ENGG1110 Problem Solving by Programming (2024-2025 Term 2)

Deadline: 24-Apr-2025 (Thu) 23:59

---

1. Introduction

Teeko is a game played by two players, while the game is divided into two phases.

- In phase 1, the **placement phase**, players take turns in placing his/her four marks (a black circle or a red cross checker) on an empty square of a 4x4 board:
    a. **the first one** who can have four of his/her marks in ***a horizontal row, a vertical column, a diagonal, or a square block*** on the board is the **winner**, leading to game over;
    b. if no one wins after all eight marks are placed, it is a **draw game**, thus game over too.
- In phase 2, the **movement phase**, the program reads a draw game configuration to begin with:
    a. Players take turns in moving one of his/her four marks, in eight possible directions, onto an empty square. The game ends until a player wins, thus no draw game is expected.

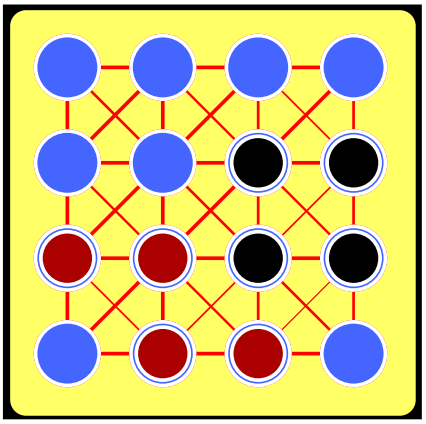


Figure 1 Sample game with black wins in a 2x2 square block; it may happen in either phase 1 or phase 2. [2]

In this project, the following game rules are used.

- The board is limited to a 4x4 one, of which all the squares are empty initially.
- Two players take turns to *manipulate* a circle (O) or a cross (X) on an unoccupied cell.
- Players cannot skip their turns. The game ends when a player wins, in phase 1 or phase 2.

[1] Wikipedia contributors, "Teeko," Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/wiki/Teeko (accessed March 3, 2025).
[2] *Teeko game, won by Black for placing four in a square by Mliu92 licensed via CC By-SA 4.0 https://en.wikipedia.org/wiki/Teeko#/media/File:Teeko_board_with_pieces_(var).svg*; reduced to 4x4 by CUHK.

2. Schedule

The suggested schedule is as follows:

Week	Lab Date	Tasks

11	Mar 19	• Complete the following functions:
		  ○ **initGameBoard()**
		  ○ **printGameBoard()**
		  ○ **placeMarkByHumanPlayer()**
		• In main():
		  ○ Test the above functions by uncommenting some given code
		• No submission is needed at this stage

12	Mar 26	• Complete the following functions:
		  ○ **hasWinner()**
		• In main():
		  ○ Complete the implementation of the game for two human players, including the change of the current player and the checking of whether the game ends
		• No submission is needed at this stage

13	Apr 2	• Complete the following function:
		  ○ **placeMarkByComputerPlayer()**
		• In main():
		  ○ Complete the implementation of the game for a human player and a computer player
		• Design your own test cases and perform the final testing of Project Part 1

