C++프로그래밍 및 실습

213996 한승한

1. 서론

목표 : Tic Tac Toe (2인용), Tic Tac Toe3 (3인용)

2. 요구 사항

1. 누구의 차례인지 출력
2. 좌표 입력 받기
3. 입력받은 좌표 유효성 체크
4. 좌표에 O / X 놓기
5. 현재 보드판 출력
6. 빙고 시 승자 출력 후 종료
7. 모든 칸이 차면 종료

3. 설계 및 구현

1-1. 사용자 요구사항 : 두명의 사용자가 번갈아가며 O와 X를 놓기

```
//차례 출력
    while(true) {
        switch(k%2) {
            case 0:
                cout << "첫번째 유저(X)의 차례입니다 -> ";
                currentUser = 'X';
                break;
            case 1:
                cout << "두번째 유저(O)의 차례입니다 -> ";
                currentUser = 'O';
                break;
        }
```

1-2. 기능 요구사항

<입력>

1. switch(k%2) //(k=0)
2. currentUser (사용자 순서에 따라 형태를 변환)
3. k++

<결과>

첫 번째 사용자는 'X' 형태의 모양이 출력

두 번째 사용자는 'O' 형태의 모양이 출력

<설명>

1. k (초기값 = 0)의 값을 받아 2로 나누어 나머지 값을 구한다.
2. 이때 나머지의 값이 0이면 currentUser의 값은 'X'로 나머지 값이 1이면 currentUser의 값은 'O'로 변환된다.
3. 마무리로 k의 값을 올려 나머지의 값이 순환되도록 한다.

2-1. 좌표 유효성 체크

```
//좌표 유효성 체크
        cout << "(x,y) 좌표를 입력하세요: ";
        cin >> x >> y;

        if (x >= numCell || y >= numCell) {
            cout << x << ", " << y << ": ";
            cout << "x와 y 둘 중 하나가 칸을 벗어납니다." << endl;
            continue;
        }
        if (board[x][y] != ' ') {
            cout << x << ", " << y << ": 이미 돌이 차있습니다." << endl;
            continue;
        }
```

2-2. 기능

<입력>

x, y (사용자가 입력하는 x좌표, y좌표)

numCell (=3)

<결과>

만일 사용자가 numCell의 값 이상을 입력하면 경고문과 함께 처음으로 돌아가 좌표를 입력하는 곳으로 간다.

<설명>

결과에 설명과 함께하였다.

3-1. 입력받은 좌표에 돌 넣기

```
//입력받은 좌표에 돌 넣기
        board[x][y] = currentUser;
```

3-2. 기능

<입력>

board[][] (2차원 배열 board)

x, y (사용자가 지정할 좌표)

currentUser (순서에 따라 바뀌는 모형)

<결과>

좌표에 돌맹이를 넣음.

<설명>

board라는 2차원 배열 [x]와 [y]를 사용자를 통해 입력받는다. 입력받은 곳에 currentUser의 모형을 집어 넣는다.

4-1. 보드판 출력

```
//보드판 출력
        for(int i = 0; i < numCell; i++) {
            cout << "---|---|---" << endl;
            for(int j = 0; j < numCell; j++) {
                cout << " " << board[i][j];
                if (j == numCell-1) {
                    break;
                }
                cout << " |";
            }
            cout << endl;
        }
        cout << "---|---|---" << endl;
```

```
"---|---|---"
board[0][0] | board[0][1] | board [0][2]
"---|---|---"
board[1][0] | board[1][1] | board [1][2]
"---|---|---"
board[2][0] | board[2][1] | board [2][2]
"---|---|---" //반복문 종료 후 추가
```

4-2. 기능

<입력>

i, j (반복 인자)

numCell (최대 반복 수)

board[i][j] (2차원 배열)

