ENGG1110 Problem Solving by Programming (2021-2022 Term 1) Project

Deadline: 3-Dec-2021 (Fri) 23:59

---

1. Introduction

Connect 4 is a game played by two players in a game board with 7 columns and 6 rows.

Image source: https://commons.wikimedia.org/wiki/File:Connect_Four.jpg [By: Popperipopp]

Each player takes turn to choose a column to drop his/her mark with the following rules:

- If there is no mark in the chosen column, the new mark drops to the bottom row.
- If there are one to five marks in the chosen column, the new mark drops above the top mark.
- A full column with six marks cannot be chosen.

The first player who can have four (or more) of his/her marks in a horizontal, a vertical, or a diagonal row on the board is the winner. If no one wins after all 7 columns are full, it is a draw game.

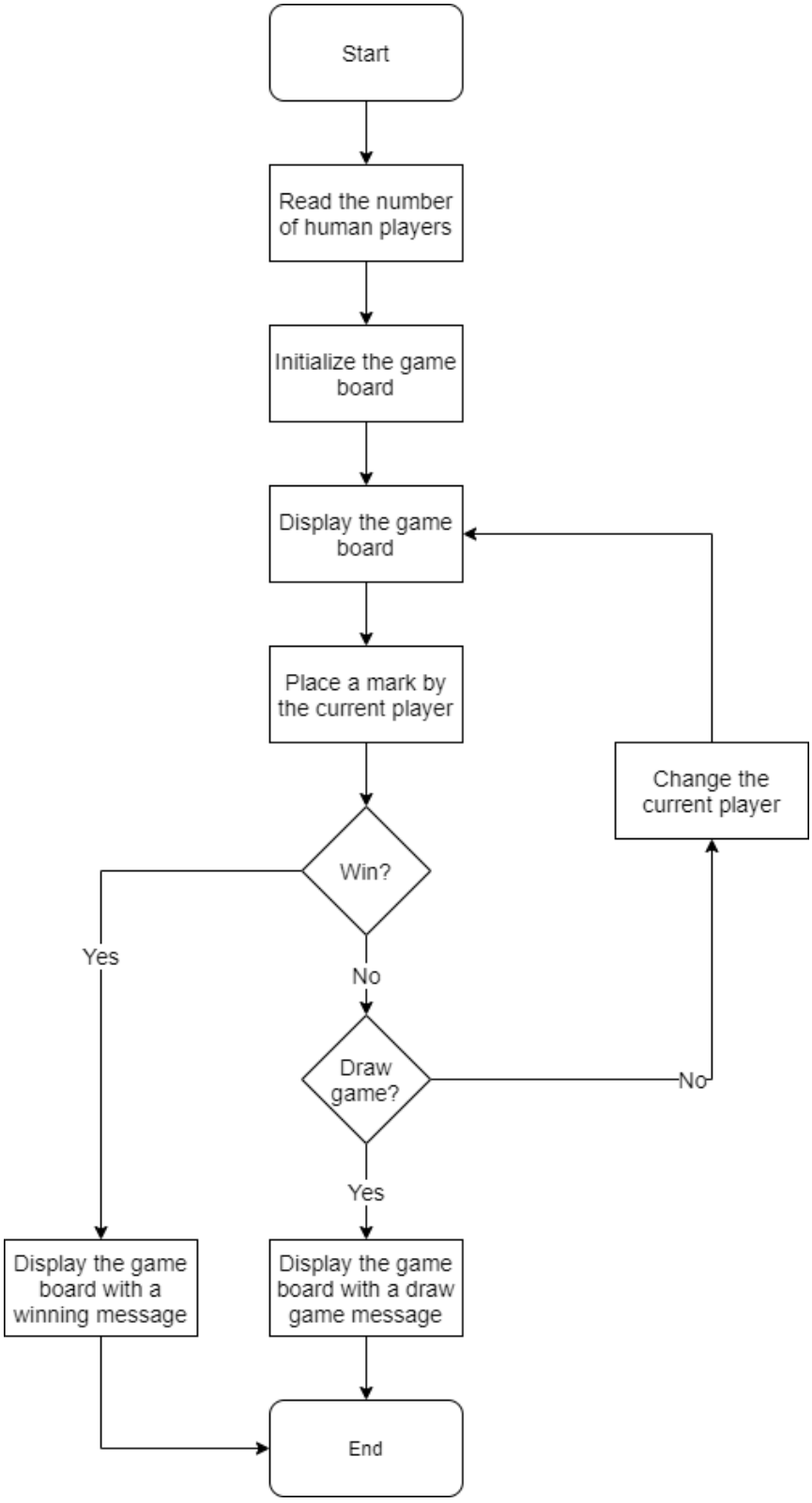
An animation for illustration can be found in: https://en.wikipedia.org/wiki/File:Connect_Four.gif

In the project, your task is to implement the game of Connect 4 for one or two human players in C language. You are required to complete the given **project.c** without modifying any existing code (except uncommenting the given source code for testing) nor any function signature. Marks will be deducted for each modification.

Please rename the filename to **project_<your student ID>.c** (E.g., project_1155123456.c).

2. Program Flow

The program flow is shown as follows:



3. Schedule

The suggested schedule is shown as follows:

Week	Tasks

9	• Complete the following functions:
	  ○ initGameBoard()
	  ○ printGameBoard()
	  ○ placeMarkByHumanPlayer()
	• In main():
	  ○ Test the above functions by uncommenting the given source code
	• No submission is needed at this stage

10-11	• Complete the following functions:
	  ○ hasWinner()
	  ○ isFull()
	• In main():
	  ○ Complete and test 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

12-13	• Complete the following function:
	  ○ placeMarkByComputerPlayer()
	• In main():
	  ○ Complete and test the implementation of the game for a human player and a computer player
	• Design your own test cases and perform the final testing of the project
	• Submit your code by the deadline

4. Detailed Program Design

4.1 Header Files and Variable Declarations

No header files other than stdio.h are allowed. In addition, no global variables (those declared outside functions) are allowed. In other words, all variables must be declared inside functions. Marks will be deducted for each violation.

