ㅋ

|  |
| --- |
| **일일 업무 사항 정리** |

|  |  |
| --- | --- |
| **작성자** | 제품팀 이민성 인턴 |
| **업무 일시** | 20230116~20230120 |

|  |
| --- |
| **세부 사항** |
| **1. 업무 내역 요약 정리**   |  |  | | --- | --- | | **목표 내역** | **Done & Plan** | | - 오류 2개 수정  - 제한 시간 적용 | **승자를 가리는 검사 알고리즘과 인공지능 알고리즘을 설계한 뒤 코드를 작성하였습니다.**  **하지만 좌표 하나만 입력해도 게임이 종료되는 오류와 첫 좌표를 입력 시 컴퓨터(X)가 3개를 연달아 입력하는 오류가 발생하였습니다. 현재 예상중인 원인은 rand 함수입니다. 최종 점검 전에 수정하겠습니다.**  **그리고 사용자(O)가 플레이 시 제한 시간을 적용하려고 했지만, 아직 작성하지 못하였습니다. 최종 점검 전에 작성하겠습니다.** | |

**2. 내용 세부 (업무 세부 내역 정리 및 기타 사항 정리)**

# 테이블이(가) 표시된 사진 자동 생성된 설명1. 검사 알고리즘

**<가로>**

(0,0) == (0,1) && (0,1) == (0,2)

(1,0) == (1,1) && (1,1) == (1,2)

(2,0) == (2,1) && (2,1) == (2,2)

🡪 (i,0) == (i,1) && (i,1) == (i,2)

🡪 (i,1) = 1 : O가 이김

(i,1) = 2 : X가 이김

**<세로>**

(0,0) == (1,0) && (1,0) == (2,0)

(0,1) == (1,1) && (1,1) == (2,1)

(0,2) == (1,2) && (1,2) == (2,2)

🡪 (0,i) == (1,i) && (1,i) == (2,i)

🡪 (1,i) = 1 : O가 이김

(1,i) = 2 : X가 이김

**<대각선>**

(0,0) == (1,1) && (1,1) == (2,2)

(0,2) == (1,1) && (1,1) == (2,0)

🡪 (1,1) = 1 : O가 이김

(1,1) = 2 : X가 이김

# 2. 인공지능 알고리즘

같은 것과 채워져 있는 것 비워져 있는 것 3가지를 생각해야 한다.

인공지능의 입장:

1. 2개가 같으면 막도록 해야 한다. (16가지)

<가로>

(0,0)==(0,1) && (0,0)=1 && (0,2)=0

(0,1)==(0,2) && (0,1)=1 && (0,0)=0

(1,0)==(1,1) && (1,0)=1 && (1,2)=0

(1,1)==(1,2) && (1,1)=1 && (1,0)=0

(2,0)==(2,1) && (2,0)=1 && (2,2)=0

(2,1)==(2,2) && (2,1)=1 && (2,0)=0

<세로>

(0,0)==(1,0) && (0,0)=1 && (2,0)=0

(1,0)==(2,0) && (1,0)=1 && (0,0)=0

(0,1)==(1,1) && (0,1)=1 && (2,1)=0

(1,1)==(2,1) && (1,1)=1 && (0,1)=0

(0,2)==(1,2) && (0,2)=1 && (2,2)=0

(1,2)==(2,2) && (1,2)=1 && (0,2)=0

<대각선>

(0,0)==(1,1) && (2,2)=0 && (0,0)=1

(1,1)==(2,2) && (0,0)=0 && (1,1)=1

(0,2)==(1,1) && (2,0)=0 && (0,2)=1

(2,0)==(1,1) && (0,2)=0 && (2,0)=1

🡪 I,j 변수를 이용하여 정리하면 8가지로 줄일 수 있다.

<가로> // y가 같다

(i,0)==(i,1) && (i,2)=0 && (i,1)=1

(i,1)==(i,2) && (i,0)=0 && (i,1)=1

<세로> // x가 같다

(0,i)==(1,i) && (2,i)=0 && (0,i)=1

(1,i)==(2,i) && (0,i)=0 && (1,i)=1

<대각선> // 변수 이용 불가

(0,0)==(1,1) && (2,2)=0 && (0,0)=1

(1,1)==(2,2) && (0,0)=0 && (1,1)=1

(0,2)==(1,1) && (2,0)=0 && (0,2)=1

(2,0)==(1,1) && (0,2)=0 && (2,0)=1

2. 가운데가 비어 있는 경우 막아야한다. (8가지)

<가로>

(0,0)==(0,2) && (0,1)=0 && (0,0)=1

(1,0)==(1,2) && (1,1)=0 && (1,0)=1

(2,0)==(2,2) && (2,1)=0 && (2,0)=1

<세로>

(0,0)==(2,0) && (1,0)=0 && (0,0)=1

(0,1)==(2,1) && (1,1)=0 && (0,1)=1

(0,2)==(2,2) && (1,2)=0 && (0,2)=1

<대각선>

(0,0)==(2,2) && (1,1)=0 && (0,0)=1

(0,2)==(2,0) && (1,1)=0 && (0,2)=1

🡪 I,j 변수를 이용하여 정리하면 4가지로 줄일 수 있다.