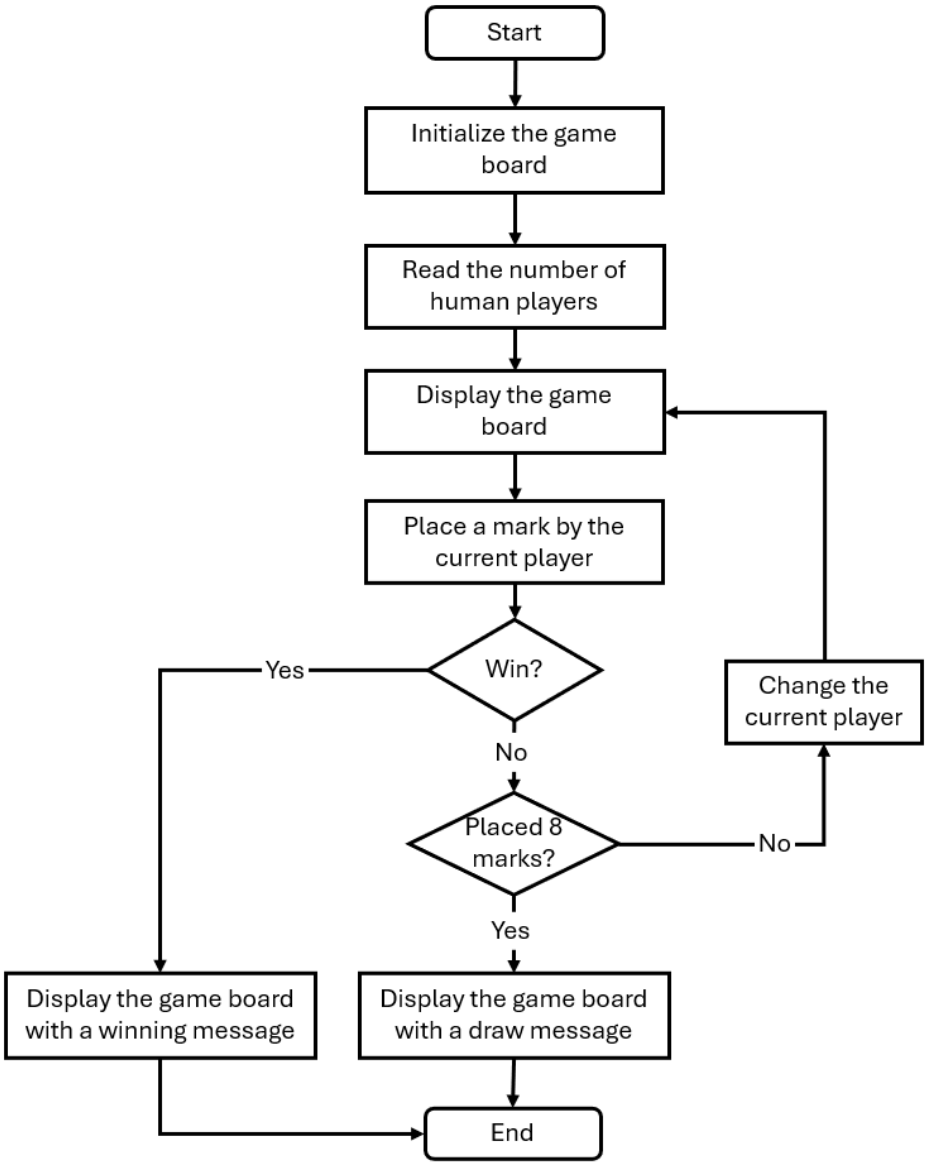
14	Apr 9	• Complete the following functions:
		  ○ **initGameBoardFromFile()**
		  ○ **moveMarkByHumanPlayer()**
		• In main():
		  ○ Test the above functions by uncommenting some given code

15	Apr 16	• Complete the following function: ○ **moveMarkByComputerPlayer()** • Design your own test cases and perform the final testing of Project Part 2 • Submit all your code

3. Project Part 1 – Placement Phase

- In Project Part 1, your task is to implement phase 1 of game of Teeko for one or two human players. **If there is only one human player, another player is the computer player.**
- You are required to complete the given **ProjectPart1_1155xxxxxx.c** without modifying the existing code. Please rename it to **ProjectPart1_yourStudentID.c** like `ProjectPart1_1155123456.c` .

The program flow in Part 1 is shown as follows:



4. Detailed Program Design (Part 1)

4.1 Header Files, Variable Declarations and Macros

Apart from <stdio.h>, do NOT use definitions/functions/macros from other sources. In addition, no global variables (those declared outside functions) are allowed. That is, all variables must be declared inside functions, thus local. Information must be exchanged through parameter passing and return value.

