R E P O R T

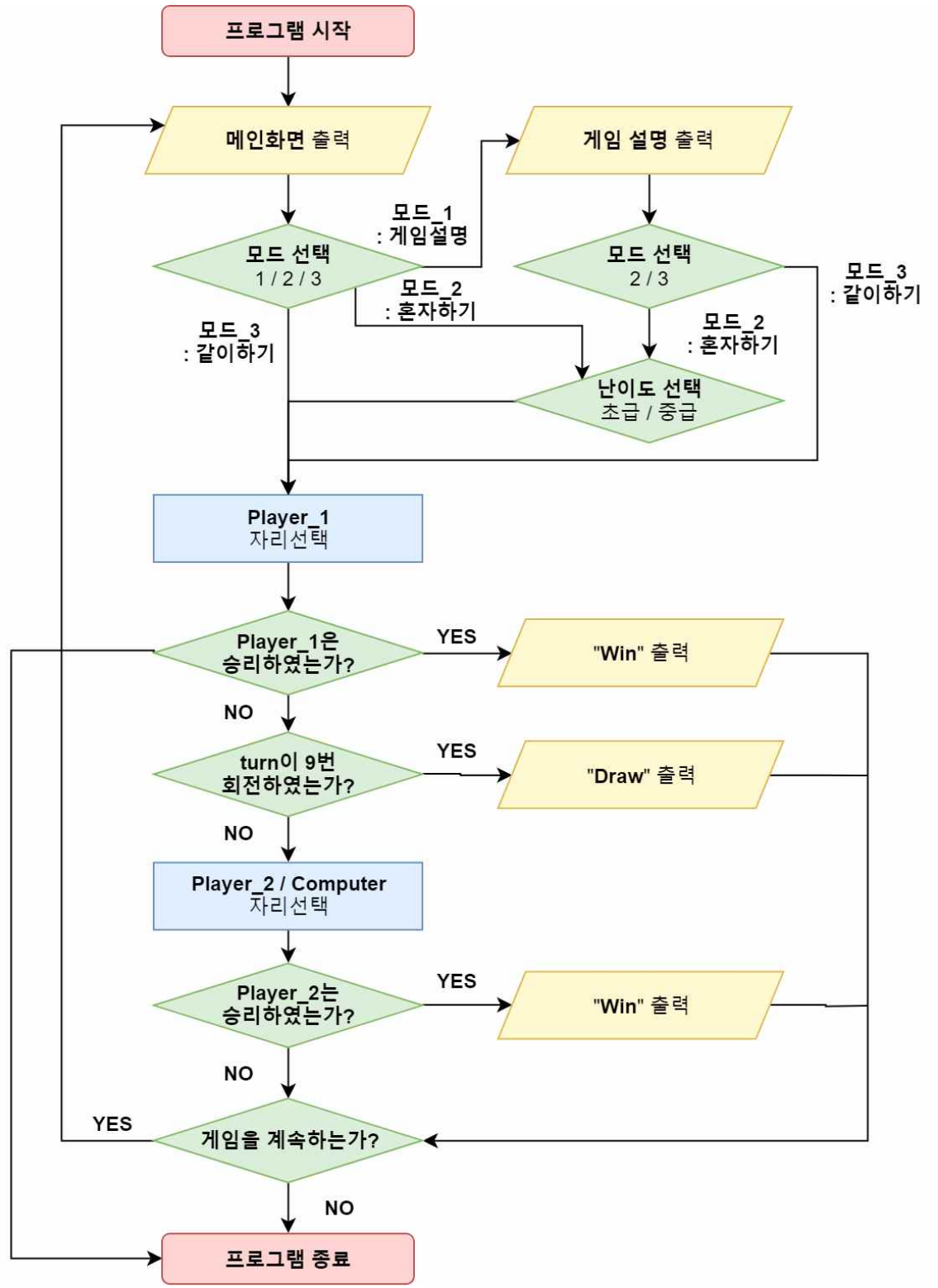
Tic Tac Toe 게임 만들기

학 과	창의소프트웨어공학부
교수님	박유현
학 번	20193148
분 반	002
이 름	황진주
제출일	2019.10.14

DONG-EUI UNIVERSITY
www.deu.ac.kr

목차

〈 순서도 〉


프로그램 시작
메인화면 출력
게임 설명 출력
모드_1
: 게임설명
모드 선택
1 / 2 / 3
모드 선택
2 / 3
모드_3
: 같이하기
모드_2
: 혼자하기
모드_3
: 같이하기
모드_2
: 혼자하기
난이도 선택
초급 / 중급
Player_1
자리선택
Player_1은
승리하였는가?
YES
"Win" 출력
NO
turn이 9번
회전하였는가?
YES
"Draw" 출력
NO
Player_2 / Computer
자리선택
Player_2는
승리하였는가?
YES
"Win" 출력
NO
YES
게임을 계속하는가?
NO
프로그램 종료

<게임 알고리즘>

1. 메인

- 1. 게임 설명
- 2. 1인용
- 3. 2인용

2. 게임 설명

1. 두 명이 번갈아가며 말을 둡니다.
2. 1,2,3,4,5,6,7,8,9 중 번호를 선택해
말을 둘 위치를 선택합니다.
3. player1 말은 'O' player2은 'X'로 표기됩니다.
4. 가로, 세로, 대각선 중 하나가 연결되면 승리합니다.

3. 게임 진행

< 이기는 경우의 수 >

- 가로 : {1, 2, 3}, {4, 5, 6}, {7, 8, 9}
- 세로 : {1, 4, 7}, {2, 5, 8}, {3, 6 ,9}
- 대각 : {1, 5, 8}, {3, 5, 7}

각각의 경우의 처음 중간 끝의 수를 따로 저장

< 말을 두는 법 >

1. char형의 array를 만들어 초기값을
{'0', '1', '2', '3', '4', '5', '6', '7', '8', '9'} 로 설정
2. 사용자에게 숫자를 입력받음 그 번호를 인덱스로 사용해 값을 수정

< 차례 >

turn을 세어 짝수일 때 'O', 홀수 일 때 'X'의 차례로 본다.

〈 1인용 게임 〉

: 컴퓨터가 말을 두는 알고리즘을 제작

- 초급 : 랜덤으로 아무데나 놓는다.
- 중급 : 이기는 경우를 for문으로 체크 후 공격 혹은 방어

〈 2인용 게임 〉

: 두 사람이 번갈아가며 말을 둔다.

4. 게임 종료

〈 WIN 〉

이기는 경우의 수를 array에 저장

```
array <int, 8> front = { 1,4,7,1,2,3,1,3 };
array <int, 8> middle = { 2,5,8,4,5,6,5,5 };
array <int, 8> end = { 3,6,9,7,8,9,9,7 };
```

for문을 이용해 승리하는 경우의 수를 모두 돌려봄
이 때 처음 == 중간 && 중간 == 끝 인 경우 참을 반환
for문이 끝나도 조건 충족 실패 시 거짓 반환

〈 DRAW 〉

플레이어 혹은 컴퓨터가 말을 둘 때 마다 count를 증가시킴
count가 9가 되는 순간 draw를 출력하며 게임 종료

〈 게임 재실행 〉

코드 전체를 while문에 넣음 조건에 true 상태인 bool 변수를 넣음
게임이 끝나고 게임 재실행 의사를 묻고 다시 하지 않을 다는 입력이
들어온 경우 이 bool을 false로 바꾸어줌.

