

C++ 프로그래밍 및 실습 Tic Tac Toe 게임 보고서

학번: 213652

학과: 소프트웨어공학과

이름: 김지환

1. 서론

1. 프로젝트 목적 및 배경: 4주차까지 배운 내용에 대한 실습을 위해 진행
2. 목표: Tic Tac Toe 게임 구현

2. 요구사항

1. 사용자 요구사항: 두 명의 사용자가 번갈아가며 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 보드판 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

1. 코드블록 스크린샷

```
//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 = 누구 차례인지 체크하기 위한 변수

3. 결과

k 나누기 2의 나머지가 0일 경우 x의 차례라고 출력

k 나누기 2의 나머지가 1일 경우 o의 차례라고 출력

출력 후 switch문을 나가고 다음 블록 실행

4. 설명

1. 두 경우를 나누기 위해 switch문 활용

2. 좌표 입력 받기

1. 코드블록 스크린샷

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

2. 입력

x = 좌표 x 값

y = 좌표 y 값

3. 결과

좌표를 입력하라는 문구를 출력

사용자의 입력 내용을 x, y에 대입

4. 설명

1. cout을 통해서 사용자에게 입력을 요구하는 문구 출력

2. cin을 통해서 사용자의 입력 내용을 받아들임

3. 입력받은 좌표의 유효성 체크

1. 코드블록 스크린샷

```
//3. 입력받은 좌표의 유효성 체크
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. 입력

x = 좌표 x 값

y = 좌표 y 값

numCell = 가로 / 세로 칸 개수

3. 결과

칸을 놓을 수 없는 이유를 출력

출력 후 while문 초반으로 이동

4. 설명

1. 사용자가 입력한 좌표가 게임 판을 벗어나는지 if로 체크

2. 사용자가 입력한 좌표에 돌이 이미 있는지 if로 체크

4. 좌표에 O / X 놓기

1. 코드블록 스크린샷

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

2. 입력

board[x][y] = 사용자가 입력한 보드판의 좌표

currentUser = 현재 유저의 돌

3. 결과

사용자가 입력한 좌표에 현재 유저의 돌이 놓임

4. 설명

1. 앞에서 생성했던 보드판 2차원배열에 'X'나 'O'가 대입
이 된다.

5. 현재 보드판 출력

1. 코드블록 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

numCell = 가로 / 세로 칸 개수

board[i][j] = 보드판에 있는 O / X / ''

3. 결과

보드판이 생성된다.

보드판에 돌이 있을 경우 O / X가 출력이 된다

보드판에 돌이 없을 경우 공백' '이 출력이 된다.

4. 설명

1. 이중 for문을 활용한다.

2 첫번째 for에서 보드판의 위쪽 경계를 생성하고 다음줄
로 넘긴다.

3. 두번째 for에서 board[i][j]에 있는 값을 출력한다.

4. 두번째 for가 끝나기 전에 | 를 이용하여 보드판의
경계를 생성해준다.

5. 그러나 끝 부분엔 경계가 없어야 하므로 if문을 활용
하여 j가 numCell - 1 즉, 2일 경우 두번째 for문을
벗어나게 하여 경계가 생성되지 않도록 한다.

6. 이중 for문이 끝난 뒤에 아래쪽 경계를 생성해야
하기 때문에 직접 경계를 출력해준다.

6.1 빙고(대각선) 시 승자 출력 후 종료

1. 코드블록 스크린샷

```
//6.1 빙고(대각선) 시 승자 출력 후 종료
if((board[0][0] != ' ') && (board[0][0] == board[1][1]) && (board[0][0] == board[2][2])){
    cout << "왼쪽 위에서 오른쪽 아래 대각선으로 모두 돌이 놓였습니다!: ";
    if((k%2) == 0){
        cout << "첫번째 유저(x)의 승리입니다!" << endl;
        cout << "게임을 종료합니다." << endl;
        return 0;
    }else{
        cout << "두번째 유저(o)의 승리입니다!" << endl;
        cout << "게임을 종료합니다." << endl;
        return 0;
    }
}

if((board[0][2] != ' ') &&(board[0][2] == board[1][1]) && (board[0][2] == board[2][0])){
    cout << "오른쪽 위에서 왼쪽 아래 대각선으로 모두 돌이 놓였습니다!: ";
    if((k%2) == 0){
        cout << "첫번째 유저(x)의 승리입니다!" << endl;
        cout << "게임을 종료합니다." << endl;
        return 0;
    }else{
        cout << "두번째 유저(o)의 승리입니다!" << endl;
        cout << "게임을 종료합니다." << endl;
        return 0;
    }
}
```

2. 입력

board[0][0] = 왼쪽 위 끝

board[1][1] = 가운데

board[2][2] = 오른쪽 아래 끝

board[0][2] = 오른쪽 위 끝

board[1][1] = 가운데

board[2][0] – 왼쪽 아래 끝

3. 결과

왼쪽 위에서 오른쪽 아래 대각선으로 모두 돌이 놓일 경우
우 멘트를 출력하며 k 나누기 2의 나머지가 0일 경우 x
의 승리 멘트가 나오며 아닐 경우 o의 승리 멘트가 나온다.