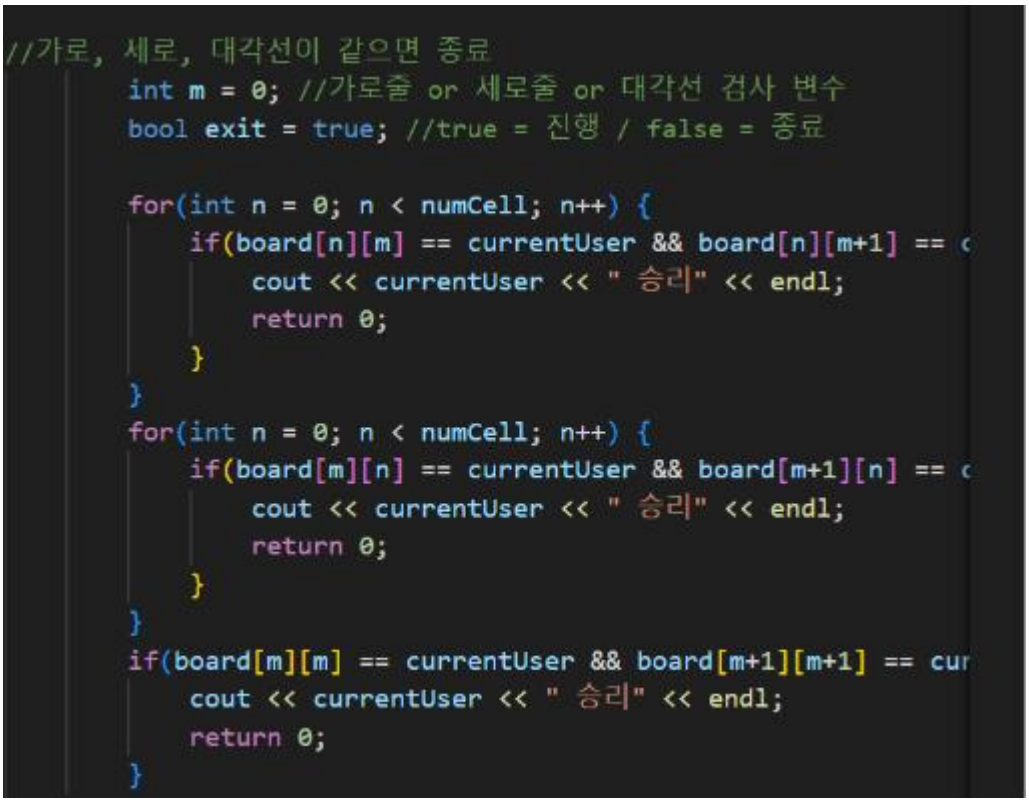
<결과>

보드판을 출력한다.

<설명>

처음 바둑판 첫 줄을 출력하고 ("---|---|---") 그 밑줄에 board[][]의 값을 순서대로 나열한다. 여기서 하나 나열할 때마다 " |" 공백과 칸막이를 추가한다. 그렇게 한 줄이 끝나면 다시 바둑판 줄을 출력하고 이 후 반복된다. 마지막엔 바둑판 한 줄을 더 추가한다. 코드 옆 설명을 그림으로 첨부했다.

5-1 종료 조건 (가로, 세로, 대각선)



```
//가로, 세로, 대각선이 같으면 종료
        int m = 0; //가로줄 or 세로줄 or 대각선 검사 변수
        bool exit = true; //true = 진행 / false = 종료

        for(int n = 0; n < numCell; n++) {
            if(board[n][m] == currentUser && board[n][m+1] == c
                cout << currentUser << " 승리" << endl;
                return 0;
            }
        }
        for(int n = 0; n < numCell; n++) {
            if(board[m][n] == currentUser && board[m+1][n] == c
                cout << currentUser << " 승리" << endl;
                return 0;
            }
        }
        if(board[m][m] == currentUser && board[m+1][m+1] == cur
            cout << currentUser << " 승리" << endl;
            return 0;
        }
```

5-2 기능

<입력>

m (반복문에 쓰이지 않고 독립적으로 초기화 된 변수)

n (반복문에 쓰일 인자)

currentUser (사용자에 따라 바뀌는 모형)

if (조건문)

return 0 (초기화)

<결과>

가로, 세로 및 대각에서 먼저 3개의 같은 모형을 맞추신 분에게 승리를 드립니다.

<설명>

if 조건문을 만족시킨다면 게임이 초기화(종료)된다. 승리자가 누구인지도 출력되니 안심해도 된다.

6-1 종료 조건 (자리 없음)

```
//모든 칸이 찼으면 종료
        for(int n = 0; n < numCell; n++) {
            for(int m = 0; m < numCell; m++) {
                if(board[n][m] == ' ') {
                    exit = false;
                    break;
                }
            }
            if (!exit) break;
        }

        if(exit) {
            cout << "칸이 부족하여 종료";
            return 0;
        }

        k++;
    }
    return 0;
}
```

6-2 기능

<입력>

n, m (반복 인자)

exit (bool의 기능으로 'false'면 게임이 계속되고, 'true'면 게임이 종료된다.)

k++ (초기에 설정한 0의 값이다. 이것이 가장 마지막에 하나씩 증가되며 사용자의 순서를 바꿔준다.)

