프로젝트형 실습 #01

<Tic_Tac_Toe 게임>

234042 황선하

① 서론

- 목표 : 4주차까지 배운 내용(데이터 타입, 변수, 입출력 함수, 수식과 연산자, 제어문, 조건문, 반복문, 배열)을 활용하여 Tic_Tac_Toe 게임을 구현한다.

② 요구사항

- 사용자 요구사항 : 두 명의 사용자가 번 갈아가며 '0'와 'X'를 놓는다.

- 기능 요구사항 :
  1. 누구의 차례인지 출력한다.
  2. 좌표를 입력받는다.
  3. 입력 받은 좌표의 유효성을 체크한다. (이미 채워짐 or 벗어난 자리)
  4. 좌표에 0/X를 놓는다.
  5. 현재 보드판을 출력한다.
  6. 빙고 시 승자 출력 후 종료한다.
  7. 모든 칸이 찼으면 종료한다.

- 제약 조건 :
  1. 보드 판 제작 시 2차원 배열을 사용한다.

③ 설계 및 구현

1.
```
// 1. 누구 차례인지 출력
switch (k % 2) {
case 0:
    cout << "첫 번째 유저(X)의 차례입니다 -> ";
    currentUser = 'X';
    break;
case 1:
    cout << "두 번째 유저(0)의 차례입니다 -> ";
    currentUser = '0';
    break;
}
```

2. 입력
   - K = 0으로 초기화된 값
   - K % 2 = 1 or 0
   - currentUser = 현재 유저가 누구인지 나타내는 char값

3. 결과
   - 현재 유저가 'X'인지 '0'인지 출력

4. 설명
   - 뒤에서 continue를 통해 다시 반복문의 시작으로 돌아올 가능성이 있기 때문에 상황에 따라 currentUser가 유지되고 변경될 수 있도록 switch문 사용
   - k값이 변경되기 전까지는 currentUser가 유지, k값이 변경되면 currentUser 변화

1.
```
// 2. 좌표 입력 받기
cout << "(x, y) 좌표를 입력하세요 : ";
cin >> y >> x;      // x, y 좌표를 제대로 입력받기 위한 수정
```

2. 입력
   - x = 좌표 x값
   - y = 좌표 y값
3. 결과
   - x와 y좌표를 입력받는다.
4. 설명
   - 위치상 정확하도록 대입 순서를 반대로 한다.

1.
```
// 3. 입력받은 좌표의 유효성 체크
if (x >= numCell || y >= numCell){
    cout << x << ", " << y << ": ";
    cout << "x와 y 둘 중 하나가 칸은 벗어납니다." << endl;
    continue;
}
if (board[x][y] != ' '){
    cout << x << ", " << y << ": 이미 돌이 차있습니다." << endl;
    continue;
}
```

2. 입력
   - x = 좌표 x값
   - y = 좌표 y값
   - numCell = 가로 / 세로 의 칸 개수

3. 결과
   - 칸을 놓을 수 없는 이유 출력(유효하지 않는 칸, 이미 채워진 칸)
   - 출력 후 continue를 통해 while문의 시작으로 이동

4. 설명
   - 사용자가 입력한 좌표가 칸의 수(numCell)를 벗어나는지 확인(if활용)
   - 사용자가 입력한 좌표가 채워져 있는지 확인(if활용)

1.
```
// 4. 입력받은 좌표에 현재 유저의 돌 놓기
board[x][y] = currentUser;
```

2. 입력
   - board = 게임 판
   - currentUser = 현재 user('X' or '0')

3. 결과
   - board[x][y](올바른 좌표)에 currentUser가 대입된다.

4. 설명
   - 좌표가 올바르게 입력되었을 경우 현재의 사용자로 채워진다.

1.
```
// 5. 현재 보드 판 출력
for (int i = 0; i < numCell; i++){
    cout << "---|---|---" << endl;
    for (int j = 0; i < numCell; j++){
        cout << board[i][j];
        if(j == numCell - 1) {
            break;
        }
        cout << "  |";
    }
    cout << endl;
}
cout << "---|---|---" << endl;
k++;
```

2. 입력
   - i = 첫 번째 for문의 조건을 위한 변수
   - numCell = 가로 / 세로 의 칸 개수
   - j = 두 번째 for문의 조건을 위한 변수
   - board = 게임 판
   - k = currentUser를 결정하는 값(0 or 1)

