# 作品説明

## タイトル：五目並べAI

## 役割：メインはプレゼン作成、サブでプログラムを設計

## 想定する環境：unix環境での想定

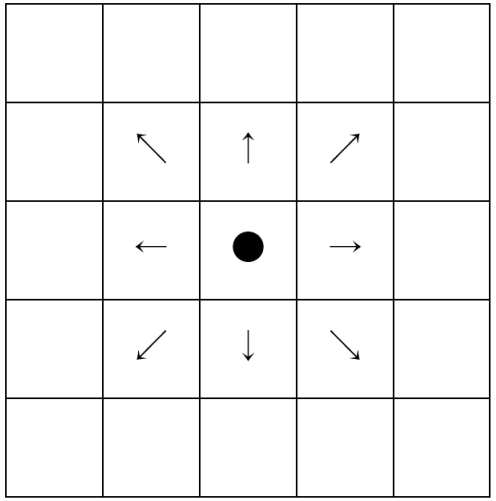
### 概要

・授業で4人1組でチームを作り、五目並べAIの作成をしました。
・「基本設計書」「詳細設計書」「プログラム」「プレゼン」4つの役割に分かれ活動しました。
・gomoku.cは先生が用意してくれたプログラムで、私達はgomoku_ai.cを作成しました。

製作期間：約3週間　使用ツール：Ubuntu,VScode

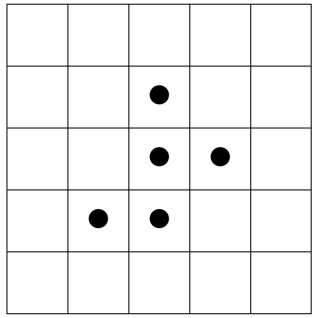
# プログラムの概要

## 打った石の8方向を探索→



この探索アルゴリズムは、方向の変化だけで
残りの処理は同じだと考えました。

位置の例→



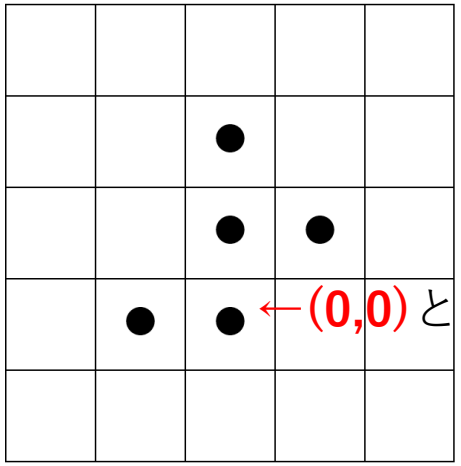
縦　：3個
横　：1個
左斜：1個
右斜：0個

1564行：check_around関数について

# プログラムの概要

## 上方向の場合

①上に石がある。
②上方向の石をカウントします。
③探索する座標を(0,0+1)に変更して再帰
④上に石がある。
⑤上方向の石をカウントします。
⑥探索する座標を（0,1+1)に変更して再帰
⑦上に石がない。
⑧その方向にある石の数分をy座標から引く（0,2-2（個））



←(0,0)とします。

## 下方向の場合

①下に石がない
②その方向にある石の数をy座標に足す(0,0+0(個））

真ん中の１つ下の座標を(0,0）とします。
上から反時計回りに処理が行われます。

この処理を8方向でそれぞれ行います。

# アピールポイント

**チームでプログラムを共有**することを考えて、
コメントやインデントを注視した。


```
void check_around(int board[][BOARD_SIZE], int com, int pos_x, int pos_y) {
    static int i = 0;
    static int Row[8] = { 1,1,1,1,1,1,1,1 };//8方向の自石の数をここに格納
    static int Total[4] = {0,0,0,0};//8方向を4方向に変換するTotal
    static int _x[8] = { 15,15,15,15,15,15,15,15 };//8近傍のxパターン　(相手石がある場合は15のままという処理)
    static int _y[8] = { 15,15,15,15,15,15,15,15 };//8近傍のyパターン　(相手石～略)
    static int max_val=0;

    while (i < 8) {
        switch (i)
        {
        case 0: //上,ここだけ詳しく書くけど、case1~7は方向が変わるだけで処理は一緒！
            if (board[pos_y - 1][pos_x] == com) {//自分の上に自石があったとき
                Row[i]++;//上方向に1足すという意味
                check_around(board, com, pos_x, pos_y-1);//ここがポイント再帰します.staticを使っているのでiは不変であるから,そこの列だけみることができる
                return;
            }
            else if(board[pos_y - 1][pos_x] == enemy) {//自分の上に相手の石だったとき　(ここに壁の時も入れたい！)
                pos_y += Row[i]-1;//リセットだけしてくれればいい
            }
            else {//そこにスペースが有る時は
                _y[i] = pos_y-1;//そのスペースの座標を入れる
                _x[i] = pos_x;//上記と同じ
                pos_y += Row[i]-1;//再帰した時用に,直前の一手にリセットする
            }
            break;
```

オンラインでの授業や役割の細分化により**分からない箇所を抱える問題**が出てきたので、
**2人１組のペア制度**を推進した結果、スムーズに成果物を作成することができました。

**チームでの作業と個人での作業には大きな違いがありました。**