ALGORITHM

DEFINITION AND ITS PURPOSE

DEFINITION:

AN ALGORITHM IS A PRECISE SERIES OF STEPS DESIGNED TO SOLVE A PARTICULAR ISSUE OR ACCOMPLISH A CERTAIN ACTIVITY.

PURPOSE:

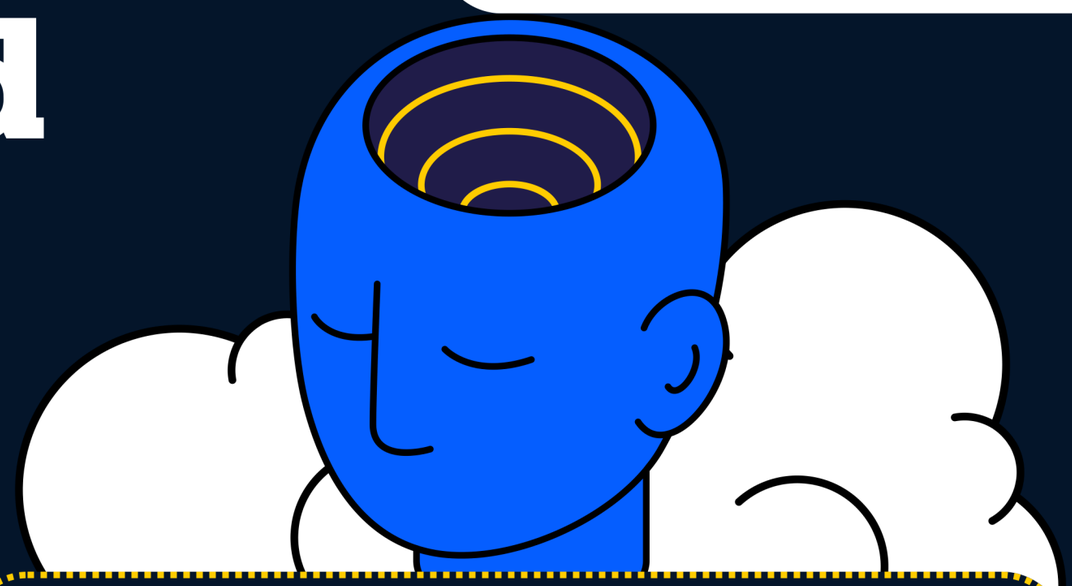
AN ALGORITHM'S OBJECTIVE IS TO PROVIDE AN ORDERED AND EFFICIENT METHOD OF SOLVING ISSUES.

JUST LIKE FOLLOWING A RECIPE, ALGORITHMS AUTOMATE TASKS, DO MATH, MAKE CHOICES, AND HANDLE INFORMATION. THEY'RE CRUCIAL IN MANY AREAS, LIKE TECHNOLOGY, SCIENCE, AND EVERYDAY LIFE.

Mark Clement Fernandez

Mark Clement Fernandez

Steps Involved in Problem Solving:



Understanding the Problem

1

The first thing you need to do before designing an algorithm is to understand completely the problem given.

Ascertaining the Capabilities of the Computational Device

2

You need to ascertain the capabilities of the computational device the algorithm is intended for.

Choosing between Exact and Approximate Problem Solvinge

3

The next principal decision is to choose between solving the problem exactly or solving it approximately. In the former case, an algorithm is called an exact algo-rithm; in the latter case, an algorithm is called an approximation algorithm.

Algorithm Design Techniques

4

An algorithm design technique (or “strategy” or “paradigm”) is a general approach to solving problems algorithmically that is applicable to a variety of problems from different areas of computing.

Designing an Algorithm and Data Structures

5

The next principal decision is to choose between solving the problem exactly or solving it approximately. In the former case, an algorithm is called an exact algo-rithm; in the latter case, an algorithm is called an approximation algorithm.

Methods of Specifying an Algorithm

6

Once you have designed an algorithm, you need to specify it in some fashion. In Section 1.1, to give you an example, Euclid’s algorithm is described in words (in a free and also a step-by-step form) and in pseudocode.

