\*\*Project Title\*\*: C-based Custom Utility Function Implementation

\*\*Project Overview\*\*:

This project demonstrates proficiency in C programming through the development of various utility functions. The custom functions are designed to solve real-world problems, such as determining pass/fail status based on grades and attendance, calculating the cube of odd numbers, generating context-specific output for different numeric inputs, estimating paint requirements, and controlling output formats. Each function is implemented in a modular format using header files, making the code easily maintainable and reusable.

\*\*Key Functions\*\*:

1. \*\*Grading System\*\* (`grader`):

Implements logic to determine if a student passes or fails based on exam averages, homework averages, and attendance. The function encapsulates conditional statements to assess performance.

2. \*\*Cube Calculation of Odd Numbers\*\* (`cubeOfOdd`):

Takes an integer as input and computes the cube of all odd numbers up to that integer, displaying results in sequence.

3. \*\*Prime and Modulo-based Output Function\*\* (`introToCS330`):

A unique function that prints custom strings based on divisibility tests, such as checking divisibility by 3 or 7, and primality checks for educational use.

4. \*\*Paint Estimator\*\* (`paintGallons`):

Computes the number of gallons of paint required to cover the surface area of a room, using dimensions of length, width, and height. The function ensures accurate calculation by rounding up the required gallons.

5. \*\*Custom Print Function\*\* (`printHELLO`):

Iteratively prints numbers from 1 to `n`, replacing powers of two with the word "HELLO" to demonstrate the use of bitwise operators and control flow.

\*\*Technologies Used\*\*:

- C Programming

- Modular Coding with Header Files

- Control Structures (conditional statements, loops)

- Mathematical Calculations (cube, prime number checking, area calculation)

- Bitwise Operations

\*\*Skills Demonstrated\*\*:

- Proficient use of C syntax and structure.

- Problem-solving using mathematical and logical functions.

- Ability to modularize code for reusability and readability.

- Understanding of basic algorithms for prime checking and divisibility.

- Implementing real-world applications like grading systems and estimations.

This project showcases practical coding skills suitable for systems programming, algorithm development, and software utility creation, making it ideal for internship opportunities in software engineering and related fields.