Week5 Tic Tac Toe 프로젝트 보고서

소프트웨어공학과 214784 이상현

1. 서론

1. 프로젝트 목적 및 배경: 9월 한달 간 배운 C++ 개발 능력을 보기 위한 실습

2. 목표: Tic Tac Toe 2인용 게임 구현

2. 요구사항

1. 사용자 요구사항: 두 명이 번갈아가면서 O와 X를 놓기 (빙고를 먼저 달성하면 승리, 빙고를 달성하지 못하고 9칸을 다 채웠을 시 무승부 처리)

2. 기능 요구사항:

① 누구의 차례인지 출력

② (x, y) 좌표 입력 받기

③ 입력 받은 좌표 유효성 체크

④ 좌표에 O / X 놓기

⑤ 현재 보드판 출력

⑥ 빙고 시 승자 출력 후 종료

⑦ 모든 칸이 찼으면 무승부를 문구 출력 후 종료

3. 제약 조건: 2차원 배열의 보드판 사용

3. 설계 및 구현

1. 기능 별 구현 사항

① 누구의 차례인지 출력

1. 코드 블록 스크린샷

```
while (true) {
    // 1. 누구 차례인지 출력
    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;
    }
```

2. 입력

   k : 입력할 차례가 누구인지 판별하기 위한 값

   currentUser : 말을 둘 차례인 유저 O와 X

3. 결과
   K의 값에 따른 유저 차례 출력.
4. 설명
   k가 짝수면 X의 차례, k가 홀수면 O의 차례
   switch문의 조건에 k가 짝수인지 홀수인지 판단하여 누구의 차례인지 출력

② (x, y) 좌표 입력 받기

1. 코드 블록 스크린샷

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

2. 결과
   x, y 좌표를 입력하라는 문구 출력
   사용자에게 입력 받은 수 x, y 변수에 할당
3. 설명

   cout을 통해 좌표를 입력하라는 문구를 출력 한 후 입력 받은 두 수를 cin을 통하여 x와 y 변수의 값으로 할당함.

③ 입력 받은 좌표 유효성 체크

1. 코드 블록 스크린샷

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

2. 입력

   x = 좌표 x값

   y = 좌표 y값

   numCell = 가로 / 세로 칸의 개수

3. 결과
   칸을 놓을 수 없는 이유 출력
   출력 후 while문 초반으로 이동
4. 설명
   첫번째 if문에서 사용자가 입력한 좌표가 numCell의 크기를 벗어나는지 판단
   두번째 if문엣더 사용자가 입력한 좌표에 돌이 이미 있는지 판단

④ 좌표에 O / X 놓기

1. 코드 블록 스크린샷

```
// 4. 입력받은 좌표에 현재 유저의 돌 놓기
// x와 y 좌표를 반대로 생각하기 위해서 board[y][x]에 값 할당
board[y][x] = currentUser;
```

2. 입력

   x = 좌표 x값

   y = 좌표 y값

   currentUser : 말을 둘 차례인 유저 O와 X

3. 결과
   입력받은 x와 y좌표의 보드판에 값 할당
4. 설명

x좌표와 y좌표를 직관적으로 보기 위해 board[y][x] 형식으로 말을 놓도록 함

⑤ 현재 보드판 출력

1. 코드 블록 스크린샷

```
// 5. 현재 보드 판 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;
```

2. 입력

   i = x좌표를 표현할 for문 지역 변수

   j = y좌표를 표현할 for문 지역 변수

   numCell = 가로 / 세로 칸의 개수

3. 결과
   사용자가 놓은 말과 보드판 형식 출력
4. 설명
   for문을 두개 사용하여 board[i][j]에 놓여진 말을 출력하고
   보드판 그림을 표현하기 위한 ---|---|---를 첫번째 for문의 시작에 출력, 종료에 출력하였다.