The following macros are defined in the beginning of the given source code:

```
/* Macros (symbolic constants) used to represent the state of each square */
#define EMPTY   0
#define CIRCLE  1
#define CROSS   2
```

How to use the above macros? For example, we have the following line in subsequent part of the source code:

```
gameBoard[i][j] = EMPTY;
```

During preprocessing of the compile/build process, the original source line above will be replaced by

```
gameBoard[i][j] = 0;
```

automatically before the source code compiles. The advantage of using macro is that we do not have to memorize 0, 1, and 2 represent an empty square, a circle mark, and a cross mark respectively. This can also reduce the chance of having typos. Such macros are usually in all CAPS and defined at the beginning of a source file.

4.2 Read Number of Human Players

The program asks the number of human players with the following prompt and then reads the input:

```
How many human players [1-2]?
```

This is handled in the **main()** function. You can assume that the user must input either 1 or 2.

- If there are two human players, they are Player 1 and Player 2.
- If there is only one human player, he/she is Player 1, and another player is the computer player.

For both cases, **Player 1 goes first and places the CIRCLE mark**, while Player 2 (or the computer player) places the CROSS mark.

4.3 Initialize Game Board

The **main()** function invokes the **initGameBoard()** function, which initializes the game board by setting all sixteen squares to EMPTY.

4.4 Print Game Board

The **main()** function invokes the **printGameBoard()** function, which displays the game board on the screen.

- If the square is empty, display a number between 1 to 16, with 1 at the bottom-left corner and 16 at the top-right corner (see the following diagram for the exact number arrangement).
- If the square is marked by a circle, display 'O' (big-oh).
- If the square is marked by a cross, display 'X'.

Some examples are shown as follows:

All sixteen squares are empty:	``` ================ |13||14||15||16| | 9||10||11||12| | 5|| 6|| 7|| 8| | 1|| 2|| 3|| 4| ================ ```
After four moves:	``` ================ |13||14||15||16| | 9|| O|| O||12| | 5|| X|| X|| 8| | 1|| 2|| 3|| 4| ================ ```
After eight moves:	``` ================ |13||14||15||16| | O|| O|| O||12| | X|| X|| X|| O| | X|| 2|| 3|| 4| ================ ```

You are required to follow exactly the above output format. Using other output formats here will result in mark deduction.

4.5 Place Mark by Human Player

The program prints one of the following prompts in the **main()** function:

```
Player 1, please place your mark [1-16]:
```

```
Player 2, please place your mark [1-16]:
```

and then invokes the **placeMarkByHumanPlayer()** function, which asks the human player to place the mark. You can assume that the user input must be valid (i.e., an empty space between 1 and 16).

4.6 Check Winner

The **main()** function invokes the **hasWinner()** function, which returns 1 if there is a winner in the game; otherwise, it returns 0. If there is a winner, the winner is the current player indicated in the **main()** function and one of the following messages is printed:

```
Player 1 wins! Congratulations!
```

```
Player 2 wins! Congratulations!
```

```
Computer wins!
```

4.7 Check Draw Game

The **main()** function checks if all eight marks are placed without a winner. If so, it is a draw game in this phase, the following message is printed in the **main()** function:

```
Draw game!
```

4.8 Place Mark by Computer Player

The program prints the following message in the main() function:

```
Computer places the mark:
```

and then invokes the **placeMarkByComputerPlayer()** function, which determines the next move of the computer player. In Project Part 1, you are required to implement the following simple strategy:

> *Place the mark in the first empty space scanning **from 1 to 16**, i.e., the available space with the lowest number.*

The program prints the chosen space number before the updated gameboard is printed.

4.9. Sample Run (Part 1)

Two sample runs are shown below (**underlined bold characters in blue** are user inputs). In addition to them, you are strongly recommended to test your program by designing your own test cases.

