WRITEUP

DEFINITION:-

ITERATION :- Executing code statements in a program until a task is done.

RECURSION :- Calling a function multiple times till a particular base case or sentinel is reached.

Complexity:-

-> **RECURSION**:- The time complexity of recursion is generally exponential. Recursion has a large amount of Overhead (Computation time)as compared to Iteration due to repetitive calling of the same function. Hence, the time complexity of the code is also increases.

Due to some mistake in the base condition, the program can get stuck in an infinte loop as the condition may never become false and hence will keep calling the function. This in turn may lead to system CPU crash.

-> **ITERATION**:- Iteration does not involve an overhead. The time complexiy of recursion is logarithmic or polynomial.
Due to mistake in iterator assignment,sentinel or increment/decreament leads to infinite loops that may or may not lead to system errors. Although, they will surely stop the program execution.

-> **IMPLEMENTATION IN OUR PROGRAM:-**

Our program is broken down in 2 sections:-

1. Recursive
2. Iterative

The recursive function uses stack memory which is allocated by the compiler and is fixed. This means that there could be stack overflow is we're not careful about the maximum recursive calls which will be made to a function in our program. This means, that there can be a stack overflow if it makes excessive recursive calls as the space allocated to the stack may not be enough in such cases.

For the iterative function that we used, we explicitly defined stacks for our use and enabled those stacks to have memory dynamically allocated to them.
This is possible as the space allocated is implemented on the heap space and is allocated to the program by the compiler at run time.
Hence we do not need to worry about stack overflowing as the memory allocation is not fixed.