Tic Tac Toe

과목: C++프로그래밍및실습

학과: 소프트웨어공학과

학번: 214270

이름: 현경서

목차

1. 서론

1) 프로젝트 및 배경

4주차까지 배운 내용(입출력, 조건문, 반복문, 배열)에 대한 실습

2) 목표

Tic Tac Toe 게임 구현

2. 요구사항

1) 사용자 요구사항

3 * 3 보드에 두명의 사용자가 번갈아 가면서 O와 X를 놓기

2) 기능 요구사항

① 누구의 차례인지 출력
② 좌표 입력 받기
③ 입력 받은 좌표 유효성 체크
④ 좌표에 O / X 놓기
⑤ 현재 보드 판 출력
⑥ 빙고 시 승자 출력 후 종료
⑦ 모든 칸이 찼으면 종료

3. 설계 및 구현

1) 기능 별 구현 사항

① 누구의 차례인지 출력

```
20          // 1. 누구 차례인지 출력
21          switch(k % 2){
22              case 0:
23                  cout << "첫번째 유저(X)의 차례입니다 -> ";
24                  currentUser = 'X';
25                  break;
26              case 1:
27                  cout << "두번째 유저(O)의 차례입니다 -> ";
28                  currentUser = 'O';
29                  break;
30          }
```

I. 입력
- k = 턴 수

II. 결과
- 현재 유저를 출력
- currentUser 변수에 현재 유저 저장

III. 설명
- k % 2가 0일 경우 현재 유저는 X이다.
- k % 2가 1일 경우 현재 유저는 O이다.

② 좌표 입력 받기

```
32          // 2. 좌표 입력 받기
33          cout << "(x, y) 좌표를 입력하세요: ";
34          cin >> x >> y;
```

I. 입력
- 변수 x, y에 대한 사용자 입력

II. 결과
- 사용자의 입력을 변수 x, y에 저장

III. 설명

③ 입력 받은 좌표 유효성 체크

```
36          // 3. 입력받은 좌표의 유효성 체크
37          if (x >= numCell || y >= numCell){
38              cout << x << ", " << y << ": ";
39              cout << "x와 y중 둘 중 하나가 칸을 벗어납니다." << endl;
40              continue;
41          }
```

I. 입력
- x = x 좌표 값
- y = y 좌표 값
- numCell = 가로 / 세로 칸의 개수

II. 결과
- 칸에 돌을 놓을 수 없는 이유를 출력
- whlile문 초반으로 이동

III. 설명
- 사용자가 입력한 좌표가 보드를 벗어나는지 if문으로 확인한다.

② -② 좌표 변환

```
43          // 2-2. 좌표 변환
44          int tmp = y;
45          y = x;
46          x = numCell - 1 - tmp;
```

I. 입력
- x = x 좌표 값
- y = y 좌표 값
- numCell = 가로 / 세로 칸의 개수

II. 결과
- x, y 값을 변환하여 저장한다.

III. 설명
- 입력 받은 x, y 값은 행렬 인덱스에 대응하는 값이므로 이를 좌표에 대응하는 값으로 변환한다.

③ -② 입력 받은 좌표의 유효성 체크

```
48          // 3-2. 입력받은 좌표의 유효성 체크
49          if (board[x][y] != ' '){
50              cout << x << ", " << y << ": 이미 돌이 차있습니다." << endl;
51              continue;
52          }
```

I. 입력
- board[x][y] = 칸의 내용

II. 결과
- 칸에 돌을 놓을 수 없는 이유를 출력
- whlile문 초반으로 이동

III. 설명
- 사용자가 입력한 좌표에 돌이 이미 있는지 if문으로 확인한다.

④ 좌표에 O / X 놓기

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

I. 입력
- currentUser = 현재 유저

II. 결과
- board[x][y]에 현재 유저 저장

III. 설명
- x, y 좌표에 해당하는 위치에 현재 유저의 돌을 놓는다.

⑤ 현재 보드 판 출력

```
55          // 5. 현재 보드 판 출력
56          for (int i = 0; i < numCell; i++){
57              cout << "---|---|---" << endl;
58              for (int j = 0; j < numCell; j++){
59                  cout << " " << board[i][j];
60                  if (j == numCell - 1){
61                      break;
62                  }
63                  cout << " |";
64              }
65              cout << endl;
66          }
67          cout << "---|---|---" << endl;
```

