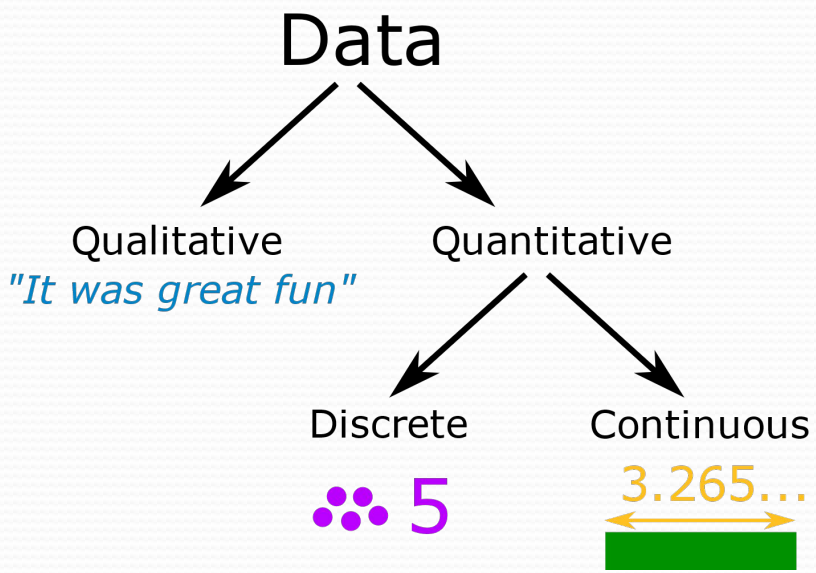
What is Data?

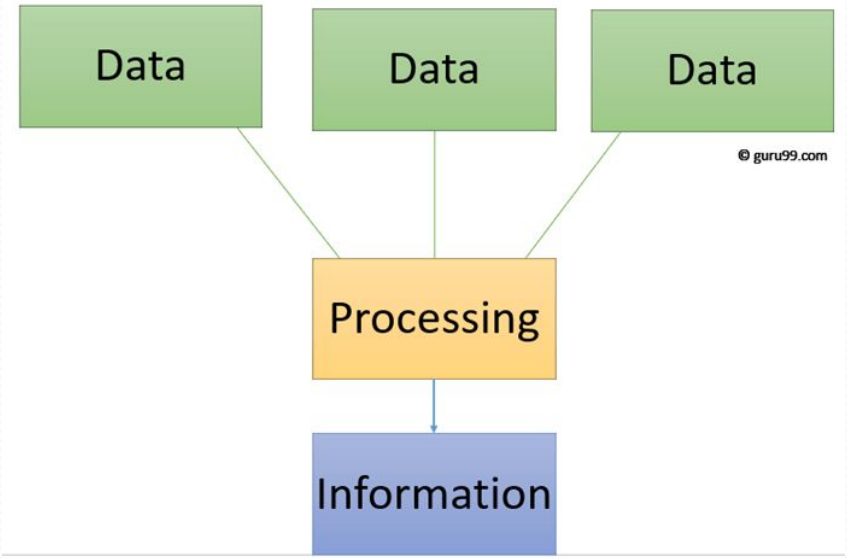
- ✔ Data are simply value or set of values
- ✔ Originally, data is plural form of “datum”, a Latin word
- ✔ A “datum” is a single entity, a single point of matter
- ✔ Data represents a collection of “data points”
- ✔ Today, “data” is used in a singular or plural form
- ✔ Data is raw fact

Representation of Data



What is Information?

- Information is data that have been processed to make them meaningful and useful.
- Data + Meaning = Information
- Information is processed data
- Information is meaningful data
- Information provides knowledge about certain matter



Example of Data & information:

- The history of temperature reading all over the world for the past 100 years is data.
- If this data is organized and analyzed to find that global temperature is rising, then that is information

What is Abstract Data Type (ADT)?

- A useful tool for specifying the logical properties of a data type is the Abstract Data Type or ADT.
- An Abstract Data Type is a mathematical model containing a set of values and a set of operations that can be performed on those values.
- An Abstract data Type is a special kind of data type, whose behavior is defined by a set of values and set of predefined operations.
- An ADT is a logical description of how we view that data and the operations.
- ADT provides an implementation independent view of data.
- The implementation of an abstract data type, often referred to as a Data Structure, will require that we provide a physical view of data.

