Current Skill Structure of an Algorithm

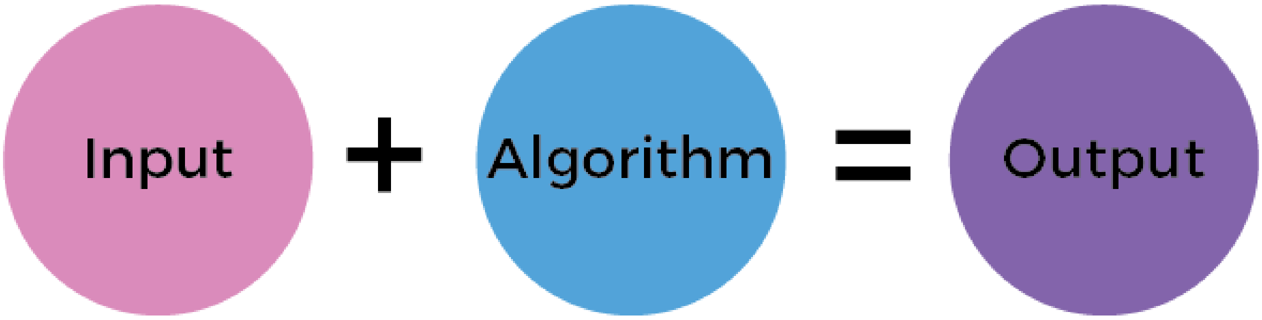
What is an algorithm :

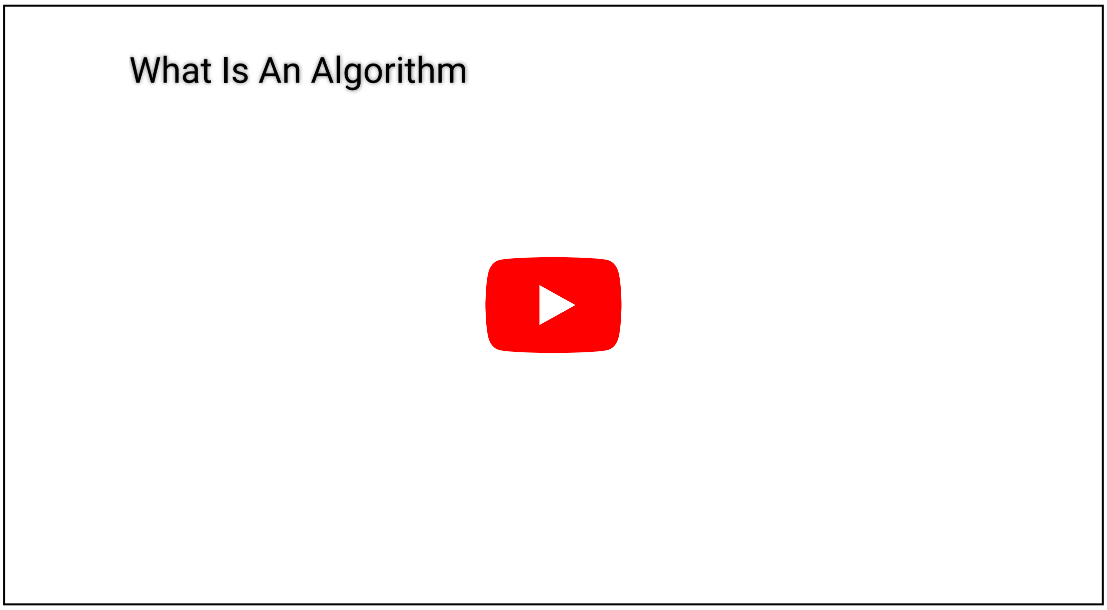
An algorithm is a set of instructions for solving a problem, in other words it a way of providing a result from data.

In other words, Algorithm presents a process or set of rules to be followed to obtain the expected output from the given input.