⑥ 빙고 시 승자 출력 후 종료

1. 코드 블록 스크린샷

```cpp
// 6. 빙고 시 승자 출력 후 종료
// 6-1 가로 빙고 확인
for (int i = 0; i < numCell; i++) {
    int count = 0;
    for (int j = 0; j < numCell; j++) {
        if (board[i][j] == currentUser) {
            count++;
        }
    }
    if (count == 3) {
        cout << currentUser << "의 가로 빙고 달성!" << endl;
        return 0;
    }
}
// 6-2 세로 빙고 확인
 for (int j = 0; j < numCell; j++) {
    int count = 0;
    for (int i = 0; i < numCell; i++) {
        if (board[i][j] == currentUser) {
            count++;
        }
    }
    if (count == 3) {
        cout << currentUser << "의 세로 빙고 달성!" << endl;
        return 0;
    }
}
// 6-3 대각선 빙고 확인
if(board[0][0] == currentUser && board[1][1] == currentUser & board[2][2] == currentUser) {
    cout << currentUser << "의 대각선 빙고 달성!" << endl;
    break;
}
else if (board[2][0] == currentUser && board[1][1] == currentUser & board[0][2] == currentUser) {
    cout << currentUser << "의  대각선 빙고 달성!" << endl;
    break;
}
```

2. 입력

   i = x좌표를 표현할 for문 지역 변수

   j = y좌표를 표현할 for문 지역 변수

   numCell = 가로 / 세로 칸의 개수

   count = 빙고 확인을 위한 돌의 개수

3. 결과
   가로, 세로, 대각선 빙고를 판단하여 빙고에 성공했음을 출력
4. 설명

   가로 빙고와 세로 빙고 판별은 이중 for문을 사용하여 한 줄이나 한 칸에 currentUser 값과 같은 값들이 3개 있는지 판단하여 존재한다면 빙고를 출력함.
   대각선은 board[0][0], board[1][1], board[2][2] 값이 currentUser 값과 같거나

   board[2][0], board[1][1], board[0][2] 값이 currentUser 값과 같다면 대각선 빙고를 출력

⑦ 모든 칸이 찼으면 무승부 문구 출력 후 종료

1. 코드 블록 스크린샷

```
// 7. 모든 칸이 찼으면 종료
if (k == 8) {
    cout << "모든 칸이 찼으므로 무승부 처리" << endl;
    break;
}
k++;
```

2. 입력

   k : 입력할 차례가 누구인지 판별하기 위한 값이자 실행 횟수를 나타내는 값

3. 결과

   k가 8이라면 무승부 처리를 알리는 출력 후, while문 종료

4. 설명
   실행 횟수를 나타내는 값인 k가 8이면 말을 놓는 횟수가 9번째라는 의미이므로 무승부인채 보드판이 꽉 참.

4. 테스트

- (입력에 따라 원하는 결과나 나오는지 확인하는 과정)

1. 기능 별 테스트 결과: (요구사항 별 스크린샷)
   1. 누구 차례인지 출력

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

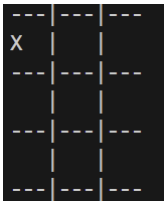
   2. (x, y)좌표 입력 받기

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

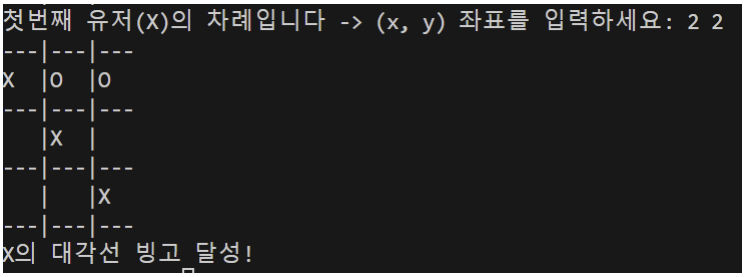
   3. 입력 받은 좌표 유효성 체크

```
두번째 유저(O)의 차례입니다 -> (x, y) 좌표를 입력하세요: 3 1
3, 1: x와 y 둘 중 하나가 칸을 벗어납니다.
두번째 유저(O)의 차례입니다 -> (x, y) 좌표를 입력하세요: 0 0
0, 0: 이미 돌이 차있습니다.
```

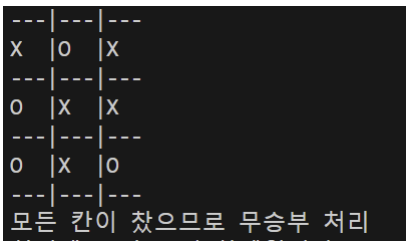
4. 좌표에 O / X 놓기
5. 현재 보드판 출력



6. 빙고 시 승자 출력 후 종료 (x의 대각선 빙고 시나리오)



7. 모든 칸이 찼으면 무승부 문구 출력 후 종료



2. 최종 테스트 스크린샷: (프로그램 전체 동작 스크린샷)