5. 예외 처리

1. int형 입력은 모두 string 형식으로 함
2. atoi를 이용해 변환 (변환 불가 시 0이 반환됨)
3. 코드 전체를 while문에 넣어 올바른 입력일 시에만 반환해줌

〈 함수 설명 〉

001

함수 이름	printBoard	반환형	void

사용 목적

보드판을 출력한다.

동작원리

System헤더 파일에 있는 함수들을 이용해 색을 입히거나 화면을 지운다.
전역에 선언된 보드판 정보를 가진 배열의 정보와 보드판을 프린트한다.
ConsoleTextAttribute와 system함수를 이용하여 색을 입히거나 콘솔창을 정리한다.

코드

```
void printBoard()
{
	system("cls");
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 14);
	cout << "\t======================";
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 6);
	cout << "\n\t  << Tic Tac Toe >>\n";
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 14);
	cout << "\t======================\n\n";

	resetColor();
	cout << "     Player 1 (O)  -  Player 2 (X)" << endl << endl;
	cout << endl;
	cout << "  ■■■■■■■■■■■■■■■■■■■" << endl;
	cout << "  ■\t\t\t\t■" << endl;
	cout << "  ■\t\t\t\t■" << endl;
	cout << "  ■\t      |      |      \t■" << endl;

	cout << "  ■\t  ";
	changeColor(1);
	cout << square[1];
	resetColor();
	cout << "  |  ";
	changeColor(2);
	cout << square[2];
	resetColor();
```

```
        cout << "   |   ";
        changeColor(3);
        cout << square[3];
        resetColor();
        cout << "\t\t■" << endl;

        cout << "   ■\t_____|_____|_____\t■" << endl;
        cout << "   ■\t      |      |      \t■" << endl;

        cout << "   ■\t   ";
        changeColor(4);
        cout << square[4];
        resetColor();
        cout << "   |   ";
        changeColor(5);
        cout << square[5];
        resetColor();
        cout << "   |   ";
        changeColor(6);
        cout << square[6];
        resetColor();
        cout << "\t\t■" << endl;

        cout << "   ■\t_____|_____|_____\t■" << endl;
        cout << "   ■\t      |      |      \t■" << endl;

        cout << "   ■\t   ";
        changeColor(7);
        cout << square[7];
        resetColor();
        cout << "   |   ";
        changeColor(8);
        cout << square[8];
        resetColor();
        cout << "   |   ";
        changeColor(9);
        cout << square[9];
        resetColor();
        cout << "\t\t■" << endl;
```

```
        cout << "   ■\t       |       |       \t■" << endl;
        cout << "   ■\t\t\t\t■" << endl;
        cout << "   ■\t\t\t\t■" << endl;
        cout << "   ■■■■■■■■■■■■■■■■■" << endl << endl;
}
```

002

함수 이름	printExplain	반환형	void

사용 목적

게임 설명 하고 다음 동작에 대한 안내를 한다.

동작원리

System헤더 파일에 있는 함수들을 이용해 색을 입히거나 화면을 지운다.

코드

```
void printExplain()
{
        system("cls");
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 014);
        cout << "\n\n\t\t<< tic tac toe 게임설명 >>\n\n\n";
        resetColor();
        cout << "     > Player1과 Player2는 번갈아가며 말을 둡니다.\n";
        cout << "     > Player1은 'O', Player2는 'X'로 표시됩니다.\n";
        cout << "     > 두 사람 중 가로, 세로, 대각선 중 하나라도 3개가 연결되면 이깁니
다.\n";
        cout << "     > 두 사람 모두 연결에 실패하면 비깁니다.\n\n";
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 9);
        cout << "              2. 혼자하기 \t 3. 두 명에서 하기\n\n";
        resetColor();
}
```

003

함수 이름	printMain	반환형	void

사용 목적

메인 화면을 출력 하고 다음 동작에 대한 안내를 한다.

동작원리

System헤더 파일에 있는 함수들을 이용해 색을 입히거나 화면을 지운다.

코드

```
void printMain()
{
	system("cls");
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 2);
	cout << endl << endl;
	cout << "---------------------------------------------------------------------------------------------------" << endl;
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 9);
	cout << "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=" << endl << endl;
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15);
	cout << " '          '               '              *              '               '  *            '" << endl;
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 12);
	cout << "\t" << R"(            ______________            ______________  ______________)" << endl;
	cout << "\t" << R"(      /                /     /                /         /    /)" << endl;
	cout << "\t" << R"(     /____         ____/     /____         ____/         /____  ____/)" << endl;
	cout << "\t" << R"(          /    /             /    /                  / /)" << endl;
	cout << "\t" << R"(          /    /    /          /    /                / /)" << endl;
```

```
        cout << "\t" << R"(            /    /       ___     /    /   ___     ___   /
/  __     ___)" << endl;
        cout << "\t" << R"(          /    /   /  /      /    /   /   \  /     /    /
/  \  /___\)" << endl;
        cout << "\t" << R"(           /____/    /    \___   /____/      \__/ \ \___
/____/    \__/  \___)" << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15);
        cout << "          '                '           *                              '
           '    " << endl;
        cout << " '                *                                    '              '
 *          '          '" << endl << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 9);
        cout                                                                          <<
"-----------=-------------=-------------=-------------=-------------=-----------
-------=-----------" << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 2);
        cout                                                                          <<
"-------------------------------------------------------------------------------
--------------------" << endl;

        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 14);
        cout << "\t\t1. 게임설명 \t2. 혼자 하기 \t3. 두 명에서 하기\n\n";
        cout << "\t\t\t원하는 모드의 번호를 입력해주세요 >> ";

        resetColor();

}
```

004

함수 이름	changeColor	반환형	void	매개변수	int

사용 목적

보드판에 'O'와 'X'의 색을 구분하여 출력하기 위함

동작원리

보드판 인덱스를 매개변수로 받아 그것이 'O'면 파란색 'X'면 빨강색 그 무엇도 아니면 흰색을 출력하도록 해주는 함수이다.

코드

```
void changeColor(int color)
{
	if (square[color] == 'O') {
		SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 9);
	}
	else if (square[color] == 'X') {
		SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
12);
	}
	else {
		SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
15);
	}
}
```

005

함수 이름	resetColor	반환형	void	매개변수	void

사용 목적

색을 변환 한 후 기본 색으로 되돌리기 위한 함수

동작원리

색을 변경하는 코드를 실행

코드

```
void resetColor()
{
SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15);
}
```

006

함수 이름	selectMode	반환형	int	매개변수	void

사용 목적

모드를 입력받아 메뉴 번호를 반환한다.
1 = 게임 설명, 2 = 싱글 플레이, 3 = 멀티 플레이

동작원리

무한 반복을 하고 있는 반복문에 코드를 넣고 옳은 값이 들어오는 경우에만 값을 반환하여 종료한다.
문자열로 모드를 입력받는다. 입력받은 문자열을 int형으로 변환해준다 유효한 값(1/ 2/ 3)인 경우에는 해당 숫자를 반환하고, 그렇지 않은 경우 재입력을 받는다.