(i,0)==(i,2) && (i,1)=0 && (i,0)=1

(0,i)==(2,i) && (1,i)=0 && (0,i)=1

(0,0)==(2,2) && (1,1)=0 && (0,0)=1

(0,2)==(2,0) && (1,1)=0 && (0,2)=1

총 12가지이다. 나머지 경우의 수는 rand 함수를 이용하여 처리해준다.

# 3. tictactoe.c

#include <stdio.h>

#include <stdlib.h> // rand, srand

#include <time.h> // time(NULL)

int main**()**

**{**

int board**[**3**][**3**]={**0**,};** // 0으로 되어있는 3X3 보드

int i**,** j**,** flag**,** win**,** xin**,** yin**;**

int x**,** y**;**

/\* 플레이어가 O이다. \*/

srand**((**unsigned int**)** time**(NULL));** // rand 문제점 해결

**while(**1**)**

**{**

system**(**"clear"**);**

**for(**i**=**0**;** i**<**3**;** i**++)**

**{**

**for(**j**=**0**;** j**<**3**;** j**++)**

**{**

**if(**board**[**i**][**j**]** **==** 1**)**

printf**(**"O |"**);**

**else** **if(**board**[**i**][**j**]** **==** 2**)**

printf**(**"X |"**);**

**else** **if(**board**[**i**][**j**]** **==** 0**)**

printf**(**" |"**);**

**}**

printf**(**"\n"**);**

**}** // 출력 부분

printf**(**"O 차례 - y x 입력 : "**);**

scanf**(**"%d %d"**,** **&**y**,&**x**);**

**if(**board**[**y**][**x**]** **==** 0**)** // 보드가 비어있다면

board**[**y**][**x**]** **=**1**;** // o를 놓는다.

/\* 검사부분 \*/

flag **=** 0**;**

**for(**i**=**0**;** i**<**3**;** i**++)** // 가로

**{**

**if((**board**[**i**][**0**]** **==** board**[**i**][**1**])** **&&** **(**board**[**i**][**1**]** **==** board**[**i**][**2**]))**

**{**

**if(**board**[**i**][**1**]** **==** 1**)** // O가 이겼을 경우에

win **=** 1**,** flag **=** 1**;**

**if(**board**[**i**][**1**]** **==** 2**)** // X가 이겼을 경우에

win **=** 2**,** flag **=** 1**;**

**}**

**}**

**for(**i**=**0**;** i**<**3**;** i**++)** // 세로

**{**

**if((**board**[**0**][**i**]** **==** board**[**1**][**i**])** **&&** **(**board**[**1**][**i**]** **==** board**[**2**][**i**]))**

**{**

**if(**board**[**1**][**i**]** **==** 1**)** // O가 이겼을 경우에

win **=** 1**,** flag **=** 1**;**

**if(**board**[**1**][**i**]** **==** 2**)** // X가 이겼을 경우에

win **=** 2**,** flag **=** 1**;**

**}**

**}**

**for(**i**=**0**;** i**<**3**;** i**++)** // 대각선

**{**

**if((**board**[**0**][**0**]** **==** board**[**1**][**1**])** **&&** **(**board**[**1**][**1**]** **==** board**[**2**][**2**]))**

**{**

**if(**board**[**1**][**1**]** **==** 1**)** // O가 이겼을 경우에

win **=** 1**,** flag **=** 1**;**

**if(**board**[**1**][**1**]** **==** 2**)** // X가 이겼을 경우에

win **=** 2**,** flag **=** 1**;**

**}**

**}**

**for(**i**=**0**;** i**<**3**;** i**++)** // 대각선

**{**

**if((**board**[**0**][**2**]** **==** board**[**1**][**1**])** **&&** **(**board**[**1**][**1**]** **==** board**[**2**][**0**]))**

