# Hong Kong Institute of Vocational Education (Tsing Yi)
# Department of Information Technology

**HD in Game Software Development (IT114107)**
**ITP4704M C++ Programming and Data Structures**
**Assignment**

### Assignment Due Date and Time

14 July 2024 (Suday) 23:55 (via moodle)

## Introduction

Welcome to the "Logic Chess Track Game" assignment! This project aims to reinforce your C++ programming skills and understanding of data structures while developing a fun and challenging two-player board game.

## Learning Objectives

By completing this assignment, you will:

1. Practice using C++ programming concepts, including loops, conditional statements, functions, and classes, to implement the game logic.
2. Apply your knowledge of data structures, such as arrays and matrices, to represent and manipulate the game state.
3. Implement different data structures and algorithms using object-oriented programming techniques.
4. Develop problem-solving skills by breaking down the game requirements into smaller, manageable tasks and designing efficient algorithms to handle game mechanics.
5. Gain experience in implementing user input validation and error handling to ensure a smooth and robust gameplay experience.

By completing this assignment, you will not only reinforce your understanding of data structures and C++ programming but also develop valuable skills in problem-solving, code organization, and software development that will benefit you in your future academic and professional endeavors.

Let's dive in and start building the "Logic Chess Track Game"!

# Game Rules

## Game Setup

- The game is played on a 4x4 gameboard consisting of two tracks: an outer track and an inner track.
- Each player has 8 marbles of their assigned colour: Player 1 uses white marbles (represented by 'O'), and Player 2 uses black marbles (represented by 'X').
- The game begins with an empty gameboard.

## Gameplay

1. The game starts with Player 1's turn.
2. On each turn, the current player performs two actions:
    - Places one of their marbles on any empty position on the gameboard.
    - After placing the marble, all marbles on both the outer and inner tracks move one step in an anti-clockwise direction automatically.
3. Players take turns alternately until a winner is determined or the game ends in a draw.

## Marble Movement

- When a marble is placed on the gameboard, it occupies a position on either the outer track or the inner track.
- After each marble placement, all marbles on the gameboard (including the newly placed marble) move one step in an anti-clockwise direction automatically.
- Marbles on the outer track move to the next position on the outer track, while marbles on the inner track move to the next position on the inner track.
- The first marble on the track moves to the end of the track.

To illustrate the movement better, numbers and alphabets are used instead of marbles in the example below:

```
Before the move:
1 2 3 4
0 a b 5
B d c 6
A 9 8 7

After the move:
2 3 4 5
1 b c 6
0 a d 7
B A 9 8
```

## Winning Condition

- A player wins the game if, after a turn (including the marble movement), they have 4 of their marbles in a row either vertically, horizontally, or diagonally on the gameboard. *Note that, it is not necessarily a player won in their own turn.*
- If all positions on the gameboard are filled and no player has achieved a winning condition, the game ends in a draw.

Below is an example:

```
Player 1 wins with a vertical row of 4 white marbles:

   0 1 2 3
-+--------
0  _ O _ X
1  X O X _
2  _ O X _
3  O O _ _
```

## Invalid Moves

- Players can only place their marbles on empty positions on the gameboard.
- Attempting to place a marble on an already occupied position or outside the gameboard boundaries is considered an invalid move.
- If a player makes an invalid move, the player will be asked for an input again until a valid input is entered.