코드

```
int selectMode()
{
	while (true) {
		string inputMode;
		cin >> inputMode;

		int mode;
		mode = atoi(inputMode.c_str());

		if (mode == 1)
			return mode;
		else if (mode == 2)
			return mode;
		else if (mode == 3)
			return mode;
		else {
			cout << "\n\t\t\t\t올바른 값을 입력해주세요.";
		}
	}
}
```

007

함수 이름	printTurn	반환형	void	매개변수	int, char

사용 목적

차례를 알려줌

동작원리

playerNumber 와 simbol을 입력받아 출력해야 할 색과 'O', 'X'여부를 판단한다.

코드

```
void printTurn(int playerNumber, char simbol)
{
	// Player1인 경우는 파란색
	if (playerNumber == 1) {
		SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightBlue);
	}
	// Player2, Computer인 경우는 빨간색으로 작성
	else if (playerNumber == 2 || playerNumber == 3){
		SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightRed);
	}
	// 사람인경우
	if (playerNumber == 1 || playerNumber == 2) {
		cout << "\tPlayer" << playerNumber;
		resetColor();
		cout << " (" << simbol << ")의 차례입니다.\n";
		cout << "     원하는 위치를 입력해주세요 >> ";
	}
	// 컴퓨터인 경우
	else {
		cout << "\tComputer" << playerNumber;
		resetColor();
		cout << " (" << simbol << ")의 차례입니다.\n";
		Sleep(800);
	}
}
```

008

함수 이름	playerChoice	반환형	void	매개변수	int

사용 목적

사용자가 말을 두도록 한다.

동작원리

예외 처리

1. '0' 혹은 '9'이상의 수를 입력해 보드판 범위를 벗어나거나 문자열과 같은 정수형으로 변환이 되지 않는 것이 입력으로 들어오는 경우 atoi의 결과가 '0'이 되는 경우

2. 'X' 나 'O'로 이미 선택이 되어있는 경우

차례 판정

1. 홀수인 경우는 'X' 짝수인 경우에는 'O'라고 판정한다.

코드

```
void playerChoice(int turn)
{
	while (true) {
		string inputPlace;
		cin >> inputPlace;
		int place = atoi(inputPlace.c_str());

		// 1~9번까지의 수가 아닌 경우와 atoi로 변환되지 않은 경우
		if (place > 9 || place == 0) {
			cout << "\r올바른 위치를 선택해주세요.\n\n";
		}
		// 'X'나 'O'로 선택되어 있는 경우
		else if (boardSet[place] == 'O' || boardSet[place] == 'X') {
			cout << "\r이미 선택된 자리 입니다.\n";
		}
		else {
			// turn이 홀수인 경우에는 'X'의 차례
			if (turn % 2) {
				boardSet[place] = 'X';
				return;
			}
			// turn이 짝수인 경우에는 'O'의 차례
```

```
                        else {
                                boardSet[place] = 'O';
                                return;
                        }
                }
        }
}
```

009

함수 이름	checkWin	반환형	bool	매개변수	void

사용 목적

사용자가 말을 두도록 한다.

동작원리

array <int, 8> front = { 1,4,7,1,2,3,1,3 };
array <int, 8> middle = { 2,5,8,4,5,6,5,5 };
array <int, 8> endNum = { 3,6,9,7,8,9,9,7 };
3개의 array에 이길 수 있는 케이스를 미리 입력해둔다.

이 조건은 반복문을 통해 탐색하여 승리 조건을 충족하였을 때 반복문 내에서 true값을 반환하여 게임을 종료한다.
조건 충족에 실패한 경우 false를 반환하여 게임을 계속 진행하도록 한다.

코드

```
bool checkWin()
{
        // 승리 조건을 반복문을 돌며 판독
        for (int i = 0; i < sizeof(front) / sizeof(int); i++) {
                if (boardSet[front[i]] == boardSet[middle[i]] && boardSet[middle[i]]
== boardSet[endNum[i]]) {
                        return true;
                }
        }
        return false;
}
```

010

함수 이름	choiceDifferenty	반환형	int	매개변수	void

사용 목적

사용자가 게임 난이도를 선택하게 한다.

동작원리

atoi를 이용해 예외 처리를 한다.
사용자에게 난이도를 입력받고 올바른 값이 입력이 된다면 반환해준다.

코드

```
int choiceDifferenty()
{
	cout << "\n\t\t\t\t난이도를 입력해주세요.\n\n";
	cout << "\t\t\t     1. 초급\t 2. 중급\n\t\t\t>>";
	while (true) {
		string inputDifferenty;
		cin >> inputDifferenty;                    // 난이도 선택

		int differenty;
		differenty = atoi(inputDifferenty.c_str());

		if (differenty == 1) {                     // 난이도 : 초급
			return differenty;
		}
		else if (differenty == 2) {                // 난이도 : 중급
			return differenty;
		}
		else {
			cout << "\n\t\t\t\t올바른 값을 입력해주세요.";
		}
	}
}
```

011

함수 이름	gameOver	반환형	bool	매개변수	void

사용 목적

사용자의 게임 진행 여부를 판단한다.

동작원리

atoi를 이용해 예외 처리를 한다.
사용자에게 게임 진행 여부를 입력받고 올바른 값이 입력이 된다면 반환해준다.

코드

```
bool gameOver() {
	while (true) {		// 프로그램 종료 의사 확인
		SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightYellow);
		cout << "\t\t\t1. 다시하기 \t 2.그만두기\n\n";

		string inputRestart;		// 종료 의사 확인
		resetColor();
		cin >> inputRestart;
		// string형식을 int 형식으로 변환
		int restart = atoi(inputRestart.c_str());

		if (restart == 1) {		// 프로그램 가동
			return true;
		}
		else if (restart == 2) {		// 프로그램 종료
			return false;
		}
		else {				// 예외 발생
			cout << "올바른 번호를 입력해주세요. \n\n";
		}
	}
}
```

012

함수 이름	printWin	반환형	void	매개변수	int

사용 목적

WIN을 출력한다.

동작원리

플레이어 정보를 넘겨받아 승자에 따라 색상을 변경후 출력한다.

코드

```
void printWin(int winNumber)
{

	int print;		// 승자에 따른 win 색상변화
	if (winNumber == 1) {
		print = lightBlue;
	}
	else {
		print = lightRed;
	}

	system("cls");
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
	cout << endl << endl;
	cout                                                                    <<
"------------------------------------------------------------------------
----------------------" << endl;
	cout                                                                    <<
"==========-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
-=-=-=-=-=-=-=-=-=-=" << endl << endl;
	cout << " '       '               '           *          '              '
	     *          '" << endl;
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), print);
	cout << "\t\t" << winner[winNumber] << endl;   // 승자 이름 출력
	cout << "\t\t" << R"(    *  ___  ___  ___  _________   _____        ___
 '              ')" << endl;
	cout << "\t\t" << R"(      /  / /  / /  / /__   ___/  /     | '   /  /
   ')" << endl;
```

```
        cout << "\t\t" << R"(   ' /   / /   / /   /     /   /       /   /|  |     /   /
                    ')" << endl;
        cout << "\t\t" << R"(    /   / /   / /   /     /   /   '   /   / |  |    /   /    '
       ')" << endl;
        cout << "\t\t" << R"(  /   / /   / /   /     /   /      /   /  |  |  /   /       *
    ')" << endl;
        cout << "\t\t" << R"( /   /_/   /_/   / *   /   /       /   /    |  | /   /
          )" << endl;
        cout << "\t\t" << R"( |             /   __/   /___   /   /     |  |/   /     ')" <<
endl;
        cout << "\t\t" << R"( \____/\____/   /________/ /__/      |_____/        *
         ')" << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
        cout << "          '                '            *                              '
              '    " << endl;
        cout << " '                *                                    '               '
    *          '           '" << endl << endl;
        cout                                                                          <<
"-------------------------------------------------------------------------------
----------------------" << endl;
        cout                                                                          <<
"-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=-=-=-=-=-=-=-=-=-=-=-" << endl;


        resetColor();
}
```

013

함수 이름	printDraw	반환형	void	매개변수	int

사용 목적

DRAW를 출력한다.