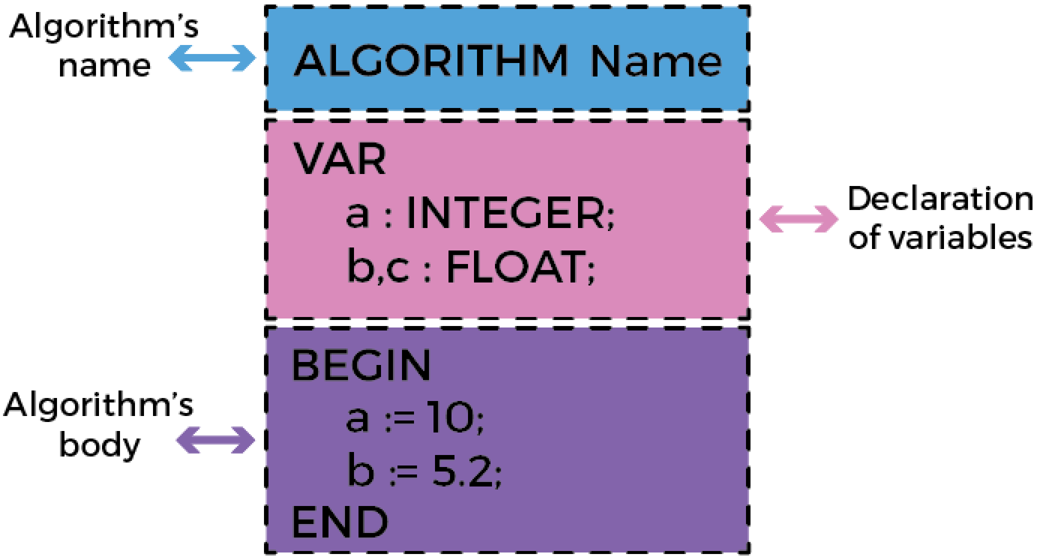
Why do we write algorithms :

- Develop the logic of programming.
- Stay focused on the logic only.
- Check the complexity of the program.





Structure of Algorithm :



```
ALGORITHM Name
VAR
    a : INTEGER;
    b,c : FLOAT;
BEGIN
    a := 10;
    b := 5.2;
END
```

Why do we write algorithms :

- Develop the logic of programming.
- Stay focused on the logic only.
- Check the complexity of the program

How to Design an Algorithm?

Before we start writing our algorithm and in order to make sure that we are on the right path there are some requirements that must be fulfilled:

Defining the problem that needs to be solved by the algorithm.

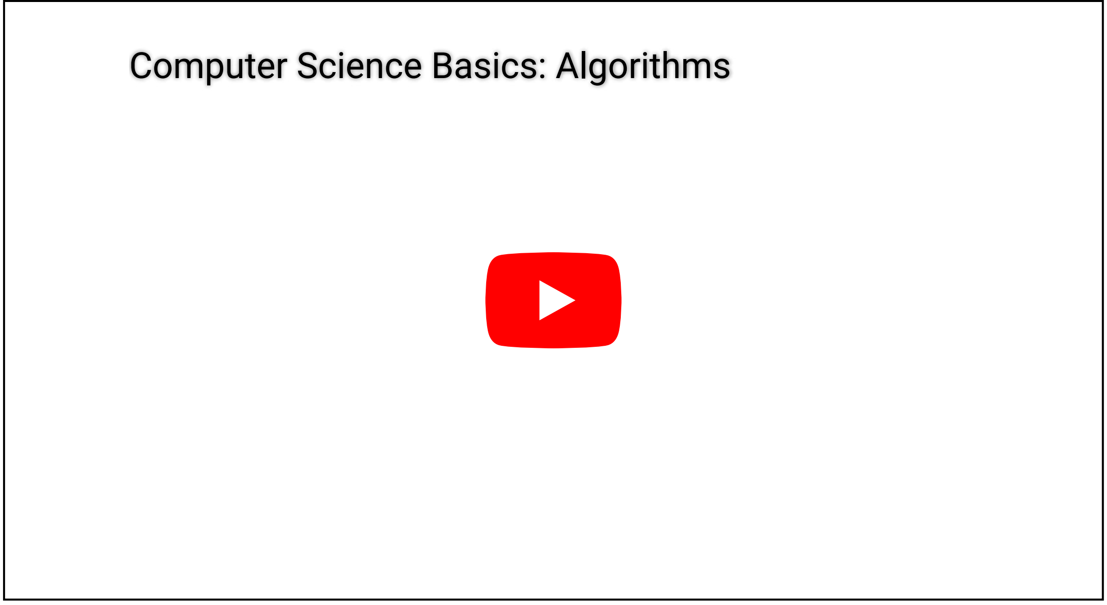
Identifying the constraints that should be taken into consideration while solving the problem.

Gathering the input to solve the problem.

Recognizing the output to be expected when the problem is solved.

Describing the solution to this problem.

Once these requirements are fulfilled, we can go ahead and start writing our algorithm.



How to Design an Algorithm?

Let's consider the following example, we need to write an algorithm that adds three numbers and prints their sum.

If we apply what we have learned from the previous slide, we'll go through 2 steps while solving our sum algorithm.