3. 결과
   - board의 테두리 출력
   - 순서에 따라 두 명의 사용자가 놓은 currentUser 출력
   - K값 1증가

4. 설명
   - 이중 반복문을 통해 board의 테두리와 currentUser를 번 갈아가며 출력
   - if문을 통해 x축의 맨 오른쪽 currentUser를 입력했는지 확인하고 아닐 경우 |(board의 테두리)를 출력
   - 반복문이 모두 끝난 후 마지막 board의 테두리 출력
   - k값을 1더해주어 차례가 넘어갈 수 있도록

1.
```
// 6. 빙고 시 승자 출력 후 종료
bool win = false;

// 가로와 세로 판단
for (int i = 0; i < numCell; i++) {
    if (board[i][0] == currentUser && board[i][1] == currentUser && board[i][2] == currentUser ){
        win = true;
    }
    if (board[0][i] == currentUser && board[1][i] == currentUser && board[2][i] == currentUser ){
        win = true;
    }
}
// 대각선 판단
if (board[0][0] == currentUser && board[1][1] == currentUser && board[2][2] == currentUser) {
    win = true;
}
if (board[0][2] == currentUser && board[1][1] == currentUser && board[2][0] == currentUser) {
    win = true;
}

if (win == true) {
    cout << currentUser<< "의 승리입니다." << endl;
    break;
}
```

2. 입력
   - win = 승자의 여부를 나타내는 변수
   - i = 반복문을 위한 변수
   - numCell = 가로 / 세로 의 칸 개수
   - board = 게임 판
   - currentUser = 현재 user('X' or '0')

3. 결과
   - win이 false인 경우 해당 과정 패스
   - win이 true인 경우 currentUser를 출력하고 프로그램 종료

4. 설명
   - 빙고가 되는 경우 : 대각선, x(y)가 0(1)(2)일 때 y(x)가 0, 1, 2인 경우
   - 빙고 조건은 if문과 for문을 통해 구현
   - 빙고 조건을 만족했을 경우 win을 true로 변경하여 currentUser를 승리자로 출력하고 break문을 통해 while문을 빠져나간다.

1.
```
// 7. 모든 칸이 찼으면 종료
int full = 0;
for (int i = 0; i < numCell; i++) {
    for (int j = 0; j < numCell; j++) {
        if (board[i][j] != ' ')
            continue;
        else
            full++;
    }
}

if (full == 0) {
    cout << "모든 칸이 찼습니다." << endl;
    break;
}
```

2. 입력
   - full = 빈칸의 개수
   - i = 첫 번째 for문의 조건을 위한 변수
   - j = 두 번째 for문의 조건을 위한 변수
   - board = 게임 판

3. 결과
   - 모든 칸이 찼을 경우 이를 알리고 프로그램 종료

4. 설명
   - 반복문을 통해 board의 모든 칸을 확인하며 채워져 있으면 continue를 통해 while문의 시작으로 돌아간다.
   - 빈칸이 있는 경우 변수 full을 증가시킨다.
   - 최종 확인(if 활용) 결과 full이 0일 경우 break를 통해 while문을 나간다.

---

④ 테스트

1. 이미 채워진 자리에 체크하려는 경우

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력
---|---|---
X  |   |
---|---|---
   |   |
---|---|---
   |   |
---|---|---
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 0
0, 0: 이미 돌이 차있습니다.
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 :
```

2. board칸은 넘어간 자리에 체크하려는 경우

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 0
---|---|---
X  |   |
---|---|---
   |   |
---|---|---
   |   |
---|---|---
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 3 3
3, 3: x와 y 둘 중 하나가 칸은 벗어납니다.
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 :
```

3. 승부가 끝나지 않고 board의 모든 칸이 채워진 경우

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 2
---|---|---
X  |X  |0
---|---|---
0  |0  |X
---|---|---
X  |0  |X
---|---|---
모든 칸이 찼습니다.
PS C:\CPP2409>
```

4. 'X'가 가로선을 통해 승리한 경우

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 0
---|---|---
X  |X  |X
---|---|---
0  |0  |
---|---|---
   |   |
---|---|---
X의 승리입니다.
PS C:\CPP2409>
```

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 1
---|---|---
0  |0  |
---|---|---
X  |X  |X
---|---|---
   |   |
---|---|---
X의 승리입니다.
PS C:\CPP2409> 
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 2
---|---|---
   |   |
---|---|---
0  |0  |
---|---|---
X  |X  |X
---|---|---
X의 승리입니다.
PS C:\CPP2409> 
```

5. 'X'가 세로선을 통해 승리한 경우

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 2
---|---|---
X  |0  |
---|---|---
X  |0  |
---|---|---
X  |   |
---|---|---
X의 승리입니다.
PS C:\CPP2409> 
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 1 2
---|---|---
   |X  |0
---|---|---
   |X  |0
---|---|---
   |X  |
---|---|---
X의 승리입니다.
PS C:\CPP2409> 
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 2
---|---|---
0  |   |X
---|---|---
0  |   |X
---|---|---
   |   |X
---|---|---
X의 승리입니다.
PS C:\CPP2409> 
```