동작원리

콘솔에 출력되어 있는 화면을 지운 뒤 DRAW를 색상을 변경해가며 출력한다.

코드

```
void printDraw()
{
	system("cls");
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
	cout << endl << endl;
	cout                                                                    <<
"------------------------------------------------------------------------
------------------" << endl;
	cout                                                                    <<
"========================================================================
==================" << endl << endl;
	cout << " '       *               '                       '             *
	    *         '" << endl;
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightPurple);
	cout  <<  R"(                        __________              ___________
__________  '    ___  ___  ___)" << endl;
	cout << R"(           /   _____  \  *   /  ________  \      /  ______
|     /  / /  / /  /)" << endl;
	cout << R"(          /   /     \ \    / /       | |    / /     | |
' /  / /  / /  /)" << endl;
	cout << R"(    '    /   /      | |   / /_______ / /    / /_______| |
 /  / /  / /  /)" << endl;
	cout << R"(         /   /      | |   /  ________    / ' /  _________/
|  /  / /  / /  /)" << endl;
	cout << R"(        /   /   '  / /   /  /        \ \    / /        / /
/  /_/  /_/  /)" << endl;
	cout << R"(  '    /   /______/  /   /  /         | |   / /        / /
```

```
|             /)" << endl;
        cout << R"(       |____________/   /__/          |__|  /__/   '    /__/ *
\____/\____/)" << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
        cout << "          '                          '                          '
             '   " << endl;
        cout << " '         '           *                        '          *       '
       '          '" << endl << endl;
        cout                                                                    <<
"_____________________________________________________________________________
__________________" << endl;
        cout                                                                    <<
"-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=-=-=-=-=-=-=-" << endl;


        resetColor();
}
```

014

함수 이름	computerChoice Beginner	반환형	void	매개변수	void

사용 목적

'혼자하기'에서 '초급' 단계를 고른 경우 랜덤하게 자리를 선택한다.

동작원리

랜덤으로 1~9의 수를 고르고 'O' 혹은 'X'가 아니라면 그 자리를 'X'로 변화시켜 자리를 골랐음을 표시하고 종료한다.

코드

```
void computerChoiceBeginner()
{
	srand((unsigned int)time(NULL));
	while (true) {
		int num = rand() % 9 + 1;
		bool flag = true;
		// 이미 선택된 경우
		if (boardSet[num] == 'O' || boardSet[num] == 'X') {
			flag = false;	// flag값 변화로
		}
		if (flag) {			// 값이 return되지 않음
					// 그렇지 않다면 그 자리를 체크 후 return
			boardSet[num] = 'X';
			return;
		}
	}
}
```

015

함수 이름	computerChoice Intermediate	반환형	void	매개변수	void

사용 목적

'혼자하기'에서 '중급' 단계를 고른 경우 랜덤하게 자리를 선택한다.

동작원리

array <int, 8> front = { 1,4,7,1,2,3,1,3 };
array <int, 8> middle = { 2,5,8,4,5,6,5,5 };
array <int, 8> endNum = { 3,6,9,7,8,9,9,7 };
전역에 우승할 수 있는 경우를 미리 array에 저장해두었다.

컴퓨터는 자신의 말('X')이 두 개가 가로, 세로, 대각선 의 한 직선 상에 2개가 되어있다면 그 줄에 자신의 말('X')을 두어 게임에서 승리한다.

그와 반대로 자신의 말이 아닌 상대의 말('O')이 위와 같은 상황으로 놓여있다면, 그 줄에 자신의 말('X')을 두어 방어한다.

만약 위의 두 조건 모두가 충족되지 않았다면 computerChoiceBeginner 함수를 호출하여 랜덤하게 말을 둔다.

코드

```
void computerChoiceIntermediate()
{
	// 만약 일직선 상의 'X'가 2개면 'X'를 두어 한 줄을 마무리하라
	for (int i = 0; i < 8; i++) {
		// 경우의 수
		char frontToChar = front[i] + 48, middleToChar = middle[i] + 48, endToChar = endNum[i] + 48;

		// 공격
		if (boardSet[front[i]] == 'X' && boardSet[middle[i]] == 'X' && boardSet[endNum[i]] == endToChar) {
			boardSet[endNum[i]] = 'X';
			return;
		}
		else if (boardSet[front[i]] == 'X' && boardSet[endNum[i]] == 'X'&& boardSet[middle[i]] == middleToChar) {
			boardSet[middle[i]] = 'X';
			return;
```

```
                }
                else if (boardSet[middle[i]] == 'X' && boardSet[endNum[i]] == 'X'
&& boardSet[front[i]] == frontToChar) {
                        boardSet[front[i]] = 'X';
                        return;
                }
        }

        // 만약 일직선 상의 'O'가 2개면 'X'를 두어 한 줄을 마무리하라
        for (int i = 0; i < 8; i++) {
                char frontToChar = front[i] + 48, middleToChar = middle[i] + 48,
endToChar = endNum[i] + 48;

                // 수비
                if (boardSet[front[i]] == 'O' && boardSet[middle[i]] == 'O' &&
boardSet[endNum[i]] == endToChar) {
                        boardSet[endNum[i]] = 'X';
                        return;
                }
                else if (boardSet[front[i]] == 'O' && boardSet[endNum[i]] ==
'O'&& boardSet[middle[i]] == middleToChar) {
                        boardSet[middle[i]] = 'X';
                        return;
                }
                else if (boardSet[middle[i]] == 'O' && boardSet[endNum[i]] == 'O'
&& boardSet[front[i]] == frontToChar) {
                        boardSet[front[i]] = 'X';
                        return;
                }
        }

        // 위의 조건이 성립하지 않았다면
        // random하게 수를 두어라
        computerChoiceBeginner();
}
```

016

함수 이름	main	반환형	void	매개변수	void

사용 목적

함수들의 조합을 통해서 Tic Tac Toe게임을 제작한다.

동작원리

동작원리는 앞의 순서도로 대체합니다.

