***PROJECT TITLE:***

Multiplication Table Generator using C program.

***PROJECT DESCRIPTION:***

C program to generate multiplication table within the specified lower and upper limits using the concepts of loops (for and nested loop)

***VARIABLE DESCRIPTION:***

|  |  |  |
| --- | --- | --- |
| **VARIABLE NAME** | **DATATYPE** | **USE** |
| lowerLimit | int | stores the lower limit |
| upperLimit | int | stores the upper limit |
| number | int | counter variable for the 1st loop |
| i | int | counter variable for the 2nd loop |
| row | int | counter variable for the nested loop |
| col | int | counter variable for the nested loop |

***PROGRAM WORKFLOW & EXPLANATION:***

1. Initialization:
   * The program begins by including the standard input-output library (stdio.h) and declaring the main function.
2. User Input:
   * Two integer variables, lowerLimit and upperLimit, are declared to store user-defined lower and upper limits for the multiplication table.
   * The program prompts the user to input the lower limit by displaying the message "Enter the lower limit:" and reads the input into the lowerLimit variable using scanf.
   * Similarly, the program prompts the user to input the upper limit by displaying the message "Enter the upper limit:" and reads the input into the upperLimit variable using scanf.
3. Multiplication Table Generation:
   * The program enters a nested loop structure to generate the multiplication table. It uses two nested for loops, one for rows and another for columns, ranging from the lowerLimit to the upperLimit.
   * Inside the nested loops, for each combination of row and col, the program calculates the product of the numbers (i.e., row \* col) and displays it with proper formatting using printf. The format is "row \* col = product," followed by a tab character for spacing.
   * After each row is printed, a newline character (\n) is added to move to the next row in the table.
4. Program Termination:
   * The program continues this process until all combinations of numbers in the specified range have been printed.
   * Finally, the program returns 0 to indicate successful execution and terminates.

***SAMPLE OUTPUT:***

A screenshot of a computer

Description automatically generated