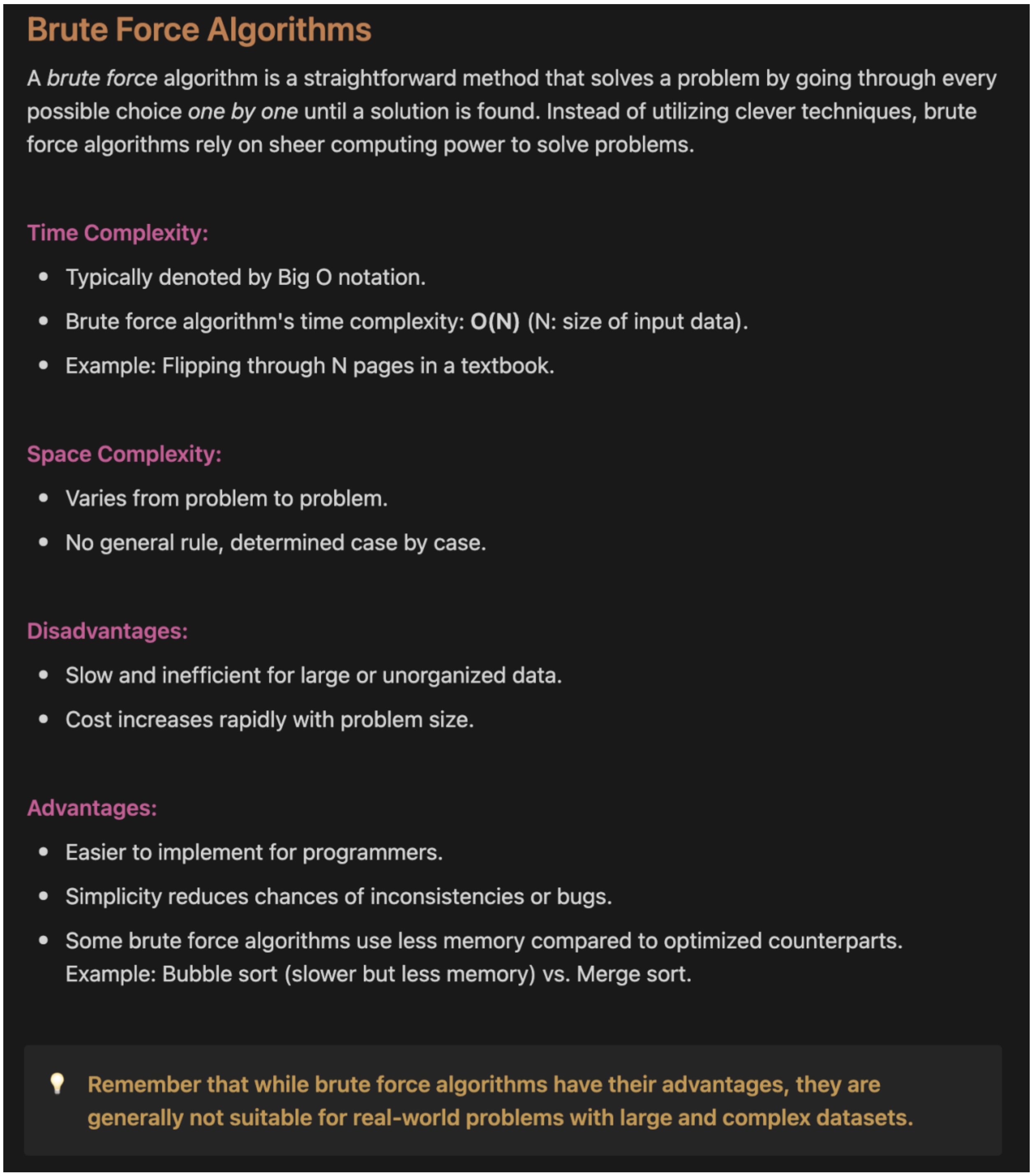

Brute Force Algorithms

A *brute force* algorithm is a straightforward method that solves a problem by going through every possible choice *one by one* until a solution is found. Instead of utilizing clever techniques, brute force algorithms rely on sheer computing power to solve problems.

Time Complexity:

- Typically denoted by Big O notation.
- Brute force algorithm's time complexity: **O(N)** (N: size of input data).
- Example: Flipping through N pages in a textbook.

Space Complexity:

- Varies from problem to problem.
- No general rule, determined case by case.

Disadvantages:

- Slow and inefficient for large or unorganized data.
- Cost increases rapidly with problem size.

Advantages:

- Easier to implement for programmers.
- Simplicity reduces chances of inconsistencies or bugs.
- Some brute force algorithms use less memory compared to optimized counterparts. Example: Bubble sort (slower but less memory) vs. Merge sort.

💡 **Remember that while brute force algorithms have their advantages, they are generally not suitable for real-world problems with large and complex datasets.**