Let's simplify things. Asymptotic notation is like a tool that helps us compare how fast different computer programs can solve problems as the problems get bigger and bigger. Imagine you have two superheroes, SuperFastMan and SuperSlowMan, and they both want to solve a puzzle. The puzzle has different levels, and as they go to higher levels, the puzzle becomes bigger and more challenging.
1. Big O notation (O): Let's say SuperFastMan can solve a puzzle in no more than 5 minutes, no matter how big it is. We write this as O(5 minutes). It means his speed to solve the puzzle is like a "speed limit" or an "upper bound." Example: O(5 minutes) means that the time it takes to solve the puzzle is never worse (never longer) than 5 minutes, no matter how big the puzzle is.

2. Omega notation (Ω): Now, let's consider SuperSlowMan, who takes at least 10 minutes to solve a puzzle, no matter how small it is. We write this as Ω(10 minutes). It means his speed to solve the puzzle is like a "slowest speed" or a "lower bound." Example: Ω(10 minutes) means that the time it takes to solve the puzzle is never better (never shorter) than 10 minutes, no matter how small the puzzle is.

3. Theta notation (Θ): Finally, let's talk about a third superhero, SuperAverageMan, who always takes around 7 minutes to solve the puzzle. We write this as Θ(7 minutes). It means his speed to solve the puzzle is like a "best guess" or an "average speed." Example: Θ(7 minutes) means that the time it takes to solve the puzzle is neither worse nor better, but it's around 7 minutes, no matter how big or small the puzzle is.

These notations help computer programmers decide which superhero (algorithm) is the best for a specific puzzle (problem) based on their speed (efficiency) for large puzzles (big data). So, when programmers talk about Big O, Omega, and Theta notation, they are just comparing superheroes' abilities to solve puzzles of different sizes