## Sample Input and Output

```
Player 1's turn
Please input row and col: 2 2

   0 1 2 3
-+--------
0  _ _ _ _
1  _ _ _ _
2  _ _ O _
3  _ _ _ _

moved:

   0 1 2 3
-+--------
0  _ _ _ _
1  _ _ O _
2  _ _ _ _
3  _ _ _ _
```

```
Player 2's turn
Please input row and col: 2 1

   0 1 2 3
-+--------
0  _ _ _ _
1  _ _ O _
2  _ X _ _
3  _ _ _ _

moved:

   0 1 2 3
-+--------
0  _ _ _ _
1  _ O _ _
2  _ _ X _
3  _ _ _ _


Player 1's turn
Please input row and col: 1 3

   0 1 2 3
-+--------
0  _ _ _ _
1  _ O _ O
2  _ _ X _
3  _ _ _ _

moved:

   0 1 2 3
-+--------
0  _ _ _ O
1  _ _ X _
2  _ O _ _
3  _ _ _ _


Player 2's turn
Please input row and col: 1 1

   0 1 2 3
-+--------
0  _ _ _ O
1  _ X X _
2  _ O _ _
3  _ _ _ _
```

```
moved:

   0 1 2 3
-+--------
0  _ _ O _
1  _ X _ _
2  _ X O _
3  _ _ _ _


Player 1's turn
Please input row and col: 1 2

   0 1 2 3
-+--------
0  _ _ O _
1  _ X O _
2  _ X O _
3  _ _ _ _

moved:

   0 1 2 3
-+--------
0  _ O _ _
1  _ O O _
2  _ X X _
3  _ _ _ _


Player 2's turn
Please input row and col: 1 3

   0 1 2 3
-+--------
0  _ O _ _
1  _ O O X
2  _ X X _
3  _ _ _ _

moved:

   0 1 2 3
-+--------
0  O _ _ X
1  _ O X _
2  _ O X _
3  _ _ _ _
```

```
Player 1's turn
Please input row and col: 0 1

   0 1 2 3
-+--------
0  O O _ X
1  _ O X _
2  _ O X _
3  _ _ _ _

moved:

   0 1 2 3
-+--------
0  O _ X _
1  O X X _
2  _ O O _
3  _ _ _ _


Player 2's turn
Please input row and col: 2 0

   0 1 2 3
-+--------
0  O _ X _
1  O X X _
2  X O O _
3  _ _ _ _

moved:

   0 1 2 3
-+--------
0  _ X _ _
1  O X O _
2  O X O _
3  X _ _ _


-------------------------------------------------------------
---
Player 1's turn
Please input row and col: 0 3

   0 1 2 3
-+--------
0  _ X _ O
1  O X O _
2  O X O _
3  X _ _ _
```

```
moved:

   0 1 2 3
-+--------
0  X _ O _
1  _ O O _
2  O X X _
3  O X _ _


Player 2's turn
Please input row and col: 1 3

   0 1 2 3
-+--------
0  X _ O _
1  _ O O X
2  O X X _
3  O X _ _

moved:

   0 1 2 3
-+--------
0  _ O _ X
1  X O X _
2  _ O X _
3  O O _ _


Player 1 won!!!
```

## Technical Requirements

To complete this assignment, you should utilize the following C++ programming concepts and data structures:

### Programming Language

- The game should be implemented in C++ using object-oriented programming principles.

### Data Structures

1. Arrays:
   - Use a 2D array to represent the 4x4 gameboard.

- Each element in the array should store the state of a position on the gameboard (empty, white marble, or black marble).

2. Classes:
    - Define appropriate classes.
3. Linked Lists:
    - Implement Linked List to model the tracks (not using existing libraries).
    - You may use more data structures to help mapping tracks to the 2D array.

## Input/Output

- The game should have a command-line interface for user interaction.
- Use `cin` and `cout` for input and output operations, respectively.
- Display the gameboard after each turn, showing the positions of marbles.
- Prompt players to enter their moves (row and column) for placing marbles on the gameboard.
- Validate user input to ensure it is within the valid range and corresponds to an empty position on the gameboard.

## Error Handling

- Implement error handling mechanisms to gracefully handle invalid user inputs, such as values out-of-bound or selecting occupied positions.
- Display appropriate error messages to guide players when invalid moves are attempted.

## Code Organization

- Use header files (`.h`) to declare classes, functions, and constants.
- Implement the corresponding source files (`.cpp`) to define the functionality of classes and functions.
- Use meaningful variable and function names that follow coding conventions and enhance code readability.
- Include comments to explain complex or non-obvious code segments.

## Compilation

- The code should be compiled using a Microsoft Visual C++ compiler
- Ensure that the code compiles without any errors.

## Documentation

Documentation includes evidence of testing (10%) and explanation on how data structure is used (10%)

## Mark Distribution

| | |
|---|---|
| Game board | 15 marks |
| Track | 20 marks |
| Marble movements | 20 marks |
| Game Play | 15 marks |
| UI/UX | 10 marks |
| Documentation: | 20 marks |

## Instructions to students

1. This is an End of Module Assessment and the weighting of this assignment is 50% of the Module Mark.

2. This assignment should be done by each individual student. Plagiarism will be treated seriously. All assignments that have been found involved wholly or partly in plagiarism (no matter these assignments are from the original authors or from the plagiarists) will score Zero mark.

3. You must use C++ to develop the programs.

4. You are required to hand in

   4.1 Explanation on how data structure is used in your program in MS Word

   4.2 Source code of all classes which should be well-commented.

   4.3 The evidence of testing. Prepare a word document with several test cases showing different inputs for different situations that your program may encounter and how your program responses to show the capability of your program. For each test case, states the objective of the test case, input data and expected result. You should also include screen dump for each test run as evidence.

5. Submit all your works (in a zip file) to the Moodle website (http://moodle.vtc.edu.hk) by 11:55 pm, 14 July 2024 (Sunday). Late submission may score ZERO mark.

6. Folders in the zip file

   Program

      Main.cpp

      LinkedList.h

      ...

   Documents

      Data_Structure_Explanation.doc