing an algorithm:

1. Time

algorithms are procedures for solving problem whether you do it manually using pen and paper or making a program and let a machine do it.

Whatever the method is how much time it is taking whether it is manually written or written by a program.

it is not watch time it is function time :/

1. Space

How much memory or space does it consume for instance if it is written in machine.

some others:

1. Network: how much data is transferred.
2. Power: how much power it is consuming.
3. CPU registers: how much CPU registers it is consuming

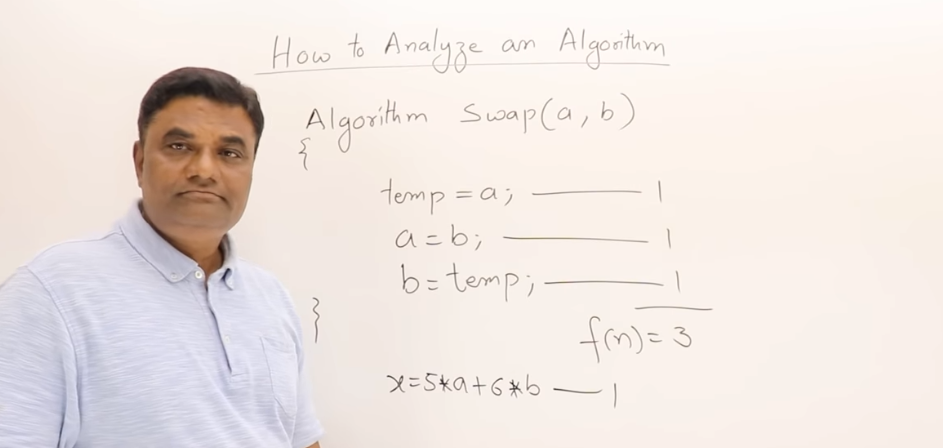
…

For example, lets analyze swap algorithm, in terms of time, space, network, power and CPU registers …

1. **time analysis of swap algorithm**

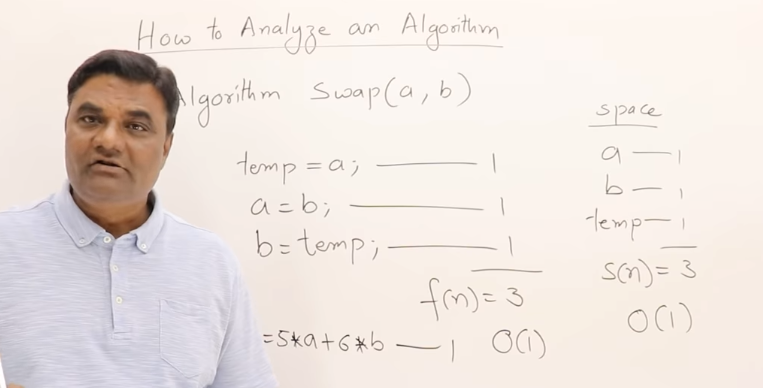
Every statement in an algorithm takes one unit of time.

So this takes 3 units of time. Namely f(n) = 3



1. **space analysis of swap algorithm**

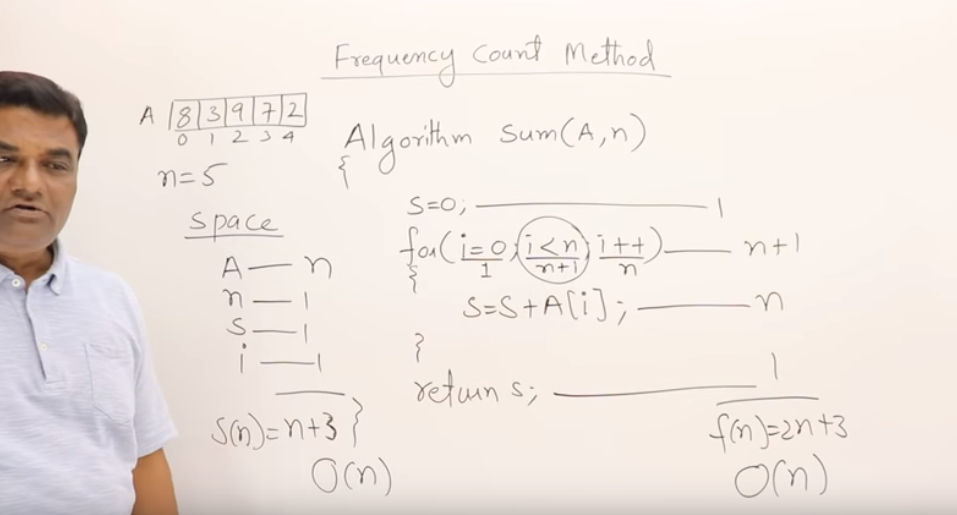
three variables are used here, so space is 3 or s(n) = 3