코드

```
int main() {

	bool startGame = true;		// 프로그램 종료 여부 판단
	printMain();
	while (startGame) {

		// 보드판 초기화
		boardSet = { 'o','1','2','3','4','5','6','7','8','9' };

		bool  gameMode = true;		// 게임 종료 여부 판단
		int mode, turnCount = 0;

		// 모드선택 : 1. 게임 설명		2. 1인 플레이	3. 2인 플레이
		mode = selectMode();

		// 1. 게임설명
		if (mode == 1) {
			printExplain();
		}

		// 2. 1인 플레이
		if (mode == 2) {
			// 컴퓨터 난이도 선택
			int difficulty = choiceDifferenty();
			while (gameMode) {

				printBoard();	// 보드 출력
				cout << endl;
	// 차례 출력 : 플레이어 정보와 'O'혹은 'X'의 아스키 코드값 전송
```

```
                printTurn(1, 79);
                // 플레이어의 선택
                playerChoice(turnCount);
                // 승리 판정
                if (checkWin()) {
                        // "WIN" 출력
                        printWin(1);
                        //게임 진행 여부 확인
                        startGame = gameOver();
                        printMain();
                        break;
                }
                // turn 횟수를 이용해 차례와 게임진행 파악
                turnCount++;
                // 9번이 진행되면 게임종료
                if (turnCount > 8) {
                        gameMode = false;
                        startGame = gameOver();
                        if (!gameMode) {
                                printMain();
                                break;
                        }
                }
                printBoard();
                printTurn(3, 88);
                // 난이도 선택에 따른 알고리즘 변화
                if (difficulty == 2) {
                        computerChoiceIntermediate();  // 중급
                }
                else if (difficulty == 1) {
                        computerChoiceBeginner();      // 초급
                }
                if (checkWin()) {
                        printWin(3);
                        startGame = gameOver();
                        printMain();
                        break;
                }
                turnCount++;
        }
```

```
                }

                // 3. 2인 플레이: 난이도를 제외한 동일한 로직
                else if (mode == 3) {
                        while (gameMode) {

                                printBoard();
                                cout << endl;
                                printTurn(1, 79);
                                playerChoice(turnCount);
                                if (checkWin()) {
                                        printWin(1);
                                        startGame = gameOver();
                                        printMain();
                                        break;
                                }
                                turnCount++;
                                if (turnCount > 8) {
                                        gameMode = false;
                                        printDraw();
                                        startGame = gameOver();
                                        if (!gameMode) {
                                                printMain();
                                                break;
                                        }
                                }
                                printBoard();
                                printTurn(2, 88);
                                playerChoice(turnCount);
                                if (checkWin()) {
                                        printWin(2);
                                        startGame = gameOver();
                                        printMain();
                                        break;
                                }
                                turnCount++;
                        }
                }
        }
}
```