Proving an Algorithm’s Correctness

7

Once an algorithm has been specified, you have to prove its correctness. That is, you have to prove that the algorithm yields a required result for every legitimate input in a finite amount of time.

Analyzing an Algorithm

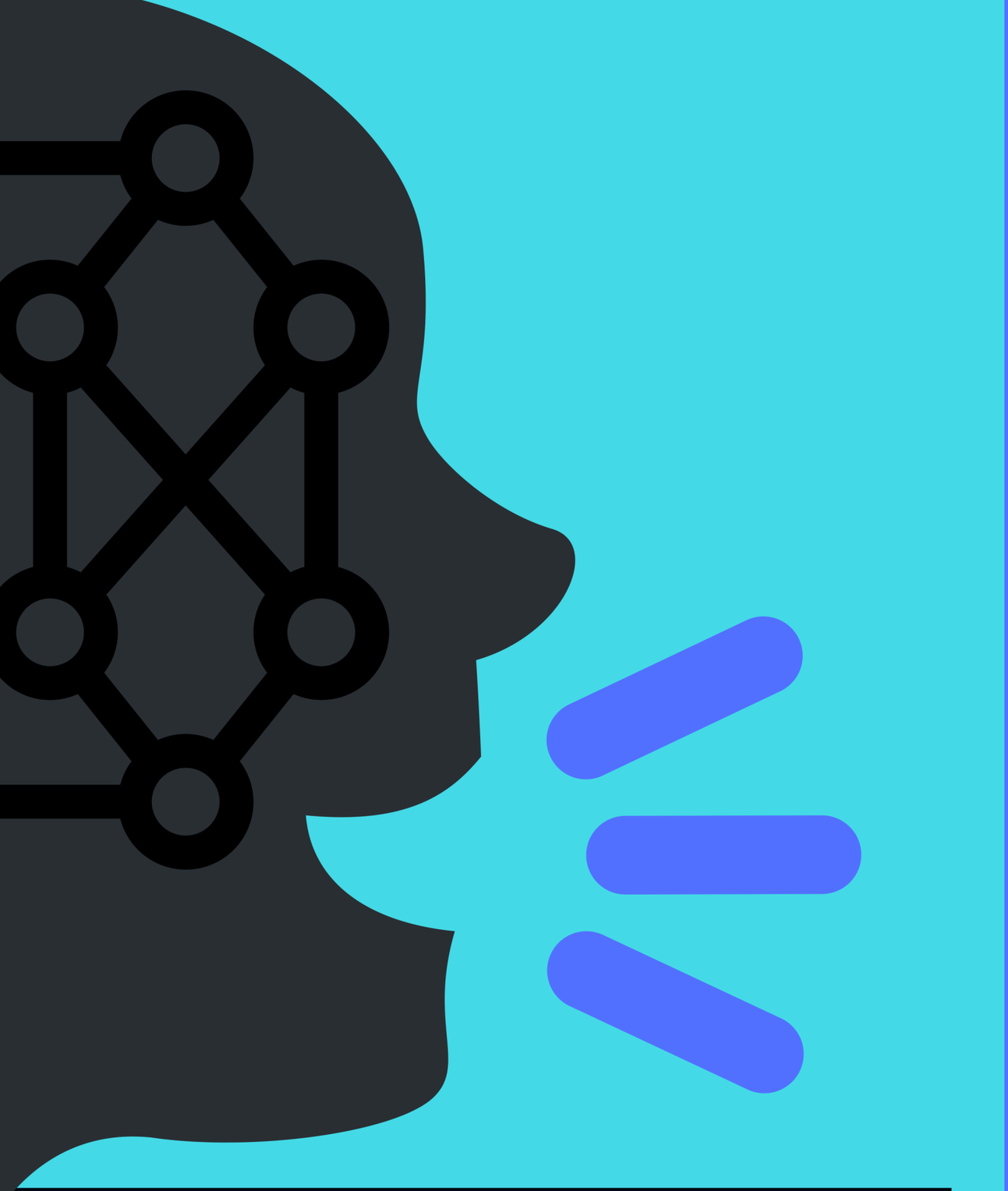
8

We usually want our algorithms to possess several qualities. After correctness, by far the most important is efficiency. In fact, there are two kinds of algorithm efficiency: time efficiency, indicating how fast the algorithm runs, and space ef-ficiency, indicating how much extra memory it uses.

