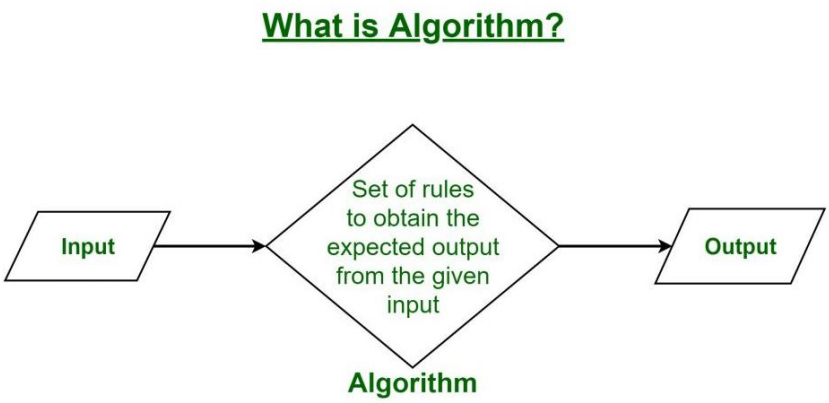
1. **Write an algorithm.**

An algorithm is a set of instructions that describes how to solve a problem (Prabhu, 2019).



For instance, here are some steps you can take to write an algorithm:

1. Determine the outcome of your code. What specific problem or task do you want it to solve?
2. Select a starting point.
3. Find the algorithm's endpoint.
4. List the steps from start to finish.
5. Determine how you will complete each step.
6. Review the algorithm.

Before writing an algorithm, you must understand how to solve a problem. The solution must be logical, and all inputs and outputs must be identified. You can print or display all of the results.

Here is a quick method for finding the maximum element in an array of integers:

Algorithm: FindMaxElement

Input: An array *of* integers, A

Output: The maximum element *in* A

1. **Set** a variable max **to** the first element **of** A.
2. **For each** subsequent element **in** A, compare it **to** max.
3. **If** the current element **is** greater than max, update max **to** be the current element.
4. **Continue until** all elements **in** A have been compared.
5. **Return** max **as** the maximum element **in** A.

This algorithm assumes that the array's first element is the maximum element. It then loops through the remaining elements in the array, updating the max variable whenever an element is found that is greater than the current maximum. Finally, it returns the maximum number of elements discovered.

2. **Investigate the quality parameters of an algorithm.**

An algorithm's quality parameters are used to assess how well an algorithm performs. The parameters can be used to determine an algorithm's efficiency, memory consumption, and accuracy (Khan Academy, N.D.).

An algorithm's quality parameters assess how well it performs its intended task (Nechvolod, 2018). For instance, here are some standard quality parameters that can be used to evaluate an algorithm's effectiveness and efficiency are as follows:

1. The algorithm's *correctness* refers to whether it produces the correct output for a given input. The algorithm must handle all possible inputs and produce the expected output.
2. The *efficiency* of an algorithm refers to how quickly it can complete its task and how much computational resources it requires. A good algorithm should be able to solve the problem in a reasonable amount of time and with few resources.
3. The algorithm's *robustness* refers to handling unexpected or incorrect input. The algorithm should be capable of dealing with errors, invalid input, and unexpected situations without crashing or producing incorrect output.
4. The algorithm's *Maintainability* refers to how simple it is to modify and maintain the algorithm over time. The algorithm should be designed to be simple to understand, modify, and extend as needed.
5. The algorithm's *Scalability* refers to how well an algorithm performs as the size of the input data increases. Scalable algorithms are those that can handle large inputs without significantly degrading performance.
6. The algorithm's *Usability* refers to how easy it is for users to use and interact with the algorithm. The algorithm should have a simple interface, be simple to understand, and be capable of providing useful feedback to users.

We can gain insight into an algorithm's strengths and weaknesses and identify areas for improvement by evaluating it against these quality parameters. Finally, how well an algorithm performs in real-world scenarios and whether it meets the needs of its users determines its quality.

References:

Prabhu, R. (2019). *Introduction to Algorithms*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/introduction-to-algorithms/.

Khan Academy. (n.d.). *Measuring an algorithm's efficiency | AP CSP (article) | Khan Academy*. [online] Available at: https://www.khanacademy.org/computing/ap-computer-science-principles/algorithms-101/evaluating-algorithms/a/measuring-an-algorithms-efficiency.

Nechvolod, A. (2018). *12 software architecture quality attributes and their types*. [online] Syndicode - Custom Software Development Company. Available at: https://syndicode.com/blog/12-software-architecture-quality-attributes/.