전체 코드

```
#include <iostream>
#include <Windows.h>            // Adornment
#include <cstdlib>              // To use "atoi"
#include <string>               // To use "string"
#include <array>                // To use "array"
#include <random>
#include <ctime>
using namespace std;


// Print
void printBoard();
void printExplain();
void printMain();
void printTurn(int, char);

// Adornment
void changeColor(int);
void resetColor();

// Judgment
int selectMode();
void playerChoice(int);
bool checkWin();
int choiceDifferenty();
bool gameOver();

// Result
void printWin(int);
void printDraw();

// Difficulty
void computerChoiceBeginner();
void computerChoiceIntermediate();

// Change Board
array <char, 50> boardSet = { 'o','1','2','3','4','5','6','7','8','9' };
```

```
// Choice Case
array <int, 8> front = { 1,4,7,1,2,3,1,3 };
array <int, 8> middle = { 2,5,8,4,5,6,5,5 };
array <int, 8> endNum = { 3,6,9,7,8,9,9,7 };

// Player Name
string winner[] = { " ","\tPLAYER_1", "\tPLAYER_2", "\tCOMPUTER" };

// Text Color
enum color {
	black, blue, green, skyblue, red,
	purple, yellow, white, grey, lightBlue,
	lightGreen, lightSkyblue, lightRed,
	lightPurple, lightYellow, darkWhite
};


int main() {

	bool startGame = true;		// 프로그램 종료 여부 판단
	printMain();
	while (startGame) {

		// 보드판 초기화
		boardSet = { 'o','1','2','3','4','5','6','7','8','9' };

		bool  gameMode = true;			// 게임 종료 여부 판단
		int mode, turnCount = 0;

		// 모드선택 : 1. 게임 설명	2. 1인 플레이	3. 2인 플레이
		mode = selectMode();

		// 1. 게임설명
		if (mode == 1) {
			printExplain();
		}
```

```
// 2. 1인 플레이
// 컴퓨터 난이도 선택
if (mode == 2) {
        int difficulty = choiceDifferenty();
        while (gameMode) {

                printBoard();    // 보드 출력
                cout << endl;

                // 차례 출력 : 플레이어 정보와
                // 'O'혹은 'X'의 아스키 코드값 전송
                printTurn(1, 79);
                playerChoice(turnCount);    // 플레이어의 선택

                if (checkWin()) {            // 승리 판정
                        printWin(1);         // "WIN" 출력
                        //게임 진행 여부 확인
                        startGame = gameOver();
                        printMain();
                        break;
                }

                turnCount++;    // turn 차례와 게임진행도 파악
                // Draw 조건 설정
                if (turnCount > 8) {
                        gameMode = false;
                        printDraw();
                        startGame = gameOver();
                        if (!gameMode) {
                                printMain();
                                break;
                        }
                }
```

```
                printBoard();
                printTurn(3, 88);
                // 난이도 선택에 따른 알고리즘 변화
                if (difficulty == 2) {
                        computerChoiceIntermediate();  // 중급
                }
                else if (difficulty == 1) {
                        computerChoiceBeginner();      // 초급
                }
                if (checkWin()) {
                        printWin(3);
                        startGame = gameOver();
                        printMain();
                        break;
                }
                turnCount++;
        }
}

// 3. 2인 플레이: 난이도를 제외한 동일한 로직
else if (mode == 3) {
        while (gameMode) {

                printBoard();
                cout << endl;
                printTurn(1, 79);
                playerChoice(turnCount);
                if (checkWin()) {
                        printWin(1);
                        startGame = gameOver();
                        printMain();
                        break;
                }
```

```
                                turnCount++;
                                if (turnCount > 8) {
                                        gameMode = false;
                                        printDraw();
                                        startGame = gameOver();
                                        if (!gameMode) {
                                                printMain();
                                                break;
                                        }
                                }
                                printBoard();
                                printTurn(2, 88);
                                playerChoice(turnCount);
                                if (checkWin()) {
                                        printWin(2);
                                        startGame = gameOver();
                                        printMain();
                                        break;
                                }
                                turnCount++;
                        }
                }
        }
}

// 메인화면(Tic Tac Toe) 출력
void printMain()
{
        system("cls");  // 콘솔의 화면 지우기
        // 색상 변경
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
        cout << endl << endl;
        cout                                                                    <<
"_____________________________________________________________________________
_____________" << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), lightBlue);
        cout                                                                    <<
"=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
-=-=-=-=-=-=-=-=-=-=-=-=-=-=" << endl << endl;
```

```
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
darkWhite);
        cout << " '       '               '           *          '             '
             *         '" << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), lightRed);
        cout << "\t" << R"(           _____________                 _____________
_____________)" << endl;
        cout << "\t" << R"(      /               /     /              /        /
     /)" << endl;
        cout << "\t" << R"(     /___       ____/     /___       ____/        /___
____/)" << endl;
        cout << "\t" << R"(            /    /            /    /
   /     /)" << endl;
        cout << "\t" << R"(           /    /   /         /    /                /
  /)" << endl;
        cout << "\t" << R"(          /    /      __     /    /   __     __    /
 /  __     __)" << endl;
        cout << "\t" << R"(         /    /   /  /      /    /   /   \  /     /
/  /  \  /__\)" << endl;
        cout << "\t" << R"(        /____/   /   \__  /____/    \__/ \ \__ /____/
 \__/  \__)" << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
darkWhite);
        cout << "        '            '          *                          '
             '   " << endl;
        cout << " '             *                              '             '
    *         '         '" << endl << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), lightBlue);
        cout                                                                 <<
"-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-" << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
        cout                                                                 <<
"_____________________________________________________________________________________________________" << endl;

        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightYellow);
        cout << "\t\t1. 게임설명 \t2. 혼자 하기 \t3. 두 명에서 하기\n\n";
        cout << "\t\t\t원하는 모드의 번호를 입력해주세요 >> ";
```

```
	resetColor();	// 흰색으로 초기화

}

// 승자의 번호를 입력받아 그에 맞는 색의 "WIN" 출력
void printWin(int winNumber)
{

	int print;		// 승자에 따른 win 색상변화
	if (winNumber == 1) {
		print = lightBlue;
	}
	else {
		print = lightRed;
	}

	system("cls");
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
	cout << endl << endl;
	cout                                                                    <<
"_____________________________________________________________________________
_____________" << endl;
	cout                                                                    <<
"=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
-=-=-=-=-=-=-=-=-=-=-=-=" << endl << endl;
	cout << " '       '               '           *          '             '
	     *         '" << endl;
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), print);
	cout << "\t\t" << winner[winNumber] << endl;   // 승자 이름 출력
	cout << "\t\t" << R"(    *  ___  ___  ___  ________   ____        ___
'             ')" << endl;
	cout << "\t\t" << R"(      /  / /  / /  / /__    ___/  /     | '   /  /
 ')" << endl;
	cout << "\t\t" << R"(  ' /  / /  / /  /    /  /     /  /|  |    /  /
	          ')" << endl;
	cout << "\t\t" << R"(   /  / /  / /  /    /  /  '  /  / |  |   /  /   '
     ')" << endl;
	cout << "\t\t" << R"(  /  / /  / /  /    /  /     /  / | | /  /      *
   ')" << endl;
```

```
	cout << "\t\t" << R"( /  /_/  /_/  / *  /  /      /  /    |  | /  /
	)" << endl;
	cout << "\t\t" << R"( |            /  __/  /___  /  /    |  |/  /    ')" <<
endl;
	cout << "\t\t" << R"( \____/\____/  /________/ /__/      |_____/         *
	')" << endl;
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
	cout << "         '              '           *                      '
	      '    " << endl;
	cout << " '              *                              '           '
   *          '          '" << endl << endl;
	cout                                                                  <<
"____________________________________________________________________________
______________" << endl;
	cout                                                                  <<
"=================================================================
========================" << endl;

	resetColor(); // 색을 흰색으로 초기화하는 함수
}

// DRAW 출력
void printDraw()
{
	system("cls");
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
	cout << endl << endl;
	cout                                                                  <<
"____________________________________________________________________________
__________" << endl;
	cout                                                                  <<
"=================================================================
===================" << endl << endl;
	cout << " '        *              '                     '             *
	     *          '" << endl;
	SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightPurple);
	cout << R"(          _________        ___________       ___________
'    ___  ___  ___)" << endl;
	cout << R"(          /   _____  \  *  /  ________  \      /  ______   |
```

```
   /  / /  / /  /)" << endl;
        cout << R"(          /   /     \ \    / /        | |    / /      | |
' /  / /  / /  /)" << endl;
        cout << R"(    '    /   /      | |   /  /______ /  /    /  /______|  |
/  / /  / /  /)" << endl;
        cout << R"(        /   /      | |   /  _______    /  ' /  _______/  |
/  / /  / /  /)" << endl;
        cout << R"(       /   /   '  /  /   /  /        \ \    / /        /  /
/  /_/  /_/  /)" << endl;
        cout << R"(  '   /   /_____/  /   /  /         | |   / /        /  /   |
         /)" << endl;
        cout << R"(      |__________/   /__/           |__|  /__/   '    /__/ *
\____/\____/)" << endl;
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), green);
        cout << "        '                          '                              '
             '   " << endl;
        cout << " '        '          *                       '         *      '
       '         '" << endl << endl;
        cout                                                                  <<
"_____
_____" << endl;
        cout                                                                  <<
"-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=-=-=-=-=-=-=-=-=-" << endl;

        resetColor();
}
```

```
// 사용자의 입력을 통해 게임 종료 여부 반환
bool gameOver() {
	while (true) {			// 프로그램 종료 의사 확인
		SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightYellow);
		cout << "\t\t\t1. 다시하기 \t 2.그만두기\n\n";

		string inputRestart;	// 종료 의사 확인
		resetColor();
		cin >> inputRestart;
		// string형식을 int 형식으로 변환
		int restart = atoi(inputRestart.c_str());

		if (restart == 1) {	// 프로그램 가동
			return true;
		}
		else if (restart == 2) {  // 프로그램 종료
			return false;
		}
		else {			// 예외 발생
			cout << "올바른 번호를 입력해주세요. \n\n";
		}
	}
}

// 컴퓨터의 난이도 선택
int choiceDifferenty()
{
	cout << "\n\t\t\t\t난이도를 입력해주세요.\n\n";
	cout << "\t\t\t    1. 초급\t 2. 중급\n\t\t\t>>";
	while (true) {
		string inputDifferenty;
		cin >> inputDifferenty;			// 난이도 선택

		int differenty;
		differenty = atoi(inputDifferenty.c_str());

		if (differenty == 1) {			// 난이도 : 초급
			return differenty;
```

```
                }
                else if (differenty == 2) {                // 난이도 : 중급
                        return differenty;
                }
                else {
                        cout << "\n\t\t\t\t올바른 값을 입력해주세요.";
                }
        }
}




// 혼자하기 초급 난이도 알고리즘
void computerChoiceBeginner()
{
        srand((unsigned int)time(NULL));
        while (true) {
                int num = rand() % 9 + 1;
                bool flag = true;
                // 자리가 이미 선택된 경우
                if (boardSet[num] == 'O' || boardSet[num] == 'X') {
                        flag = false;              // flag를 통한 반환 제어
                }
                if (flag) {
                        boardSet[num] = 'X';    // 자리 선택
                        return;
                }
        }
}
```

```
// 혼자하기 중급 난이도 알고리즘
void computerChoiceIntermediate()
{
	// 만약 일직선 상의 'X'가 2개면 'X'를 두어 한 줄을 마무리하라
	for (int i = 0; i < 8; i++) {
		// 경우의 수
		char frontToChar = front[i] + 48;
		char middleToChar = middle[i] + 48;
		char endToChar = endNum[i] + 48;

		// 공격
		if (boardSet[front[i]] == 'X' && boardSet[middle[i]] == 'X' &&
boardSet[endNum[i]] == endToChar) {
			boardSet[endNum[i]] = 'X';
			return;
		}
		else if (boardSet[front[i]] == 'X' && boardSet[endNum[i]] ==
'X'&& boardSet[middle[i]] == middleToChar) {
			boardSet[middle[i]] = 'X';
			return;
		}
		else if (boardSet[middle[i]] == 'X' && boardSet[endNum[i]] == 'X'
&& boardSet[front[i]] == frontToChar) {
			boardSet[front[i]] = 'X';
			return;
		}
	}

	// 만약 일직선 상의 'O'가 2개면 'X'를 두어 한 줄을 마무리하라
	for (int i = 0; i < 8; i++) {
		char frontToChar = front[i] + 48;
		char middleToChar = middle[i] + 48;
		char endToChar = endNum[i] + 48;

		// 수비
		if (boardSet[front[i]] == 'O' && boardSet[middle[i]] == 'O' &&
boardSet[endNum[i]] == endToChar) {
			boardSet[endNum[i]] = 'X';
			return;
```

```
                }
                else if (boardSet[front[i]] == 'O' && boardSet[endNum[i]] ==
'O'&& boardSet[middle[i]] == middleToChar) {
                        boardSet[middle[i]] = 'X';
                        return;
                }
                else if (boardSet[middle[i]] == 'O' && boardSet[endNum[i]] ==
'O' && boardSet[front[i]] == frontToChar) {
                        boardSet[front[i]] = 'X';
                        return;
                }
        }

        // 위의 조건이 성립하지 않았다면
        // random하게 수를 두어라
        computerChoiceBeginner();
}

// 턴 횟수에 따른 플레이어 자리 선택
void playerChoice(int turn)
{
        while (true) {
                string inputPlace;
                cin >> inputPlace;
                int place = atoi(inputPlace.c_str());

                if (place > 9 || place == 0) {
// 1~9번까지의 수가 아닌 경우와 atoi로 변환되지 않은 경우
                        cout << "\r올바른 위치를 선택해주세요.\n\n";
                }
                else if (boardSet[place] == 'O' || boardSet[place] == 'X') {
// 'X'나 'O'로 선택되어 있는 경우
                        cout << "\r이미 선택된 자리 입니다.\n";
                }
                else {
                        if (turn % 2) { // turn이 홀수인 경우에는 'X'의 차례
                                boardSet[place] = 'X';
                                return;
                        }
```

```
                        else {          // turn이 짝수인 경우에는 'O'의 차례
                                boardSet[place] = 'O';
                                return;
                        }
                }
        }
}

// 승리 여부 확인
bool checkWin()
{
        // 승리 조건을 반복문을 돌며 판독
        for (int i = 0; i < front.size(); i++) {
                if (boardSet[front[i]] == boardSet[middle[i]] &&
boardSet[middle[i]] == boardSet[endNum[i]]) {
                        return true;
                }
        }
        return false;
}


void printTurn(int playerNumber, char simbol)
{
        // Player1인 경우는 파란색
        if (playerNumber == 1) {
                SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightBlue);
        }
        // Player2, Computer인 경우는 빨간색으로 작성
        else if (playerNumber == 2 || playerNumber == 3) {
                SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightRed);
        }
```

```
        // 사람인경우
        if (playerNumber == 1 || playerNumber == 2) {
                cout << "\tPlayer" << playerNumber;
                resetColor();
                cout << " (" << simbol << ")의 차례입니다.\n";
                cout << "    원하는 위치를 입력해주세요 >> ";
        }
        // 컴퓨터인 경우
        else {
                cout << "\tComputer" << playerNumber;
                resetColor();
                cout << " (" << simbol << ")의 차례입니다.\n";
                Sleep(800);
        }
}

// 모드 선택
int selectMode()
{

        while (true) {
                string inputMode;   // 모드 선정
                cin >> inputMode;  // 1. 게임 설명, 2. 혼자 하기, 3. 두 명에서 하기

                int mode;
                mode = atoi(inputMode.c_str());

                if (mode == 1) {
                        return mode;
                }
                else if (mode == 2) {
                        return mode;
                }
                else if (mode == 3) {
                        return mode;
                }
                else {  // 예외 처리
                        cout << "\n\t\t\t\t올바른 값을 입력해주세요.";
                }
```

```
        }
}

// 설명 출력
void printExplain()
{
        system("cls");
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), lightRed);
        cout << "\n\n\t\t<< tic tac toe 게임설명 >>\n\n\n";
        resetColor();
        cout << "    > Player1과 Player2는 번갈아가며 말을 둡니다.\n";
        cout << "    > Player1은 'O', Player2는 'X'로 표시됩니다.\n";
        cout << "    > 두 사람 중 가로, 세로, 대각선 중 하나라도 3개가 연결되면
이깁니다.\n";
        cout << "    > 두 사람 모두 연결에 실패하면 비깁니다.\n\n";
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), lightBlue);
        cout << "            2. 혼자하기 \t 3. 두 명에서 하기\n\n";
        resetColor();
}

// 게임보드 그리기
void printBoard()
{
        system("cls");
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightYellow);
        cout << "\t======================";
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), yellow);
        cout << "\n\t  << Tic Tac Toe >>\n";
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightYellow);
        cout << "\t======================\n\n";

        resetColor();
        cout << "    Player 1 (O)  -  Player 2 (X)" << endl << endl;
        cout << endl;
        cout << "  ■■■■■■■■■■■■■■■■■" << endl;
        cout << "  ■\t\t\t\t■" << endl;
        cout << "  ■\t\t\t\t■" << endl;
        cout << "  ■\t     |     |     \t■" << endl;
```

```
cout << "  ■\t  ";
changeColor(1);
cout << boardSet[1];
resetColor();
cout << "  |  ";
changeColor(2);
cout << boardSet[2];
resetColor();
cout << "  |  ";
changeColor(3);
cout << boardSet[3];
resetColor();
cout << "\t\t■" << endl;

cout << "  ■\t_____|_____|_____\t■" << endl;
cout << "  ■\t     |     |     \t■" << endl;

cout << "  ■\t  ";
changeColor(4);
cout << boardSet[4];
resetColor();
cout << "  |  ";
changeColor(5);
cout << boardSet[5];
resetColor();
cout << "  |  ";
changeColor(6);
cout << boardSet[6];
resetColor();
cout << "\t\t■" << endl;

cout << "  ■\t_____|_____|_____\t■" << endl;
cout << "  ■\t     |     |     \t■" << endl;

cout << "  ■\t  ";
changeColor(7);
cout << boardSet[7];
resetColor();
cout << "  |  ";
```