6. 'X'가 대각선을 통해 승리한 경우

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 2
---|---|---
X  |   |
---|---|---
0  |X  |
---|---|---
0  |   |X
---|---|---
X의 승리입니다.
PS C:\CPP2409> 
```

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 2
---|---|---
0  |   |X
---|---|---
0  |X  |
---|---|---
X  |   |
---|---|---
X의 승리입니다.
PS C:\CPP2409> 
```

7. '0'가 가로선을 통해 승리한 경우

```
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 0
---|---|---
0  |0  |0
---|---|---
   |X  |
---|---|---
   |X  |X
---|---|---
0의 승리입니다.
PS C:\CPP2409> 
```

```
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 1
---|---|---
X  |X  |0
---|---|---
0  |0  |0
---|---|---
X  |   |X
---|---|---
0의 승리입니다.
PS C:\CPP2409> 
```

```
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 2
---|---|---
X  |   |
---|---|---
X  |X  |
---|---|---
0  |0  |0
---|---|---
0의 승리입니다.
PS C:\CPP2409> 
```

8. '0'가 세로선을 통해 승리한 경우

```
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 2
---|---|---
0  |   |X
---|---|---
0  |X  |
---|---|---
0  |   |X
---|---|---
0의 승리입니다.
PS C:\CPP2409>
```

```
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 1 2
---|---|---
X  |0  |
---|---|---
X  |0  |
---|---|---
   |0  |X
---|---|---
0의 승리입니다.
PS C:\CPP2409>
```

```
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 2
---|---|---
X  |   |0
---|---|---
   |X  |0
---|---|---
X  |   |0
---|---|---
0의 승리입니다.
PS C:\CPP2409>
```

9. '0'가 대각선을 통해 승리한 경우

```
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 2
---|---|---
0  |X  |X
---|---|---
   |0  |X
---|---|---
   |   |0
---|---|---
0의 승리입니다.
PS C:\CPP2409>
```

```
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 0
---|---|---
X  |   |0
---|---|---
   |0  |X
---|---|---
0  |   |X
---|---|---
0의 승리입니다.
PS C:\CPP2409>
```

10. 잘 못된 입력 값을 넣은 경우

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 3, 0: x와 y 둘 중 하나가 칸은 벗어납니다.
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 3, 0: x와 y 둘 중 하나가 칸은 벗어납니다.
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 3, 0: x와 y 둘 중 하나가 칸은 벗어납니다.
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 3, 0: x와 y 둘 중 하나가 칸은 벗어납니다.
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 3, 0: x와 y 둘 중 하나가 칸은 벗어납니다.
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 3, 0: x와 y 둘 중 하나가 칸은 벗어납니다.
PS C:\CPP2409>
```

(무한루프 발생)

11. 최종 테스트 결과

```
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 0
---|---|---
X  |   |
---|---|---
   |   |
---|---|---
   |   |
---|---|---
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 1 1
---|---|---
X  |   |
---|---|---
   |0  |
---|---|---
   |   |
---|---|---
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 1
---|---|---
X  |   |
---|---|---
X  |0  |
---|---|---
   |   |
---|---|---
두 번째 유저(0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 0
---|---|---
X  |   |0
---|---|---
X  |0  |
---|---|---
   |   |
---|---|---
첫 번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 2
---|---|---
X  |   |0
---|---|---
X  |0  |
---|---|---
X  |   |
---|---|---
X의 승리입니다.
PS C:\CPP2409>
```

⑤ 결과 및 결론

1. 프로젝트를 통해 Tic Tac Toe 게임을 만들어보았다. (배열, 조건문, 반복문, break, continue를 주로 사용)

2. 느낀 점 : 개념에 따른 단계별 문제를 스스로 생각하여 해결해 본 경험이 적어서 프로젝트를 진행하는 데에 어려움이 컸다. 각각의 개념이 어떻게 사용될 수 있는지 예시를 많이 접해본 후(연습해본 후) 프로젝트를 진행했다면 더욱 빠르게 답에 접근할 수 있었을 것 같다. 또한 이것저것 작성하여 실행해보고 보고서까지 작성하기에 2시간이 부족하게 느껴졌다. 시간이 충분했다면 여러 방면으로 코드를 작성해보고 가장 효율적인 방법을 찾아갈 수 있었을 것 같다.