I. 입력
- numCell = 가로 / 세로 칸의 개수
- board[i][j] = 칸의 내용

II. 결과
- 보드 판 출력

III. 설명
- 중첩 for문과 2차원 배열 board에 있는 정보를 통해 보드판을 출력한다.

⑥ 빙고 시 승자 출력 후 종료

```
69          // 6. 빙고 시 승자 출력 후 종료
70          if (k >= 4){ // k < 4 이면 승자가 나올 수 없다
71              int i = 0, j = 0;
72              for (i = 0; i < numCell; i++){ // 가로
73                  for (j = 0; j < numCell - 1; j++){
74                      if (board[i][j] == ' ') break;
75                      if (board[i][j] == board[i][j+1]) continue;
76                      else break;
77                  }
78                  if (j == numCell - 1){
79                      cout << board[i][j] << "의 승리입니다" << endl;
80                      return 0;
81                  }
82              }
83
84              for (j = 0; j < numCell; j++){ // 세로
85                  for (i = 0; i < numCell - 1; i++){
86                      if (board[i][j] == ' ') break;
87                      if (board[i][j] == board[i+1][j]) continue;
88                      else break;
89                  }
90                  if (i == numCell - 1){
91                      cout << board[i][j] << "의 승리입니다" << endl;
92                      return 0;
93                  }
94              }
```

```
95
96          for (i = 0; i < numCell - 1; i++){ // 대각선
97              if (board[i][i] == ' ') break;
98              if (board[i][i] == board[i+1][i+1]) continue;
99              else break;
100         }
101         if (i == numCell - 1){
102             cout << board[i][i] << "의 승리입니다" << endl;
103             return 0;
104         }
105
106         for (i = 0; i < numCell - 1; i++){ // 대각선
107             if (board[(numCell-1)-i][i] == ' ') break;
108             if (board[(numCell-1)-i][i] == board[(numCell-1)-(i+1)][i+1]) continue;
109             else break;
110         }
111         if (i == numCell - 1){
112             cout << board[(numCell-1)-i][i] << "의 승리입니다" << endl;
113             return 0;
114         }
115     }
```

I. 입력
- k = 턴 수
- numCell = 가로 / 세로 칸의 개수
- board[i][j] = 칸의 내용

II. 결과
- 승자 출력
- 출력 후 종료

III. 설명
- 빙고가 나온 사용자가 있는지 중첩 for문과 if문을 이용해 확인한다.
- 빙고가 나오기 위해서는 한 사용자가 최소 3개의 돌을 놓아야 한다. 즉, 보드에 최소 5개의 돌이 놓아져 있어야 하며 4턴일 때 5개의 돌이 놓인다. 따라서 k >= 4 일때부터 빙고 확인을 하면 된다.

⑦ 모든 칸이 찼으면 종료

```
117     // 7. 모든 칸이 찼으면 종료
118     if (k == 8){
119         cout << "무승부입니다.";
120         break;
121     }
```

I. 입력
- k = 턴 수

II. 결과
- 무승부 출력

- 출력 후 종료

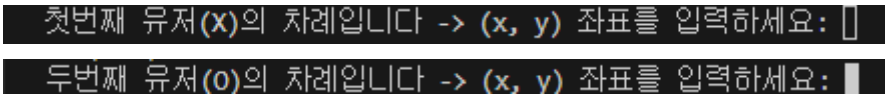
III. 설명

- 모든 칸이 찬 경우는 8턴일 때를 의미하므로 if문을 통해 k == 8일 경우 종료한다.