Sample Run 1

```
How many human players [1-2]?
2
===============
|13||14||15||16|
| 9||10||11||12|
| 5|| 6|| 7|| 8|
| 1|| 2|| 3|| 4|
===============
Player 1, please place your mark [1-16]:
6
===============
|13||14||15||16|
| 9||10||11||12|
| 5|| O|| 7|| 8|
| 1|| 2|| 3|| 4|
===============
Player 2, please place your mark [1-16]:
7
===============
|13||14||15||16|
| 9||10||11||12|
| 5|| O|| X|| 8|
| 1|| 2|| 3|| 4|
===============
Player 1, please place your mark [1-16]:
10
===============
|13||14||15||16|
| 9|| O||11||12|
| 5|| O|| X|| 8|
| 1|| 2|| 3|| 4|
===============
Player 2, please place your mark [1-16]:
2
===============
|13||14||15||16|
| 9|| O||11||12|
| 5|| O|| X|| 8|
| 1|| X|| 3|| 4|
===============
Player 1, please place your mark [1-16]:
9
===============
|13||14||15||16|
| O|| O||11||12|
| 5|| O|| X|| 8|
| 1|| X|| 3|| 4|
===============
Player 2, please place your mark [1-16]:
3
===============
|13||14||15||16|
| O|| O||11||12|
| 5|| O|| X|| 8|
| 1|| X|| X|| 4|
===============
Player 1, please place your mark [1-16]:
5
===============
|13||14||15||16|
| O|| O||11||12|
| O|| O|| X|| 8|
| 1|| X|| X|| 4|
===============
Player 1 wins! Congratulations!
```

Sample Run 2

```
How many human players [1-2]?
1
===============
|13||14||15||16|
| 9||10||11||12|
| 5|| 6|| 7|| 8|
| 1|| 2|| 3|| 4|
===============
Player 1, please place your mark [1-16]:
2
===============
|13||14||15||16|
| 9||10||11||12|
| 5|| 6|| 7|| 8|
| 1|| O|| 3|| 4|
===============
Computer places the mark:
1
===============
|13||14||15||16|
| 9||10||11||12|
| 5|| 6|| 7|| 8|
| X|| O|| 3|| 4|
===============
Player 1, please place your mark [1-16]:
5
===============
|13||14||15||16|
| 9||10||11||12|
| O|| 6|| 7|| 8|
| X|| O|| 3|| 4|
===============
Computer places the mark:
3
===============
|13||14||15||16|
| 9||10||11||12|
| O|| 6|| 7|| 8|
| X|| O|| X|| 4|
===============
Player 1, please place your mark [1-16]:
7
===============
|13||14||15||16|
| 9||10||11||12|
| O|| 6|| O|| 8|
| X|| O|| X|| 4|
===============
Computer places the mark:
4
===============
|13||14||15||16|
| 9||10||11||12|
| O|| 6|| O|| 8|
| X|| O|| X|| X|
===============
Player 1, please place your mark [1-16]:
6
===============
|13||14||15||16|
| 9||10||11||12|
| O|| O|| O|| 8|
| X|| O|| X|| X|
===============
Computer places the mark:
8
===============
|13||14||15||16|
| 9||10||11||12|
| O|| O|| O|| X|
| X|| O|| X|| X|
===============
Draw game!
```