4.2 Macros

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

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

Suppose we have the following in the other part of the source code:

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

The 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.

4.3 Read Number of Human Players

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

```
Enter the number of human players [1-2]:
```

This is handled in the main() function. You can assume that the user must be valid (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 moves first and places the CIRCLE mark; while Player 2 (or the computer player) places the CROSS mark.

4.4 Initialize Game Board

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

4.5 Print Game Board

The main() function invokes the printGameBoard() function, which displays the game board on the screen. The following shows some examples.

All squares are empty in the beginning of every game	||||||||| ||||||||| ||||||||| ||||||||| ||||||||| ||||||||| 1 2 3 4 5 6 7
After 10 moves in one game	||||||||| ||||||||| ||||||||| |||0|0||| |||X|X|0|| ||X|0|X|X|0| 1 2 3 4 5 6 7
After 42 moves in another game	|X|X|X|0|X|X|X| |0|0|0|X|0|0|0| |X|X|X|0|X|X|X| |0|0|0|X|0|0|0| |X|X|X|0|X|X|X| |0|0|0|X|0|0|0| 1 2 3 4 5 6 7

In the project, you are required to follow exactly the above output format. Using other output format will result in mark deduction.

4.6 Place Mark by Human Player

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

```
Player 1's turn:
```

```
Player 2's turn:
```

and then invokes the placeMarkByHumanPlayer() function, which asks the human player to place the mark. You can assume that the user input must be an integer. However, if the input is not between 1 to 7 inclusively, the program will print the following message and the user will need to input again:

```
Input out of range. Please input again:
```

On the other hand, if the input column is full, the program will print the following message and the user will need to input again:

```
Column is full. Please input again:
```

4.7 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:

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

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

```
Computer wins!
```

After that, the program terminates.

4.8 Check Draw Game

The main() function invokes the isFull() function, which returns 1 if the game board is full; otherwise it returns 0. If it is a draw game, the following messages is printed in the main() function:

```
Draw game.
```

After that, the program terminates.

4.9 Place Mark by Computer Player

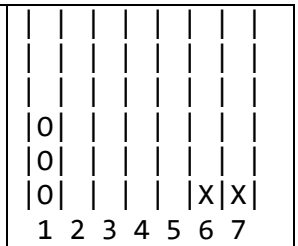
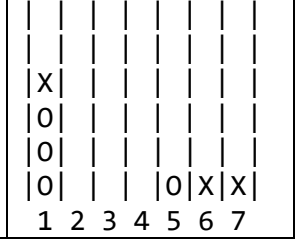
The program prints the following message in the main() function:

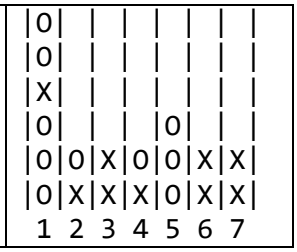
```
Computer's turn:
```

and then invokes the placeMarkByComputerPlayer() function, which determines the next move of the computer player. In this project, you are required to implement the following strategy (using other strategies will result in mark deduction):

- First, search for the winning move of the computer player (a move that let the computer win the game immediately)
  - If a winning move is found, the next move is the winning move
  - If there are multiple winning moves, the one with a smaller column number is chosen
  - [Hint: Try to put the mark in each column and use the hasWinner() function to check if it is a winning move; if it is not a winning move, remove the mark and try the next column]
- If there is no winning move for the computer player, then search for the human player's winning move (a move that let the human player win the game immediately)
  - If a winning move is found, the next move is the winning move
    - That is, place the mark in there to block the human player from winning
  - If there are multiple winning moves, the one with a smaller column number is chosen
    - That is, the human player can win afterwards and there is no way to block it
  - [Hint: Try to put the human player's mark in each column and use the hasWinner() function to check if it is a winning move; if it is a winning move, replace it by the computer player's mark]
- If there are no winning moves for both players, the next move is the column with the fewest marks
  - In case of tie, the one with a larger column number is chosen
  - [Hint: Scan the bottom row to see if there is an empty square; if not, scan the second bottom row, and so on]

Here are some examples:

The next move of the computer player (X) is column 1 to block the winning moving of the human player (O).	
There are no winning moves for both players. The next move of the computer player (X) is column 4 (the largest column number having the fewest marks).	

The next move of the computer player (X) is column 2, which is the winning move.	

5. Sample Run

Some sample runs are provided on Blackboard. In addition to them, you are strongly recommended to test your program by designing your own test cases.

6. 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/.

You are required to place the following declaration statement as the comment in the beginning of your .c source code and fill in your information.

```
/**
 * 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>
 */
```

7. Grading Platform

We will grade your work in **Code::Blocks** under **Windows 10**.

8. Submission

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

1. Login Blackboard
2. Go to 2021R1 Problem Solving By Programming (ENGG1110X), where X is your section
3. Go to Project → Project Submission
4. Upload **project_<your student ID>.c** (E.g., project_1155123456.c)
    - Note: Do not submit other files used in your IDE

Resubmissions are allowed. But only the latest one will be graded. 30% of the marks will be deducted for late submissions within one week. Late submissions more than one week will not be graded.