**{**

**if(**board**[**1**][**1**]** **==** 1**)** // O가 이겼을 경우에

win **=** 1**,** flag **=** 1**;**

**if(**board**[**1**][**1**]** **==** 2**)** // X가 이겼을 경우에

win **=** 2**,** flag **=** 1**;**

**}**

**}**

**if(**flag **==** 1**)**

**break;** // 나간다

flag **=** 1**;** // 비기는 경우를 위한 코드

**for(**i**=**0**;** i**<**3**;** i**++)**

**{**

**for(**j**=**0**;** j**<**3**;** j**++)**

**{**

**if(**board**[**i**][**j**]** **==** 0**)**

flag **=** 0**;** // flag = 1이면 빠져나온다

**}**

**}**

**if(**flag **==** 1**)**

**break;** // 나간다

/\* 인공지능 \*/

system**(**"clear"**);**

**for(**i**=**0**;** i**<**3**;** i**++)**

**{**

**for(**j**=**0**;** j**<**3**;** j**++)**

**{**

**if(**board**[**i**][**j**]** **==** 1**)**

printf**(**"O |"**);**

**else** **if(**board**[**i**][**j**]** **==** 2**)**

printf**(**"X |"**);**

**else** **if(**board**[**i**][**j**]** **==** 0**)**

printf**(**" |"**);**

**}**

printf**(**"\n"**);**

**}** // 출력부분

/\* 2개 이상 이어진 것을 찾아서 검사해서 한쪽에 놓는다. \*/

**for(**i**=**0**;** i**<**3**;** i**++)** // 반복

**{**

**if((**board**[**0**][**0**]** **==** board**[**1**][**1**])** **&&** **(**board**[**1**][**1**]** **==** 1**)** **&&** **(**board**[**2**][**2**]** **==** 0**))** // 대각선

**{**board**[**2**][**2**]** **=** 2**;** **break;}**

**else** **if((**board**[**1**][**1**]** **==** board**[**2**][**2**])** **&&** **(**board**[**1**][**1**]** **==** 1**)** **&&** **(**board**[**0**][**0**]** **==** 0**))** // 대각선

**{**board**[**0**][**0**]** **=** 2**;** **break;}**

**else** **if((**board**[**0**][**2**]** **==** board**[**1**][**1**])** **&&** **(**board**[**0**][**2**]** **==** 1**)** **&&** **(**board**[**2**][**0**]** **==** 0**))** // 대각선

**{**board**[**2**][**0**]** **=** 2**;** **break;}**

**else** **if((**board**[**2**][**0**]** **==** board**[**1**][**1**])** **&&** **(**board**[**2**][**0**]** **==** 1**)** **&&** **(**board**[**0**][**2**]** **==** 0**))** // 대각선

**{**board**[**0**][**2**]** **=** 2**;** **break;}**

**else** **if((**board**[**0**][**0**]** **==** board**[**2**][**2**])** **&&** **(**board**[**0**][**0**]** **==** 1**)** **&&** **(**board**[**1**][**1**]** **==** 0**))** // 대각선

**{**board**[**1**][**1**]** **=** 2**;** **break;}**

**else** **if((**board**[**0**][**2**]** **==** board**[**2**][**0**])** **&&** **(**board**[**0**][**2**]** **==** 1**)** **&&** **(**board**[**1**][**1**]** **==** 0**))** // 대각선

**{**board**[**1**][**1**]** **=** 2**;** **break;}**

**else**

**{**

**if((**board**[**i**][**0**]** **==** board**[**i**][**1**])** **&&** **(**board**[**i**][**1**]** **==** 1**)** **&** **(**board**[**i**][**2**]** **==** 0**))** // 가로

**{**board**[**i**][**2**]** **=** 2**;** **break;}**

**else** **if((**board**[**i**][**1**]** **==** board**[**i**][**2**])** **&&** **(**board**[**i**][**1**]** **==** 1**)** **&** **(**board**[**i**][**0**]** **==** 0**))** // 가로

**{**board**[**i**][**0**]** **=** 2**;** **break;}**

**else** **if((**board**[**i**][**0**]** **==** board**[**i**][**2**])** **&&** **(**board**[**i**][**0**]** **==** 1**)** **&** **(**board**[**i**][**1**]** **==** 0**))** // 가로

**{**board**[**i**][**1**]** **=** 2**;** **break;}**

**else** **if((**board**[**0**][**i**]** **==** board**[**1**][**i**])** **&&** **(**board**[**0**][**i**]** **==** 1**)** **&** **(**board**[**2**][**i**]** **==** 0**))** // 세로

**{**board**[**2**][**i**]** **=** 2**;** **break;}**

**else** **if((**board**[**1**][**i**]** **==** board**[**2**][**i**])** **&&** **(**board**[**1**][**i**]** **==** 1**)** **&** **(**board**[**0**][**i**]** **==** 0**))** // 세로

