ECEC-201 Documentation for Term Project 2

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 converts calendar information from numeric format to written/verbal format. The program incorporates the usage of command-line arguments and basic error detection for more responsive UI experience.

(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) , 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), String Variables (13.2), Reading and Writing Strings (13.3), C string library (13.5), Array of Strings (13.7))
- Advance use of Pointers (Ch. 17)

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

The code uses the following chapeters listed above to create a program that can take user input via the command line and convert the input into written/verbal format of the calendar information. The program passes all test cases given in the program instructions. Edge-case testing was also done to ensure the program is robust and handles invalid inputs gracefully. The program is modular and uses functions to validate the date input, ensuring that it is a valid date before converting it to the written format. The program also handles leap years correctly.

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

The code was tested using the provided test cases and the conversions were also done by hand to validate of a given date was valid. After doing both the results of the program was then compared to the hand calculation. Since the results were identical, it verified the results of the program. Moreover, all the other test cases were also complied and tested. All test cases gave valid results. Thus, proving the validity of the program. 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. Download project2B.c file
2. Install a C compiler (e.g., GCC)
3. Open a terminal or command prompt
4. Navigate to the directory where project2B.c is located
5. Compile the program using the command: gcc project2B.c -o project2B or clang project2B.c -o project2B (depending on your compiler)
6. Run the program using the command: ./project2B <Month> <day> <year>
   - For example: ./project2B January 31 2023

(x) An estimate of the man-hours spent to (a) develop the code (b) test the code
Man hours spent Kahan Patel: 3 hours, Gary Chen: 3 hours

Test Cases Required (15 test cases each worth 1 point)
Test cases should be numbered with extra command line arguments like

./Project2A 11 18 2005 Testcase 1

Or

./Project2B July 31 1999 Testcase 9

The extra arguments should be counted in argc but their argv's can be ignored in the code

Part 1 mm dd yyyy to month dd yyyy

1. 11 18 2005 Input: 11 18 2005 test case 1 Output: November 18, 2005

2.	04 31 1978	Input: 04 31 1978 test case 1	Output: Invalid
3.	08 32 1889	Input: 08 32 1889 test case 1	Output: Invalid
4.	13 12 2020	Input: 13 12 2020 test case 1	Output: Invalid
5.	02 29 1996	Input: 02 29 1996 test case 1	Output: February 29, 1996
6.	02 29 1877	Input: 02 29 1877 test case 1	Output: Invalid
7.	02 29 1700	Input: 02 29 1700 test case 1	Output: Invalid
8.	02 29 2000	Input: 02 29 2000 test case 1	Output: February 29, 2000

Part 2 month dd yyyy to mm dd yyyy

9.	July 31 1999	Input: July 31 1999	Output: 07 31 1999
10.	october 7 1988	Input: october 7 1988	Output: 10 07 1988
11.	June 31 2017	Input: June 31 2017	Output: Invalid
12.	February 28 2016	Input: February 28 2016	Output: 02 28 2016
13.	Feb 29 2000	Input: Feb 29 2000	Output: Invalid
14.	Feb 29 1800	Input: Feb 29 1800	Output: Invalid
15.	February 29 1971	Input: February 29 1971	Output: Invalid