<결과>

빈 좌석에 돌맹이가 하나씩 자리 잡아 더 이상의 자리가 없을 때 게임이 종료된다.

<설명>

if문을 통해 지속적으로 빈자리가 있나 호시탐탐 노린다. 빈자리가 있을 때는 지속적으로 'exit = false'를 유지하여 게임을 지속하다 만일 빈자리가 없어진다면 그 즉시 'exit = true'의 값으로 반복문을 통과하게 되고 제일 하단 if문에 잡혀가 시스템을 종료하게 된다.

3. TEST

1. 기능 별 테스트 결과

1. 누구의 차례인지 출력 + **2.** 좌표 입력 받기

```
첫번째 유저(X)의 차례입니다 -> (x,y) 좌표를 입력하세요: 0
0
```

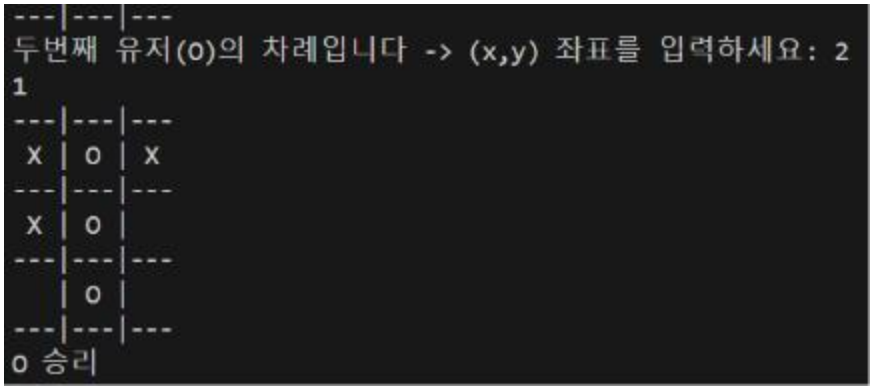
3. 입력 받은 좌표 유효성 체크

```
첫번째 유저(X)의 차례입니다 -> (x,y) 좌표를 입력하세요: 4
4
4, 4: x와 y 둘 중 하나가 칸을 벗어납니다.
첫번째 유저(X)의 차례입니다 -> (x,y) 좌표를 입력하세요:
```

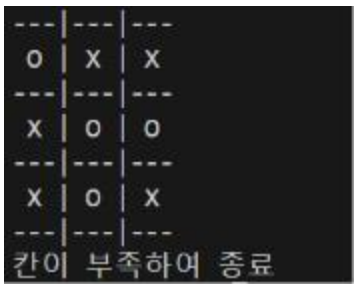
4. 좌표에 O/ X 놓기 + **5.** 현재 보드판 출력

```
첫번째 유저(X)의 차례입니다 -> (x,y) 좌표를 입력하세요: 0
0
---|---|---
 X |   |
---|---|---
   |   |
---|---|---
   |   |
---|---|---
두번째 유저(O)의 차례입니다 -> (x,y) 좌표를 입력하세요: 0
1
---|---|---
 X | O |
---|---|---
   |   |
---|---|---
   |   |
---|---|---
첫번째 유저(X)의 차례입니다 -> (x,y) 좌표를 입력하세요: 0
2
---|---|---
 X | O | X
---|---|---
   |   |
---|---|---
   |   |
---|---|---
```