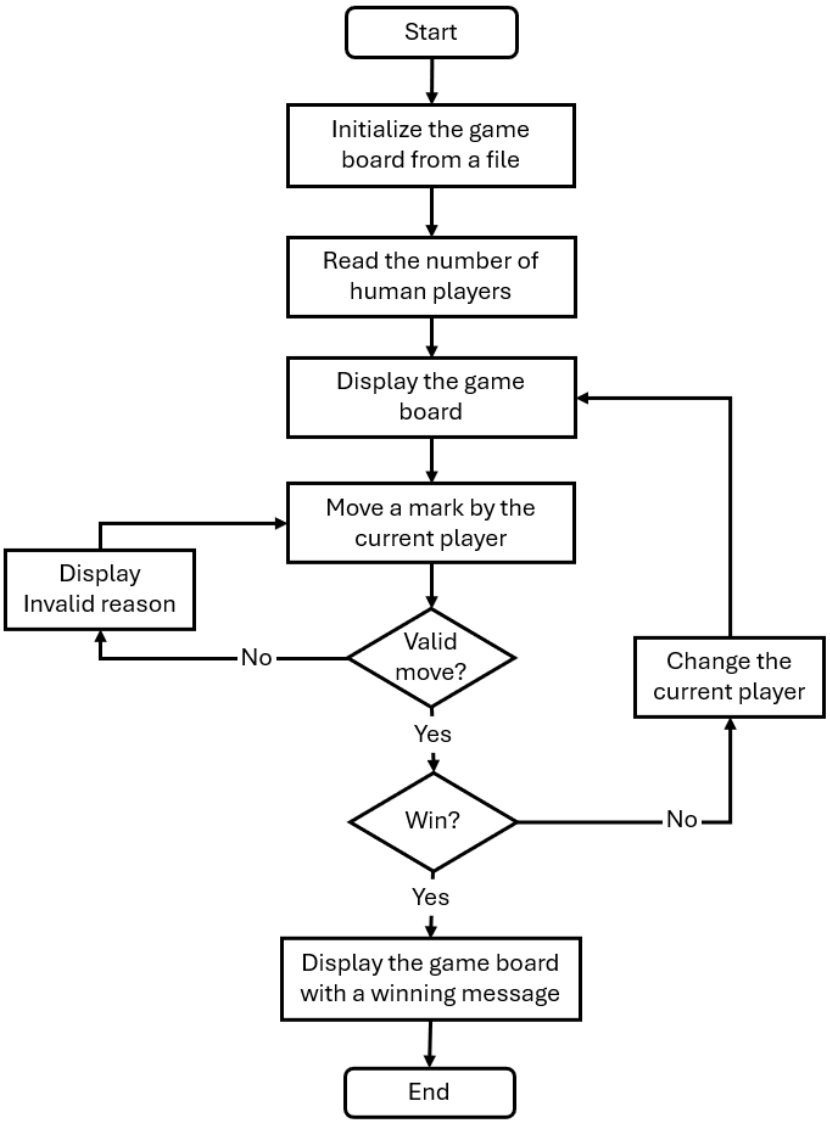
5. Project Part 2 – Movement Phase

In Project Part 2, your task is to implement phase 2 of the game Teeko for one or two human players. If there is only one human player, another player is the computer player.

You are required to:

a) copy your completed **ProjectPart1_yourStudentID.c**;
b) rename it to **ProjectPart2_your StudentID.c** like `ProjectPart2_1155123456.c`
c) add the extra functionalities, test, debug, finish and submit the work.

The program flow in Part 2 is shown as follows:



6. Detailed Program Design (Part 2)

6.1 Header Files, Variable Declarations and Macros

Governing rules on header files, variable declarations and macros remain the same as Part 1.

6.2 Initialize a Game Board from a file

The **main()** function invokes the **initGameBoardFromFile()** function, which initializes the game board by reading sixteen marks of the squares from a file named "gameboard.txt" . Assume the file format is correct and contains only 4 CIRCLE marks, 4 CROSS marks, and 8 EMPTY marks; the board must represent a draw game configuration. A sample "gameboard.txt" is show below:

A sample draw game board from **`gameboard.txt`**:	`0 0 0 0`
	`1 1 0 0`
	`2 1 0 0`
	`2 2 1 2`

6.3 Read Number of Human Players

Same as Part 1, the program asks the number of human players with the following prompt and then read the input:

```
How many human players [1-2]?
```

This is handled in the **main()** function. You can assume that the user must input either 1 or 2.

6.4 Print Game Board

Same as Part 1. The **main()** function invokes the **printGameboard()** function and prints the full game board.