What is Data Structure?

- Data may be organized in many different ways; the logical or mathematical model of a particular organization of data is called Data Structure.
- Data structure is a way to store and organized data so that it can be used efficiently.

Algorithm:

What is an Algorithm?

- An algorithm is a well defined list of steps for solving a particular problem.
- Algorithm refers to a set of steps / instructions that step-by-step define how a work is to be executed in order to get the expected results.
- An algorithm is a mathematical process to solve a problem using a finite number of steps.
- An algorithm is the set of instructions that defines not just what needs to be done but how to do it.

Characteristic of an Algorithm:

An algorithm should have the following characteristics:

- **Input:** An algorithm should have zero or more well defined inputs.
- **Output:** An algorithm should have at least one desired output.
- **Finiteness:** Algorithm must terminate after a finite number of steps.
- **Definiteness:** Algorithm should be clear and unambiguous. Each of its steps and their inputs / outputs should be clear and must lead to only one meaning.
- **Feasibility:** Should be feasible with the available resources.

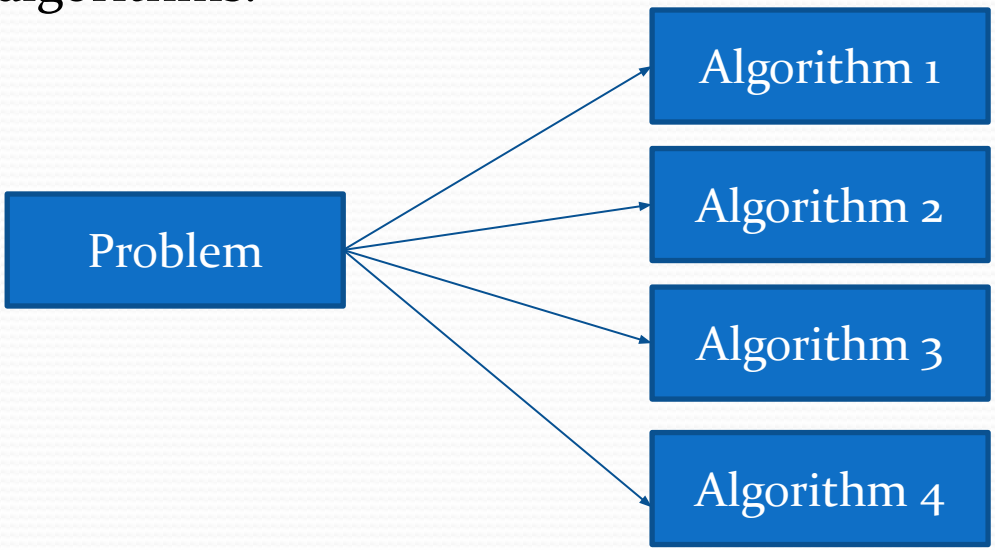
Note:

1. An algorithm should have step-by-step directions, which should be independent of any programming language.
2. Each algorithm will involve at least one particular data structure.

Analysis of Algorithm:

Why do we analyze an Algorithm?

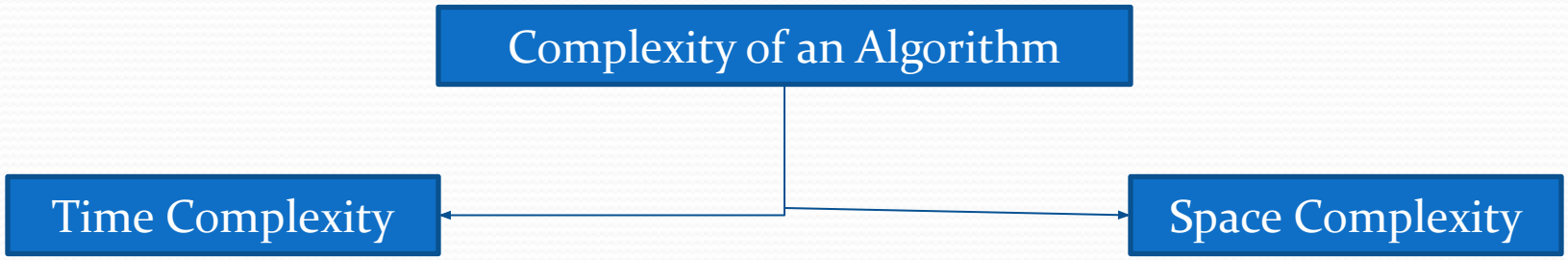
- One problem can have many algorithms (solutions) which can provide the required solutions. So, we need to compare the algorithms of that problem with respect to time and space. To find the best algorithm, we analyze algorithms.



