# ECEC-201 Documentation for Term Project 1

Documentation is important in this project and it is worth 10 points out of 25

Outside the code

(i) State whether you worked alone or together with a partner on the project. Give your name(s) and student ID(s).
Worked in a group. Group Member(s): Kahan Patel 14638615, Gary Chen 14600255

**(ii)** A statement to the effect that the project is exclusively your work and you have completed the project without getting assistance from any other persons (other than those mentioned in no (i) above) within or outside the class.
This project is the work of the developer(s) listed above. No assistance was received from unauthorized persons other than those mentioned above.

(iii) A statement to the effect that you have not used any AI tools
No AI tools were used in the development of this code.

(iv) If you worked with a partner, the division of responsibilities with reference to ( a) Code development (b) Code testing (c) Documentation

Kahan Patel: Code Development & Testing & Documentation | Gary Chen: Code Development & Testing & Documentation

Signed: Kahan Patel, Gary Chen

**The above statements (i) to (iv) should be signed by one or both developers as the case may be. Violation of any of them may result in disciplinary action.**

(v) A brief description of the project and its practical application
This program calculates the estimated tax payable for the year 2025 based on user inputs. It includes filing status selection, income input, and tax calculation using the 2025 tax brackets. The program is designed to be user-friendly and provides clear output of the tax calculations. The tax calculation follows the Qualified Dividends and Capital Gain Tax Worksheet for 2025.

(vi) The features of C language which have been especially useful in developing the code. Mention chapter and section numbers in the textbook

The following C features & libraries were used:

- Standard I/O library for input/output operations (Ch. 2 & 3) (Printf (3.1), Scanf (3.2))
- Math library for mathematical operations (e.g., fmax) (Ch. 4) (Arithmetic Operators (4.1), Assignment Operators (4.2), Increment/Decrement Operators (4.3), Expressions (4.4 & 4.5))
- Functions for modular code organization (Ch. 9) (Defining/Call Functions (9.1), Function Declarations (9.2), Arguments (9.3), Return Values (9.4)) Constants for standard deductions and tax brackets
- Conditional statements for filing status and tax bracket selection (Ch. 5) (Logical Expressions (5.1), If-Else Statements (5.2), Switch Statements (5.3))
- Loops for input validation (Ch. 6) (While Loops (6.1), For Loops (6.3), Exiting Loops (6.4))
- Basic data types (int, double) for variable declarations (Ch. 7) (Integers (7.1), Variable Declarations (7.2), Type Definitions (7.5))
- Program Organization (Ch. 10) (Local Variables (10.1), External Variables (10.2), Block Statements (10.3), Scope of Variables (10.4))
- Pointers for passing variables to functions (Ch. 11) (Pointers (11.1), Pointer Addressing (11.2), Pointer Assignment (11.3), Pointer as Argument (11.4))
- Strings for UI prompts (Ch.13) (String Literals (13.1)

(vii) A brief description of the code you have developed and tested

The code uses the following chapters listed above to create a program that can take user input and calculate the tax based on the 2025 tax brackets. The program passes all test cases given in the program instructions. Edge-case testing was also done handle user-inputs that are not valid. The program responds dynamically to user inputs and provides the correct tax calculation based on the 2025 tax brackets. The program is modular and organized, making it easy to read and maintain.

(viii) A discussion of test results and how you verified them. Submit screen shots of results.

The code was tested via hand calculation. The first testcase was done by hand the answer was verified via the answer provided by professor. The result of the program was then compared to the hand calculation. Since the results were identical, it verified the results of the program. Moreover, the other test cases were complied and tested with the results the professor provided. All tests gave valid results. Thus, proving that validity of the results. Screenshots of the results and tests are provided in the assignment submission.

(ix) Detailed instructions for the user of your code

Instructions for running the program:

1. Compile main.c using a C compiler (e.g., gcc) (gcc -o main main.c). Note* if you experience a linker error include the the -lm flag during
the compilation (gcc -o main main.c -lm).
2. Run the compiled program (./main).
3. The program will display a menu for selecting the filing status. Navigation is handled via keyboard inputs (1-4).
4. Enter the income details as prompted. (double)

(x) An estimate of the man-hours spent to (a) develop the code (b) test the code

Man hours spent: Kahan 6 hours, Gary Chen: 5 hours

<u>Within the code</u>

(xi) Code should not be monolithic i.e. just a main program. It should be divided into functions each with a specific purpose which should be briefly explained by means of comment lines suitably placed. Prototypes of each function should be provided before the main.

- The program uses multiple functions with prototypes declared at the top of the program. The main function serves as a driver function with a high-level of abstraction for easy readability.

(xii) Code should be indented properly using tabs

- The program follows general C99 indenting & formatting. Snake case and caps were used for variables and #defines (constants).

(xiii) Prompts should be provided for the user to enter data by means of printf statements wherever needed

- The following requirement is met. Both printf and scanf are used for general I/O.

(xiv) Extra instructions planted for the purpose of debugging only should be completely removed and not merely commented out

- All unnecessary code snippets have been deleted.