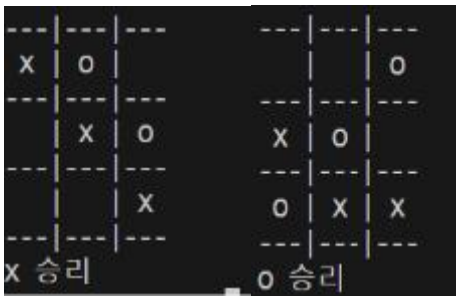
6. 빙고 시 승자 출력 후 종료



7. 모든 칸이 찼으면 종료

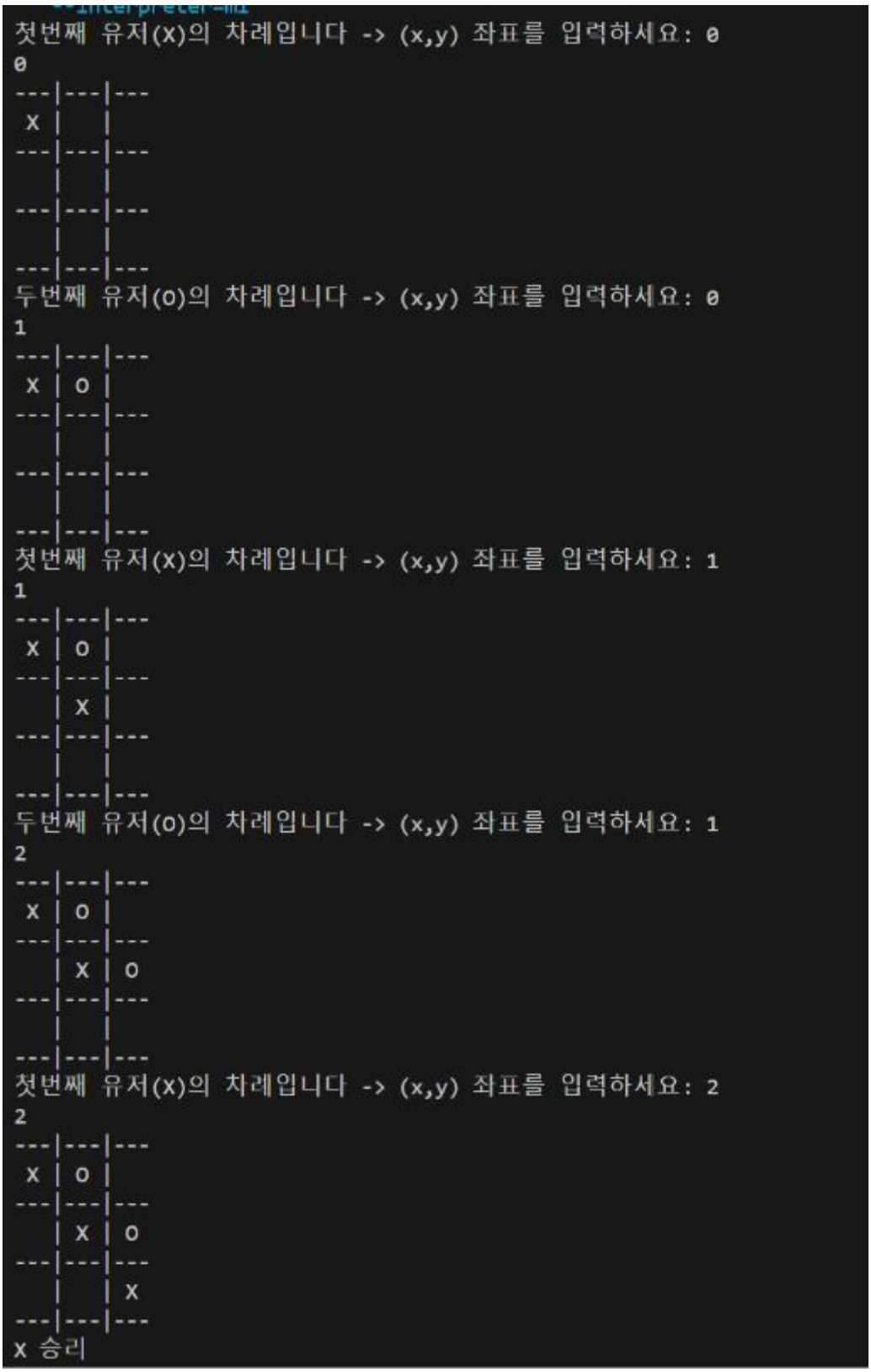


8. 대각으로 승리 시 종료



9. 한 줄 승리 시 종료 (세로, 가로)

2. 최종 테스트 스크린샷


첫번째 유지(X)의 차례입니다 -> (x,y) 좌표를 입력하세요: 0
0
X
두번째 유지(O)의 차례입니다 -> (x,y) 좌표를 입력하세요: 0
1
X | O
첫번째 유지(X)의 차례입니다 -> (x,y) 좌표를 입력하세요: 1
1
X | O
X
두번째 유지(O)의 차례입니다 -> (x,y) 좌표를 입력하세요: 1
2
X | O
X | O
첫번째 유지(X)의 차례입니다 -> (x,y) 좌표를 입력하세요: 2
2
X | O
X | O
X
X 승리

5. 결과 및 결론

1. 프로젝트 결과 : Tic Tac Toe 게임을 만들었음 / 3인용도 만들었음
2. 느낀 점: 이번이 처음이자 마지막이라고 생각하겠습니다.