- Note: The time complexity of an algorithm largely depend on the problem input size (n).

Analysis of Algorithm: (Cont….)

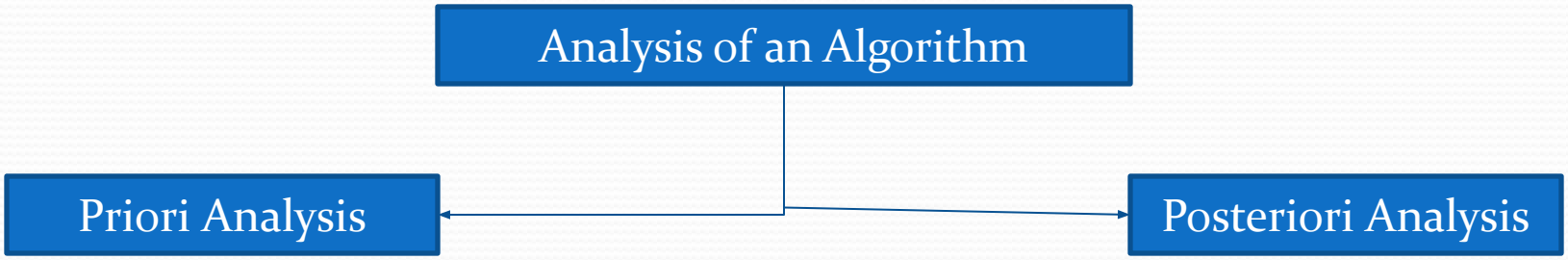
- Complexity of an algorithm measure how "fast" is the algorithm (time complexity) and what "amount" of memory it uses (space complexity)



- Time Complexity: Time complexity of an algorithm is measured by counting the total primitive operations performed by the algorithm.
- Space Complexity: Space complexity of an algorithm is the maximum memory space required by the algorithm in its life cycle.

Note: Efficiency of an algorithm depends on time complexity and space complexity.

Analysis of Algorithm: (Cont….)



- Time complexity of an algorithm can be calculated by using the above two methods.
- Efficiency of an algorithm can be analyzed at two different stages, before implementation and after implementation.

Priori Analysis: This is a theoretical analysis of an algorithm.

- It is independent of any programming language and types of hardware.
- The time complexity of an algorithm using priori analysis is same for every system.
- If the programming running faster, credit goes to the programmers.

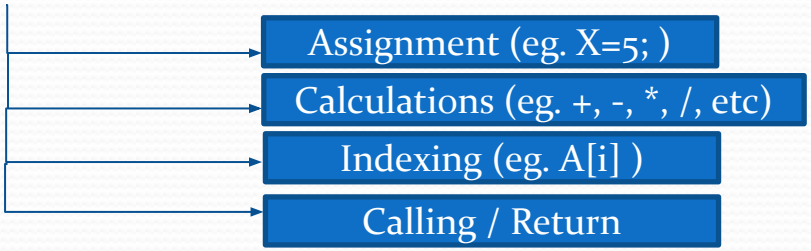
Analysis of Algorithm: (Cont….)

Posteriori Analysis: This is an empirical analysis of an algorithm. The selected algorithm is implemented using a programming language. This is then executed on a target computer machine.

- It is dependent on the programming language and type of hardware used during execution of the algorithm.
- The time complexity of an algorithm using posterior analysis differ from system to system.
- If the time taken by the algorithm is less, then the credit will go to the programmer as well as compiler and hardware used.

Rules for calculating Time Complexity of an Algorithm:

- Primitive operations:



Note: Each primitive operation takes one unit time. (Assumption)

Example:

	Statement	Unit Time
1.	i=3;	1
2.	A[i]=5;	2

1+1

How to calculate time complexity of an algorithm?

Example:

```
int sum_till_n(int n)
{
        int sum=0;                          1

        for( i=1;  i<=n;  i++ )             1
                                            n
        {                                   (n+1)
         sum = sum + i ;                    2n
        }

        return sum;                         1
}
```

Unit Time

$$T(n) = 1 + 1 + n + (n+1) + 2n + 1 \quad = \quad 4n + 4$$