**{**board**[**0**][**i**]** **=** 2**;** **break;}**

**else** **if((**board**[**0**][**i**]** **==** board**[**2**][**i**])** **&&** **(**board**[**0**][**i**]** **==** 1**)** **&** **(**board**[**1**][**i**]** **==** 0**))** // 세로

**{**board**[**1**][**i**]** **=** 2**;** **break;}**

**else** // 오류 예상 부분

**{**

**while(**1**)**

**{**

xin **=** rand**()%**3**;**

yin **=** rand**()%**3**;**

**if(**board**[**yin**][**xin**]** **==** 0**)**

**{**board**[**yin**][**xin**]** **=** 2**;** **break;}**

**}**

**}**

**}**

**}**

/\* 검사부분 \*/

flag **=** 0**;**

**for(**i**=**0**;** i**<**3**;** i**++)** // 가로

**{**

**if((**board**[**i**][**0**]** **==** board**[**i**][**1**])** **&&** **(**board**[**i**][**1**]** **==** board**[**i**][**2**]))**

**{**

**if(**board**[**i**][**1**]** **==** 1**)** // O가 이겼을 경우에

win **=** 1**,** flag **=** 1**;**

**if(**board**[**i**][**1**]** **==** 2**)** // X가 이겼을 경우에

win **=** 2**,** flag **=** 1**;**

**}**

**}**

**for(**i**=**0**;** i**<**3**;** i**++)** // 세로

**{**

**if((**board**[**0**][**i**]** **==** board**[**1**][**i**])** **&&** **(**board**[**1**][**i**]** **==** board**[**2**][**i**]))**

**{**

**if(**board**[**1**][**i**]** **==** 1**)** // O가 이겼을 경우에

win **=** 1**,** flag **=** 1**;**

**if(**board**[**1**][**i**]** **==** 2**)** // X가 이겼을 경우에

win **=** 2**,** flag **=** 1**;**

**}**

**}**

**for(**i**=**0**;** i**<**3**;** i**++)** // 대각선

**{**

**if((**board**[**0**][**0**]** **==** board**[**1**][**1**])** **&&** **(**board**[**1**][**1**]** **==** board**[**2**][**2**]))**

**{**

**if(**board**[**1**][**1**]** **==** 1**)** // O가 이겼을 경우에

win **=** 1**,** flag **=** 1**;**

**if(**board**[**1**][**1**]** **==** 2**)** // X가 이겼을 경우에

win **=** 2**,** flag **=** 1**;**

**}**

**}**

**for(**i**=**0**;** i**<**3**;** i**++)** // 대각선

**{**

**if((**board**[**0**][**2**]** **==** board**[**1**][**1**])** **&&** **(**board**[**1**][**1**]** **==** board**[**2**][**0**]))**

**{**

**if(**board**[**1**][**1**]** **==** 1**)** // O가 이겼을 경우에

win **=** 1**,** flag **=** 1**;**

**if(**board**[**1**][**1**]** **==** 2**)** // X가 이겼을 경우에

win **=** 2**,** flag **=** 1**;**

**}**

**}**

**if(**flag **==** 1**)**

**break;** // 나간다

flag **=** 1**;** // (이해 X)

**for(**i**=**0**;** i**<**3**;** i**++)**

**{**

**for(**j**=**0**;** j**<**3**;** j**++)**

**{**

**if(**board**[**i**][**j**]** **==** 0**)**

flag **=** 0**;** // flag = 1이면 빠져나온다

**}**

**}**

**if(**flag **==** 1**)**

**break;** // 나간다

/\* 마무리 부분 \*/

system**(**"clear"**);**

**for(**i**=**0 **;** i**<**3 **;** i**++)**

**{**

**for(**j**=**0 **;** j**<**3 **;** j**++)**

**{**

**if(**board**[**i**][**j**]** **==** 1**)**

printf**(**"O |"**);**

**else** **if(**board**[**i**][**j**]** **==** 2**)**

printf**(**"X |"**);**

**else** **if(**board**[**i**][**j**]** **==**0**)**

printf**(**" |"**);**

**}**

printf**(**"\n"**);**

**}** // 출력부분

**if(**win **==**1**)**

printf**(**"\n\nO이 이겼습니다.\n"**);**

**else** **if(**win **==**2**)**

printf**(**"\n\nX이 이겼습니다.\n"**);**

**else**

printf**(**"\n\n비겼습니다.\n"**);**

**return** 0**;**

**}**

**}**