오른쪽 위에서 왼쪽 아래 대각선으로 모두 돌이 놓일 경우
우 멘트를 출력하며 k 나누기 2의 나머지가 0일 경우 x
의 승리 멘트가 나오며 아닐 경우 o의 승리 멘트가 나온다.

4. 설명

1. &&와 if를 활용하여 맨 왼쪽 윗부분과 가운데 그리고
맨 오른쪽 부분이 같을 경우 대각선이 만들어진
것이기 때문에 x나 y의 승리임을 알 수 있다.

2. &&와 if를 활용하여 오른쪽 윗부분과 가운데 그리고
맨 왼쪽 아랫부분이 같을 경우 대각선이 만들어진 것
이기 때문에 x나 y의 승리임을 알 수 있다.

3. 만약에 누군가가 승리할 경우 누구의 차례인지를 알면

알 수 있기 때문에 이와 if문을 활용하여 $k \% 2 = 0$ 인 경우와 $k \% 2 = 1$ 인 경우로 누구의 승리인지 판단할 수 있다.

6.2 빙고(가로, 세로) 시 승자 출력 후 종료

1. 코드블록 스크린샷

```
//6.2 빙고(가로,세로) 시 승자 출력 후 종료
for(int i = 0; i < numCell; i++){
    if((board[i][0]!=' ') && (board[i][0] == board[i][1]) && (board[i][0] == board[i][2])){
        cout << "가로줄로 모두 들이 놓았습니다!: ";
        if(k%2) == 0{
            cout << "첫번째 유저(x)의 승리입니다!" << endl;
            cout << "게임을 종료합니다." << endl;
            return 0;
        }else{
            cout << "두번째 유저(o)의 승리입니다!" << endl;
            cout << "게임을 종료합니다." << endl;
            return 0;
        }
    }

    if((board[0][i] != ' ') && (board[0][i] == board[1][i]) && (board[0][i] == board[2][i])){
        cout << "세로줄로 모두 들이 놓았습니다!: ";
        if(k%2) == 0{
            cout << "첫번째 유저(x)의 승리입니다!" << endl;
            cout << "게임을 종료합니다." << endl;
            return 0;
        }else{
            cout << "두번째 유저(o)의 승리입니다!" << endl;
            cout << "게임을 종료합니다." << endl;
            return 0;
        }
    }
}
k++;
```

2. 입력

board[i][0] = i 번째 열의 0 번째 행 부분

board[i][1] = i 번째 열의 1 번째 행 부분

board[i][2] = i 번째 열의 2 번째 행 부분

board[0][i] = 0 번째 열의 i 번째 행 부분

board[1][i] = 1 번째 열의 i 번째 행 부분

board[2][i] = 2 번째 열의 i 번째 행 부분

3. 결과

열이 같고 행이 모두 다를 경우 가로로 돌이 모두 놓인 것이기 때문에 가로줄 멘트 출력후 k 나누기 2의 나머지가 0일 경우 x의 승리 멘트 출력 k 나누기 2의 나머지가 1일 경우 o의 승리 멘트 출력

행이 같고 열이 다를 경우 세로로 돌이 모두 놓인 것이기 때문에 세로줄 멘트 출력후 k 나누기 2의 나머지가 0일 경우 x의 승리 멘트 출력 k 나누기 2의 나머지가 1일 경우 o의 승리 멘트 출력

4. 설명

1. if와 &&를 활용 열이 같고 행이 모두 다를 경우 가로의 줄이 완성이 된 것이다. 따라서 누군가의 승리이다.
2. if와 &&를 활용 행이 같고 열이 모두 다를 경우 세로줄이 완성이 된 것이다 따라서 누군가의 승리이다.
3. 누군가의 승리일 경우 차례를 알면 누구의 승리인지 판단할 수 있기 때문에 $k \% 2 = 0$ 일 경우와 1일 경우를 if 문을 활용하여 두가지의 경우로 나눠서 x나 o의 승리 멘트를 출력

7. 모든 칸이 찼으면 종료

1. 코드블록 스크린샷

```
//7. 모든 칸이 찼으면 종료
if(k == numCell * numCell){
    cout << "모든 칸이 찼으므로 게임을 종료합니다." << endl;
    return 0;
}
```

2. 입력

k = 누구 차례인지 체크하기 위한 변수

`numCell = 가로 / 세로 칸 개수`

3. 결과

k 가 보드판의 모든 칸의 개수가 되었을 때 게임이 종료 된다.

