***Algorithm/ Program*** is a step-by-step procedure to solve a computational problem

***Algorithm*** is a set of steps to accomplish a task

General E.g.:

* algorithm to take a path form home to school
* algorithm to make cake

Real world E.g.:

* google hangout transmit live video using **audio & video compression algorithm**
* google map uses **route finding algorithm** to find us a path
* pixar color a 3D model based on lighting in a virtual room using **rendering algorithm**
* NASA arrange solar panels on international space station **using optimizing & scheduling algorithm**

***Good Algorithm:*** Correctness & Efficiency

SDLC (Software Development Lifecycle is broadly divided into two phases)

1. Designing Phase
2. Implementation Phase/ Development Phase

|  |  |  |
| --- | --- | --- |
| Sr. No. | Algorithm | Programs |
| Written at | Designing Phase | Implementation Phase |
| Written by | Domain Knowledge/ Designer | Programmer |
| Written in | Any Language (pseudo code) | Programming Languages |
| Machine Dependencies | Independent of Hardware and Software (Operating System) | Dependent of Hardware and  Operating System |
| Checking | Analyze | Testing |

|  |  |
| --- | --- |
| Priori Analysis | Posteriori Testing |
| Algorithm | Programs |
| Language Independent | Programming Language Dependent |
| Hardware Independent | Hardware Dependent |
| Measure Time and Space Complexity in Functions | Watch time and bytes of a program |

***Characteristics/ Properties of an Algorithm***

* **Input**- It should take 0 or more inputs.
* **Output**- Must generate some result even None is accepted. It should possess at least 1 output.
* **Definiteness**- Every statement should be unambiguous/ clear meaning should possess some meaning and able to solve.
* **Finiteness**- An algorithm must stop/ terminate at some point.
* **Effectiveness**- Every statement should have some meaning and purpose.

While Writing algorithm we don’t bother for data types use, declaration of variables and any programming language syntax.

***Analyze of an Algorithm***

1. Time
2. Space

**Additional factors**

Network Consumption (Data Transfer) | Power Consumption | CPU Register Utilization

# Time and Space Analysis

Every simple statement of an algorithm takes one units of time

**Example**

**Time Analysis**

Algorithm Swap(a, b)

{

Temp = a ----------- 1

a = b ----------------- 1

b = temp ------------ 1

}

Total time taken by each statement is 1 and the time function f(n) = 3 units of time i.e. O(1) [constant]

**Space Analysis**

a = 1

b = 1

temp = 1

The space function s(n) = 3 words i.e. O(1) [constant]

X = 5\*a + 6\*b --------------- 1

Above statement will take one unit of time f(n) = 1

But in broader **view** with **respect to machine code** the statement will break into multiple statements

5\*a ---------- 1 => p

6\*b ---------- 1 => q

p + q -------- 1 => o

X <= o --- ---- 1

And the total time of the function will be f(n) =4 i.e. O(1) [constant]

Frequency Count Method

Generally, every loop run for n+1 times and whatever the body of the loop runs for n times when it is incremented by 1 step.

n => when the condition is true + 1 => when the condition becomes false

If it is incremented by 2 steps, then the time function for the algorithm (loop) will be n/2.

If it is incremented by t steps, then the time function will be n/t.

**Reference** [1.4 Frequency Count Method - YouTube](https://www.youtube.com/watch?v=1U3Uwct45IY&list=PLDN4rrl48XKpZkf03iYFl-O29szjTrs_O&index=5)