6.5 Check Winner

Same as Part 1. The **main()** function invokes the **hasWinner()** function and prints the corresponding winning message.

6.6 Move Mark by Human Player

The program invokes the **moveMarkByHumanPlayer()** function, which asks the human player to select a mark for being moved by printing one of the following prompts in the **main()** function:

```
Player 1, please select your mark [1-16]:
```

```
Player 2, please select your mark [1-16]:
```

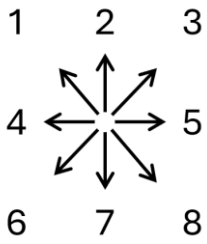
You can assume the user input must be between 1 and 16. However, if the player selected an empty space or a mark of the opponent, prints the following error message:

```
Wrong pick, try again.
```

The program goes back to ask the human player to select the mark again.

Once the selected mark is valid, the program continues and prints the following prompts in the **moveMarkByHumanPlayer ()** function:

```
Please select moving direction [1-8] or back [0]:
```



As shown above, 1 refers to the up-left direction, 5 refers to the right direction, etc.

The player can input 0 to go back, thus choosing the mark again, i.e., going back to the start of step 6.6.

You can assume the user input must be between 0 and 8. However, if the direction is invalid, the program prints one of the following error messages and asks the player to select moving directions again:

```
Out of bound, try again.
```

- The above message is shown when the target cell is out of the game board.

```
Occupied, try again.
```

- The above is shown when the target cell has been occupied with a mark already.

The program prints the following message after a valid move:

```
Mark moved.
```

6.7 Move Mark by Computer Player

The program prints the following message in the **main()** function:

```
Computer moves the mark:
```

and then invokes the **moveMarkByComputerPlayer()** function, which determines the next move of the computer player.

In Project Part 2, you are required to implement the following simple strategy:

1. choose the first placed (computer's X) mark on the draw-game board scanning **from 16 to 1**, i.e., reversely.
2. select the first valid direction scanning **from 8 to 1**; if all directions of the selected mark are invalid, go back to step 1 and choose the next placed mark.

The program prints the chosen space number, then the following message:

```
Direction
```

The program then prints the chosen direction number before the updated gameboard is printed.

6.8. Sample Run (Part 2)

Two sample runs are shown below (**underlined bold characters in blue** are user inputs). In addition to them, you are strongly recommended to test your program by designing your own test cases.

Sample Run 1

```
How many human players [1-2]?
2
===============
|13||14||15||16|
| O|| O||11||12|
| X|| O|| 7|| 8|
| X|| X|| O|| X|
===============
Player 1, please select your mark [1-16]:
6
Please select moving direction [1-8] or back [0]:
5
Mark moved.
===============
|13||14||15||16|
| O|| O||11||12|
| X|| 6|| O|| 8|
| X|| X|| O|| X|
===============
Player 2, please select your mark [1-16]:
4
Please select moving direction [1-8] or back [0]:
2
Mark moved.
===============
|13||14||15||16|
| O|| O||11||12|
| X|| 6|| O|| X|
| X|| X|| O|| 4|
===============
Player 1, please select your mark [1-16]:
3
Please select moving direction [1-8] or back [0]:
5
Mark moved.
===============
|13||14||15||16|
| O|| O||11||12|
| X|| 6|| O|| X|
| X|| X|| 3|| O|
===============
Player 2, please select your mark [1-16]:
8
Please select moving direction [1-8] or back [0]:
1
Mark moved.
===============
|13||14||15||16|
| O|| O|| X||12|
| X|| 6|| O|| 8|
| X|| X|| 3|| O|
===============
Player 1, please select your mark [1-16]:
9
Please select moving direction [1-8] or back [0]:
7
Occupied, try again.
Please select moving direction [1-8] or back [0]:
9
Occupied, try again.
Please select moving direction [1-8] or back [0]:
6
Out of Bound, try again.
Please select moving direction [1-8] or back [0]:
9
Out of Bound, try again.
Please select moving direction [1-8] or back [0]:
2
Mark moved.
===============
| O||14||15||16|
| 9|| O|| X||12|
| X|| 6|| O|| 8|
| X|| X|| 3|| O|
===============
Player 1 wins! Congratulations!
```