4. 설명

1. k 가 `numCell * numCell` 즉, 9가 될 경우 돌은 총 9번 둔 것으로 보드판의 칸이 꽉 찬 것이므로 if문을 활용하여 $k = 9$ 가 될 경우 프로그램을 종료한다.

4. 테스트

1. 가능 별 테스트 결과:

1. 누구의 차례인지 출력

첫 번째 유저 (x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : █

두 번째 유저 (0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : █

2. 좌표 입력 받기

첫 번째 유저 (x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 2

3. 입력 받은 좌표 유효성 체크

첫 번째 유저 (x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 3 3
3, 3: x와 y 둘 중 하나가 칸을 벗어납니다.

첫 번째 유저 (x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 4
0, 4: x와 y 둘 중 하나가 칸을 벗어납니다.

첫 번째 유저 (x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 4 1
4, 1: x와 y 둘 중 하나가 칸을 벗어납니다.

4, 5. 좌표에 O / X 놓기, 현재 보드판 출력

첫 번째 유저 (x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 0 2



두 번째 유저 (0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 1



6. 빙고 시 승자 출력 후 종료

세로줄로 모두 돌이 놓였습니다!: 첫번째 유저(x)의 승리입니다!
게임을 종료합니다.

왼쪽 위에서 오른쪽 아래 대각선으로 모두 돌이 놓였습니다!: 첫번째 유저(x)의 승리입니다!
게임을 종료합니다.

오른쪽 위에서 왼쪽 아래 대각선으로 모두 돌이 놓였습니다!: 두번째 유저(o)의 승리입니다!
게임을 종료합니다.

가로줄로 모두 돌이 놓였습니다!: 두번째 유저(o)의 승리입니다!
게임을 종료합니다.

7. 모든 칸이 찼으면 종료

첫번째 유저(x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 1

---	---	---
X	0	0
---	---	---
0	X	X
---	---	---
X	X	0
---	---	---

모든 칸이 찼으므로 게임을 종료합니다.

2. 최종 테스트 스크린샷: (프로그램 전체 동작 스크린샷)

두번째 유저(o)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 1 2

---	---	---
X	X	
---	---	---
0	0	0
---	---	---
		X
---	---	---

가로줄로 모두 돌이 놓였습니다!: 두번째 유저(o)의 승리입니다!
게임을 종료합니다.

첫번째 유저(x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 1 2

---	---	---
0		X
---	---	---
		X
---	---	---
	0	X
---	---	---

세로줄로 모두 돌이 놓였습니다!: 첫번째 유저(x)의 승리입니다!
게임을 종료합니다.

```

첫 번째 유저 (x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 2
---|---|---
X |   | 0
---|---|---
0 | X |   |
---|---|---
   |   | X
---|---|---

원쪽 위에서 오른쪽 아래 대각선으로 모두 돌이 놓였습니다!: 첫 번째 유저 (x)의 승리입니다!
게임을 종료합니다.

```

```

두 번째 유저 (0)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 0
---|---|---
X | X | 0
---|---|---
   |   |   |
---|---|---
0 |   | X
---|---|---

오른쪽 위에서 원쪽 아래 대각선으로 모두 돌이 놓였습니다!: 두 번째 유저 (0)의 승리입니다!
게임을 종료합니다.

```

```

첫 번째 유저 (x)의 차례입니다 -> (x, y) 좌표를 입력하세요 : 2 1
---|---|---
X | 0 | 0
---|---|---
0 | X | X
---|---|---
X | X | 0
---|---|---

모든 칸이 찬으로 게임을 종료합니다.

```

5. 결과 및 결론

- 프로젝트 결과: Tic Tac Toe 게임을 만들었음. Tic Tac Toe 3인용은 만들지 못함.
- 느낀 점: 코드들이 대부분 작성되어 있어서 3인용 또한 시간 안에 만들 수 있을 거라고 생각했지만 2차원 배열에 관한 프로그램이다 보니 기본 Tic Tac Toe 게임 만드는 데에도 시간이 많이 걸렸다. 조금 더 배열의 차원에 관해서 많은 생각을 해준 프로젝트였던 것 같다.