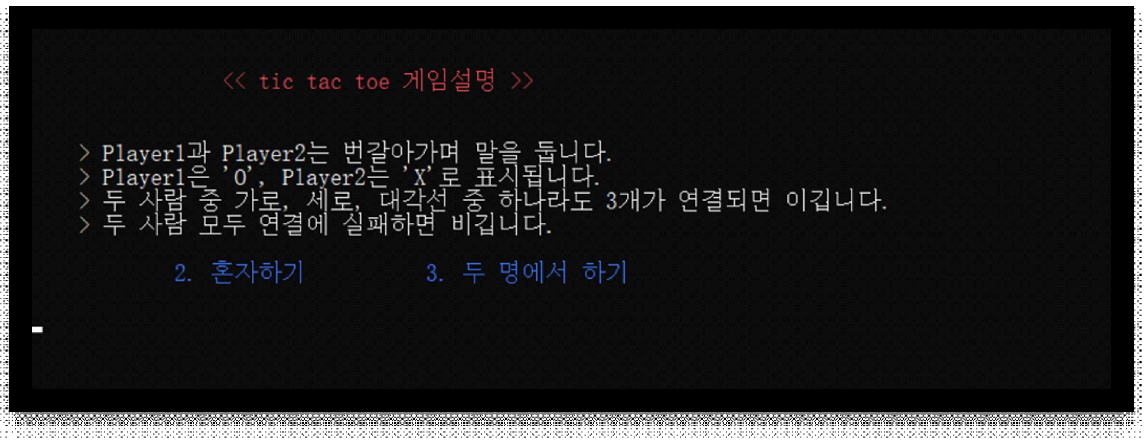
```
        changeColor(8);
        cout << boardSet[8];
        resetColor();
        cout << "  |  ";
        changeColor(9);
        cout << boardSet[9];
        resetColor();
        cout << "\t\t■" << endl;

        cout << "  ■\t      |      |      \t■" << endl;
        cout << "  ■\t\t\t\t■" << endl;
        cout << "  ■\t\t\t\t■" << endl;
        cout << "  ■■■■■■■■■■■■■■■■■■■" << endl << endl;
}

void changeColor(int color)
{
        // 보드판 출력시
        if (boardSet[color] == 'O') {     // 'O'면 파란색으로 변경
                SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightBlue);
        }
        else if (boardSet[color] == 'X') {        // 'X'면 빨간색으로 변경
                SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
lightRed);
        }
        else {  // 그 외에는 흰색
                SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
darkWhite);
        }
}

void resetColor()
{
        // 흰색으로 색 초기화시 사용
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
darkWhite);
}
```