Coding an Algorithm

9

Most algorithms are destined to be ultimately implemented as computer pro-grams. Programming an algorithm presents both a peril and an opportunity. The peril lies in the possibility of making the transition from an algorithm to a pro-gram either incorrectly or very inefficiently.



ALGORITHMIC THINKING

Imagine you're solving a big puzzle. Algorithmic thinking is like breaking the puzzle into smaller pieces, step by step. People use it in computer stuff and engineering because it's organized and makes sense. It's like planning out each move in a video game.

CONVENTIONAL THINKING

Think of conventional thinking as using your smarts and gut feeling. You don't always follow a set plan, but you rely on what you've learned before. It's great for making decisions when there's no clear rulebook, like when you're being creative or figuring out personal stuff.

Mark Clement Fernandez

COMPONENTS OF AN ALGORITHM

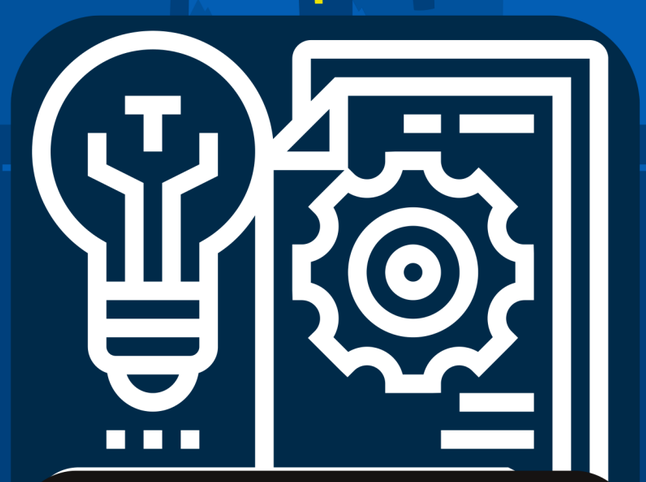
Mark Clement Fernandez

INPUT

This is the information or data that the algorithm starts with. It's what the algorithm works on to produce a desired output.

OUTPUT

The result that the algorithm produces after processing the input. It's what you expect to get after running the algorithm.



INSTRUCTIONS/STEPS

These are the detailed actions or operations that the algorithm performs on the input to generate the output. Each step needs to be well-defined and specific.

CONTROL STRUCTURES

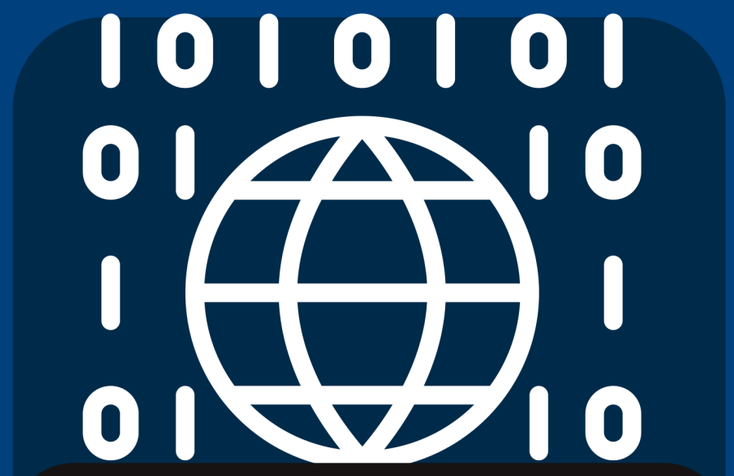
These are the decision-making elements of the algorithm. The determine what actions to based on certain conditions. Thi might include things like loops (for repeating actions) and conditi al statements (if this, then that).