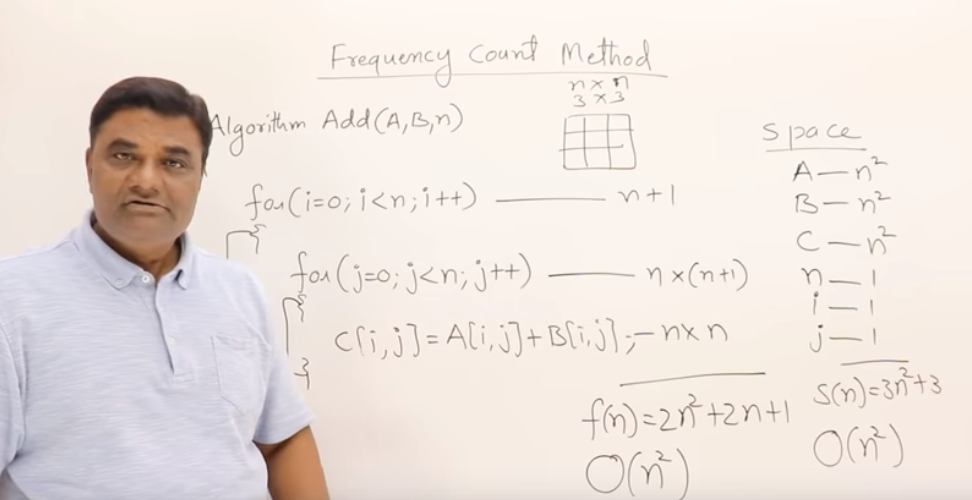
# Frequency Count Method

In this chapter we will take 3 algorithms and figure out their time and space complexity using frequency count method.

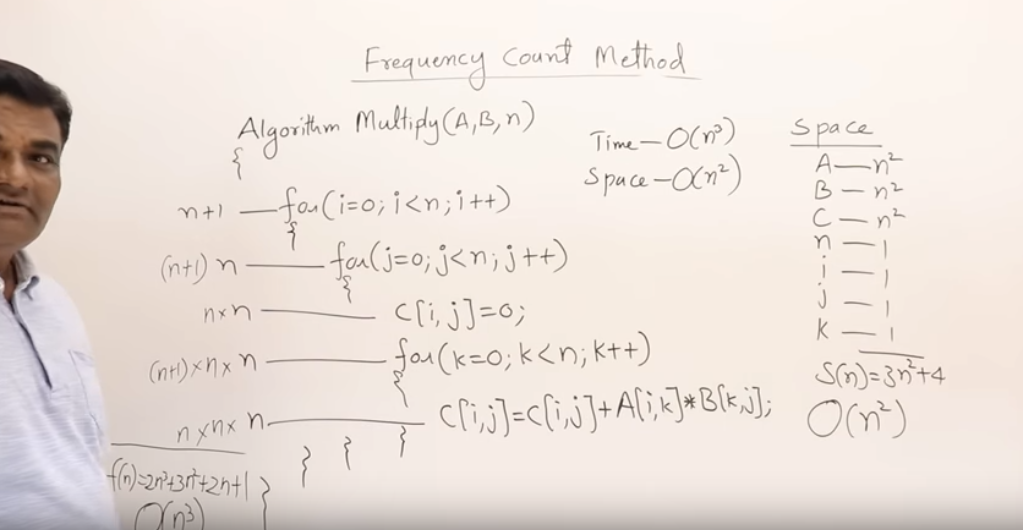
Calculating time and space complexity of a simple sum Algorithm **eg#1**: this algorithm is to find sum of all elements of an array.



Calculating time and space complexity of a simple add Algorithm **eg#2**:



Calculating time and space complexity of a simple sum Algorithm **eg#3**:



# Time Complexity #1