- **Step 1: Fulfilling the requirements:**

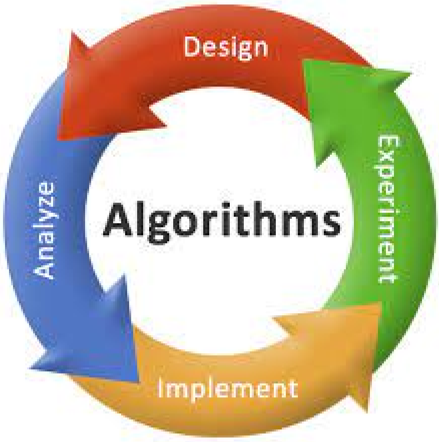
1. **The problem:** Add three numbers and print their sum.
2. **The constraints**: The numbers must contain only digits and no other characters.
3. **The input:** The three numbers to be added.
4. **The output:** the sum of these three numbers.
5. **The solution:** The solution consists of reading the three numbers, and adding them. It can be done using the '+' operator, or bitwise (&), or any other method.

Once we complete the requirements, we simplify our algorithm into several steps of instructions.

- **Step 2: Designing the algorithm:**

1. **START**: We declare the beginning of our algorithm.
2. **Declare 3 integer variables:** num1, num2 and num3.
3. **Read the value of each variable:** Respectively put them into the variable as inputs.
4. **Declare an integer variable sum:** This variable will contain the resultant sum of the 3 numbers.
5. **Add the three numbers and store the result in the variable sum.**
6. **Print the value of the variable sum.**

7. **END**: We declare the end of our algorithm.



How to design an Algorithm?

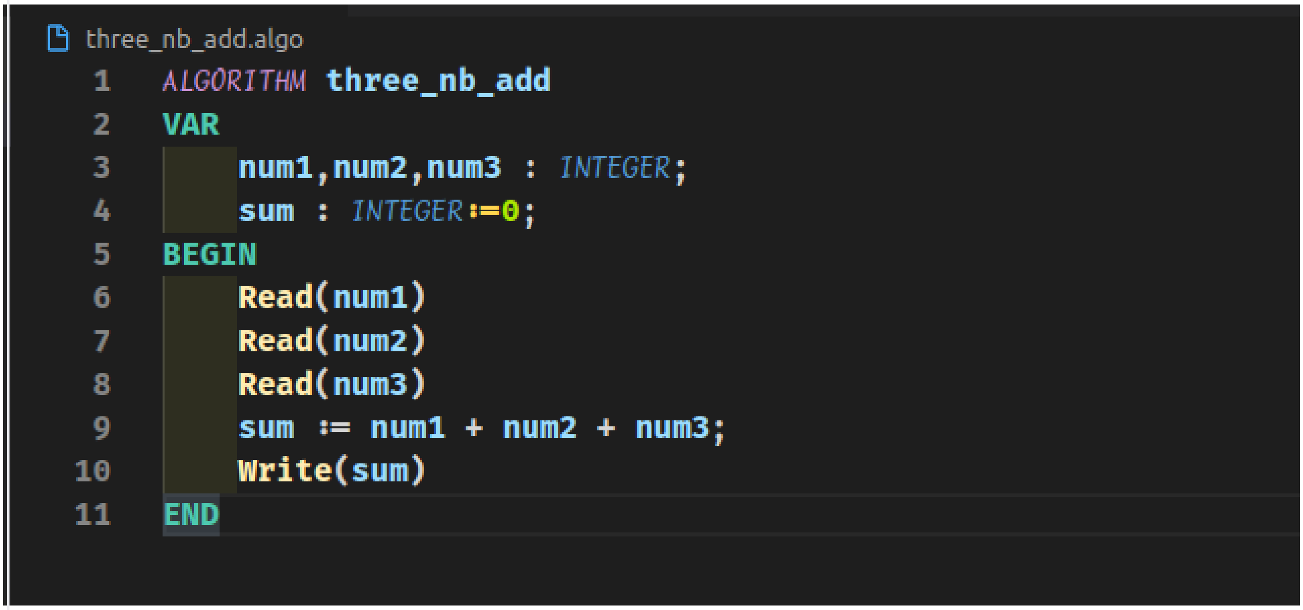
After defining the pre-requests and make a draft of our algorithm let's make it look like a real algorithm. To do so, we'll use the visual code as our default IDE and we have to add these two extensions to make it easier:

1. algo-gmc by "Hmida Rojabani":
2. indent-rainbow by "oderwat":

Now after setting up the environment, we need to create a file named three_nb_add with the extension ".algo"

The image below is the implementation of our algorithm in VSCode.

Go ahead and try it yourself !



```
ALGORITHM three_nb_add
VAR
    num1,num2,num3 : INTEGER;
    sum : INTEGER:=0;
BEGIN
    Read(num1)
    Read(num2)
    Read(num3)
    sum := num1 + num2 + num3;
    Write(sum)
END
```

‹ Previous

next ›