VARIABLES

These are placeholders for storing temporary data or values that the algorithm needs during its execution.

DATA STRUCTURES

These are ways to organize and store data efficiently. They can include things like arrays, lists, and dictionaries.



COMMENTS

Explanatory notes within the algorithm's code that help humans understand what the code is doing. Comments are not executed by the computer.

ERROR HANDLING

Prepares for and manages errors or unexpected situations.

TERMINATION CONDITION

Signals when the algorithm should stop to avoid endless loops.

FLOWCHART/PSEUDOCODE

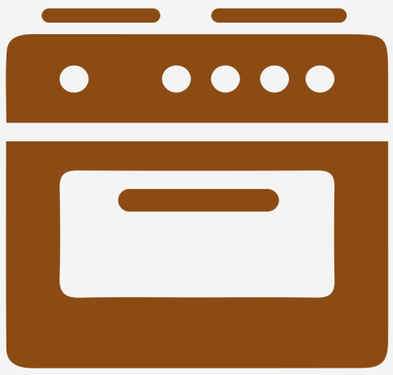
Prepares for and manages errors or unexpected situations.

Algorithm For Baking a Cake

Mark Clement Fernandez

GATHER INGREDIENTS

Collect flour, sugar, eggs, butter, baking powder, and other needed items.

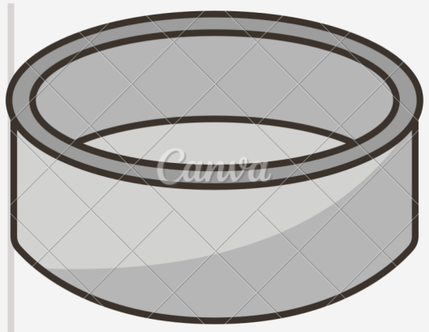


PREHEAT OVEN

Set the oven to the correct baking temperature.

MIX DRY AND WET

Combine dry ingredients in one bowl, beat eggs and mix wet ingredients in another.

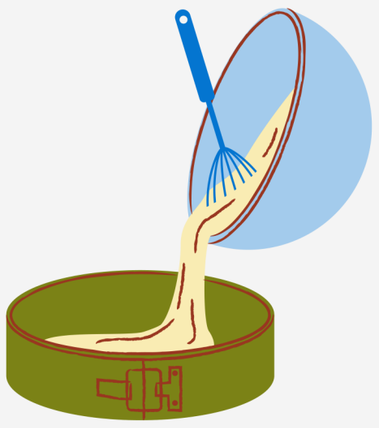


PREPARE PANS

Grease and flour the cake pans to prevent sticking.

BLEND MIXTURES

Gradually mix wet ingredients into dry until smooth.

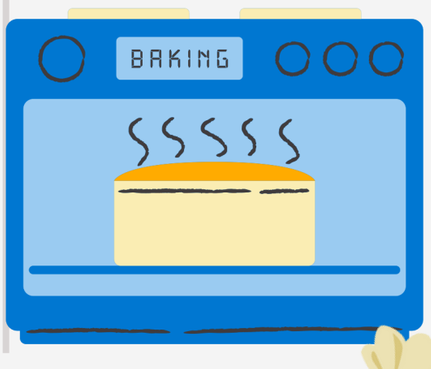


POUR INTO PANS

Distribute the batter evenly into the prepared pans.

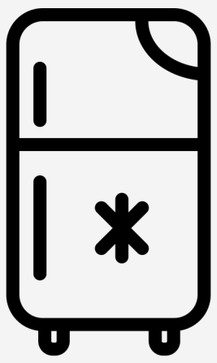
CHECK DONENESS

Insert a toothpick to check if cakes are fully baked.



BAKE

Place pans in the oven and bake for the recommended time.



COOL AND FROST

Let cakes cool, then frost with desired icing.

DECORATE AND SERVE

Add decorations, and your cake is ready to enjoy!