4. 테스트

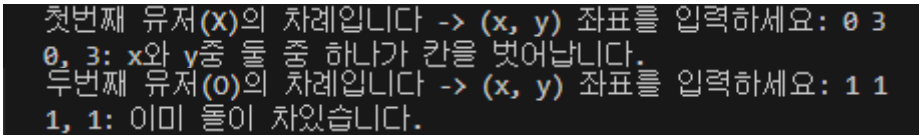
1) 기능 별 테스트 결과

① 누구의 차례인지 출력



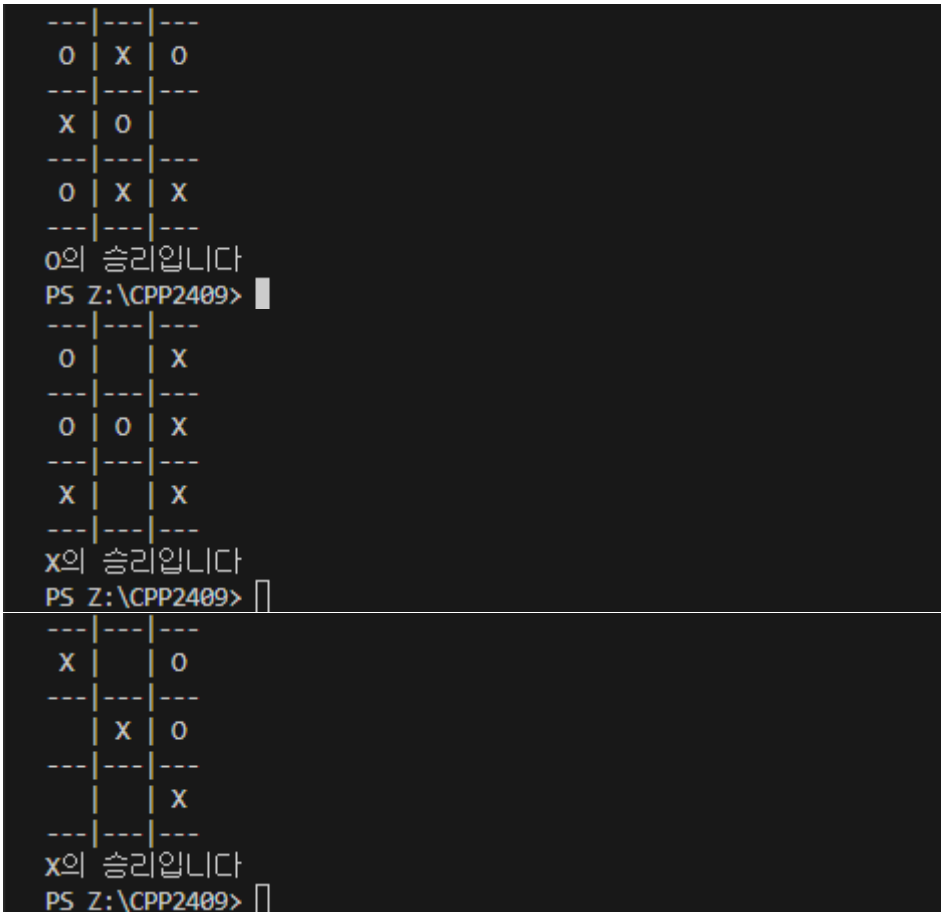
② 좌표 입력 받기

③ 입력 받은 좌표 유효성 체크



⑤ 현재 보드 판 출력

⑥ 빙고 시 승자 출력 후 종료



⑦ 모든 칸이 찼으면 종료

2) 최종 테스트 스크린샷

```
첫번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요: 0 0
---|---|---
   |   |
---|---|---
   |   |
---|---|---
 X |   |
---|---|---
두번째 유저(O)의 차례입니다 -> (x, y) 좌표를 입력하세요: 1 1
---|---|---
   |   |
---|---|---
   | O |
---|---|---
 X |   |
---|---|---
첫번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요: 1 0
---|---|---
   |   |
---|---|---
   | O |
---|---|---
 X | X |
---|---|---
두번째 유저(O)의 차례입니다 -> (x, y) 좌표를 입력하세요: 2 0
---|---|---
   |   |
---|---|---
   | O |
---|---|---
 X | X | O
---|---|---
첫번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요: 0 2
---|---|---
 X |   |
---|---|---
   | O |
---|---|---
 X | X | O
---|---|---
두번째 유저(O)의 차례입니다 -> (x, y) 좌표를 입력하세요: 0 2
0, 0: 이미 돌이 차있습니다.
두번째 유저(O)의 차례입니다 -> (x, y) 좌표를 입력하세요: 3 -1
3, -1: x와 y중 둘 중 하나가 칸을 벗어납니다.
두번째 유저(O)의 차례입니다 -> (x, y) 좌표를 입력하세요: 2 2
---|---|---
 X |   | O
---|---|---
   | O |
---|---|---
 X | X | O
---|---|---
첫번째 유저(X)의 차례입니다 -> (x, y) 좌표를 입력하세요: 0 1
---|---|---
 X |   | O
---|---|---
 X | O |
---|---|---
 X | X | O
---|---|---
X의 승리입니다
PS Z:\CPP2409>
```

5. 결과 및 결론

1) 프로젝트 결과

지금까지 배운 내용을 이용하여 Tic Tac Toe 게임을 만들었다.

2) 느낀점

시간 안에 문제를 해결하는 능력이 없다는 것을 알게 되었다. 많은 연습이 필요하다고 생각했다. 코드를 너무 급하게 한번에 짜지 말고 천천히 실행해가면서 작성하는게 오히려 빠를 수 있다는 것을 느꼈고 체계적으로 작성해 나가야 효율적인 코드가 나올 수 있겠구나 라는 생각을 했다.