Name: Cano, Marc Allen E. Code: 7599

Course: CPE 223/L

Questions:

1. What are the advantages of recursion over iteration?

Programming recursion is the process by which a function calls itself to resolve smaller examples of an issue. One of its primary benefits is that it offers a more user-friendly and elegant way to solve recursive structure issues like tree traversal, graph algorithms, as well as divide and conquer strategies. Recursion can improve the readability of code much simpler to put into practice, particularly when handling intricate issues that can be divided into more manageable, related subproblems. Furthermore, recursive solutions frequently need less code than their iterative equivalents, which makes them more brief.

2. How can recursion be optimized in large-scale problems?

Although recursion might be graceful, it can also result in excessive memory usage because deep call stacks. In order to maximize recursion, strategies like dynamic, it is possible to use programming. Memorization saves previously calculated outcomes to avoid unnecessary computations, increasing effectiveness in issues such as Fibonacci sequences or algorithms for pathfinding. Tail recursion optimization can also be used in certain programming languages to minimize the use of stacks. For situations of a very vast scale where the recursive method becomes an iterative one when recursion depth becomes a problem. A better way to prevent stack overflow errors might be to use loops.

Observation

Recursion provides an organized and natural method for solving issues that are inherently recursive characteristics, but it may have performance issues, especially with regard to memory usage. Recursion makes code easier to read and understand, but iteration is frequently more effective because of its improved performance and less memory footprint for managing big datasets. The one that depending on the particular issue, the decision between recursion and iteration should prioritize efficiency for performance-critical applications and readability for problems that are inherently recursive.