〈 구동모습 〉

▼ 메인 화면


1. 게임설명 2. 혼자 하기 3. 두 명에서 하기
원하는 모드의 번호를 입력해주세요 >>

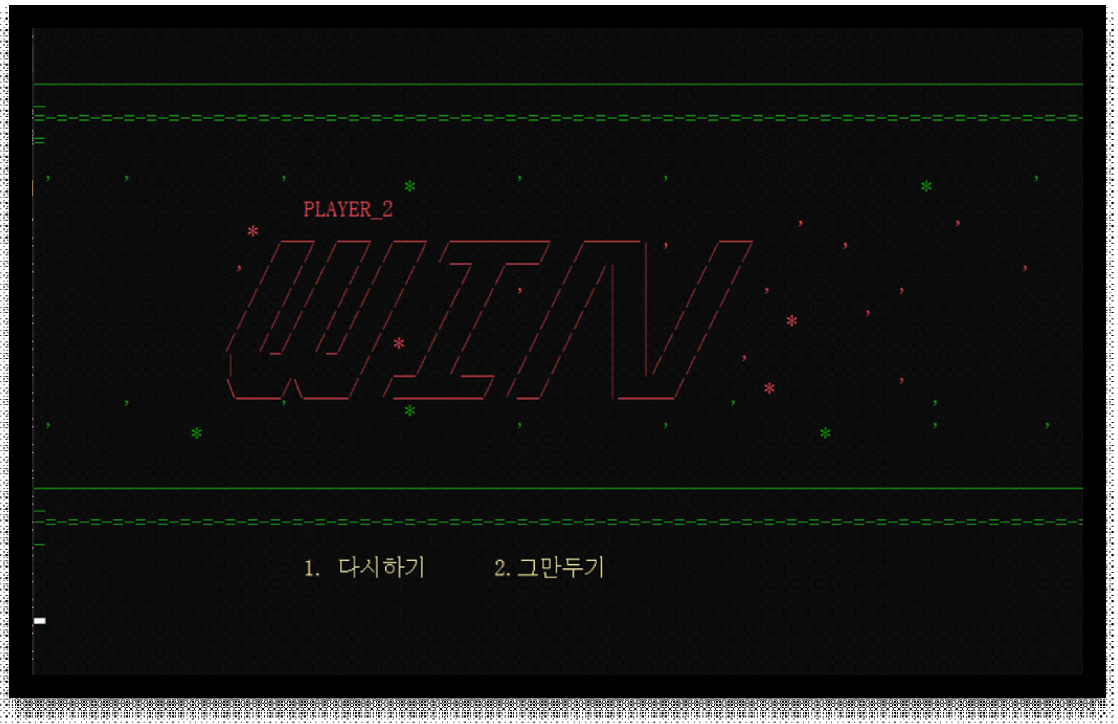
▼ 설명


<< tic tac toe 게임설명 >>
> Player1과 Player2는 번갈아가며 말을 둡니다.
> Player1은 'O', Player2는 'X'로 표시됩니다.
> 두 사람 중 가로, 세로, 대각선 중 하나라도 3개가 연결되면 이깁니다.
> 두 사람 모두 연결에 실패하면 비깁니다.
2. 혼자하기 3. 두 명에서 하기

▼ 게임 진행

▼ 게임 결과 - DRAW

▼ 게임 결과 - WIN



〈 피드백 〉

프로그램을 만들면서 코드 혹은 알고리즘의 수정이 많았었다. 적지 않은 내용의 코드를 수정하다가 이전의 코드로 되돌아가고 싶지만 너무 많은 양을 변경한 경우나 수정한 내용보다 이전의 내용이 더 좋아 과거의 코드가 필요한 경우가 있었다. github의 계정이 있었지만 그저 코딩 문제를 해결하고 그 코드들을 백업해 두는 정도로만 사용하였었다. 이번에 프로그램을 만들면서 git의 필요성을 느끼게 되었고, 미숙하게나마 사용을 해보았다. 큰 오류가 있었던 적은 없었지만 코드 수정 시에 도움을 받았었다. 현재는 복잡한 내용은 없지만 조금 더 큰 프로젝트나 복잡한 내용을 할 때 혹은 다른 사람들과 함께 코드를 작성할 때 그 유용함이 있을 것이라는 생각이 들었다.

1인용 게임을 만들면서 코드를 정리 능력이 부족함이 느껴졌다. 초기 알고리즘은 컴퓨터가 말을 둘 때의 모든 경우의 수를 if문과 else if문만을 이용해서 작성했다. 그러다보니 코드의 길이가 길어지고 실수가 생기기 쉬워 오류가 잦았었다. 모든 경우의 수를 효율적으로 확인하기 위한 알고리즘을 떠올리다 else if문들의 차이가 인덱스 숫자만의 변화라는 것을 보고 인덱스의 경우의 수를 처음, 중간, 끝으로 나누어 저장하였다. 그리고 for문을 이용하여 모든 경우의 수를 탐색하였다. 그 외에 처음 중간 끝의 경우를 3자리 수로 저장하여 연산을 통해 각 자리를 추출해 사용하는 방법 등 다른 방법을 떠올렸지만 만족스러울 정도의 완성도가 나오지 않아 고민이 되었다. 그래서 아쉬움이 많이 남게 되었다.

코드를 작성하면 할수록 새로운 기능 추가에 대한 욕구가 생겼다. 처음에는 단순한 2인용 게임에 단조로운 화면이었지만 아스키 아트를 추가하고, 색을 넣고, 컴퓨터와의 대결과 설명이 생겼다. 이번 과제를 하다 생긴 흥미로운 점이 많아서 제출 후에도 계속 코드들을 수정해갈 것 같다. 현재 추가 하고 싶은 기능은 '선공, 후공 선택', '승리 라운드 카운트', '혼자하기 고급 모드 추가', '마우스 조작 방식'의 기능으 추가하는 것을 목표로 두고 있다.