Sample Run 2

```
How many human players [1-2]?
1
===============
|13||14||15||16|
| O|| O||11||12|
| X|| O|| 7|| 8|
| X|| X|| O|| X|
===============
Player 1, please select your mark [1-16]:
3
Please select moving direction [1-8] or back [0]:
2
Mark moved.
===============
|13||14||15||16|
| O|| O||11||12|
| X|| O|| O|| 8|
| X|| X|| 3|| X|
===============
Computer moves the mark:
4
Direction
4
===============
|13||14||15||16|
| O|| O||11||12|
| X|| O|| O|| 8|
| X|| X|| X|| 4|
===============
Player 1, please select your mark [1-16]:
9
Please select moving direction [1-8] or back [0]:
3
Mark moved.
===============
|13|| O||15||16|
| 9|| O||11||12|
| X|| O|| O|| 8|
| X|| X|| X|| 4|
===============
Computer moves the mark:
5
Direction
2
===============
|13|| O||15||16|
| X|| O||11||12|
| 5|| O|| O|| 8|
| X|| X|| X|| 4|
===============
Player 1, please select your mark [1-16]:
14
Please select moving direction [1-8] or back [0]:
8
Mark moved.
===============
|13||14||15||16|
| X|| O|| O||12|
| 5|| O|| O|| 8|
| X|| X|| X|| 4|
===============
Player 1 wins! Congratulations!
```

7. Academic Honesty and Declaration Statement

Attention is drawn to University policy and regulations on honesty in academic work, and to the disciplinary guidelines and procedures applicable to breaches of such policy and regulations. Details may be found at https://www.cuhk.edu.hk/policy/academichonesty/.

Please place the following declaration statement as the comment in the beginning of your .c source code and fill in your information. Assignments without the properly signed declaration will not be graded.

Tools such as software similarity measurement, AI-tool fingerprint detection, etc., might be used to inspect all submissions.

```
/**
 * ENGG1110 Problem Solving by Programming
 *
 * Course Project
 *
 * I declare that the project here submitted is original
 * except for source material explicitly acknowledged,
 * and that the same or closely related material has not been
 * previously submitted for another course.
 * I also acknowledge that I am aware of University policy and
 * regulations on honesty in academic work, and of the disciplinary
 * guidelines and procedures applicable to breaches of such
 * policy and regulations, as contained in the website.
 *
 * University Guideline on Academic Honesty:
 *   https://www.cuhk.edu.hk/policy/academichonesty/
 *
 * Student Name  : <your name>
 * Student ID    : <your student ID>
 * Class/Section : <your class/section>
 * Date          : <date>
 */
```

8. Testing Platform

You shall work on **CLion** under **Windows 11/MacOS**.

9. Submission

Please follow the steps below to submit your work by the deadline specified on the first page.

1. Your work may be graded with the aid of Gradescope autograder, under markers' supervision and further inspection. Please note that while passing the test cases is necessary, it is not sufficient on its own for getting full marks. Additional marks will be allocated based on the completeness and adherence to all requirements specified in the project documentation, including code organization, functionality, and adherence to the design specifications.
2. Submit **ProjectPart1_yourStudentID.c** like `ProjectPart1_1155123456.c` and **ProjectPart2_your StudentID.c** like `ProjectPart2_1155123456.c` .
   - Note: No need to submit other files used in your IDE (if any)

The late submission penalty is as follows:

- Within 3 days (72 hours): 10% per 24 hours
- More than 3 days but within 7 days: 50% penalty
- More than 7 days: 100% penalty (you will receive 0 marks in the project AND fail the course)

NOTE: Finishing and submitting the project properly is crucial in PASSING the course.