알고리즘및실습
Term Project

3분반

한연희 교수님

컴퓨터공학부

2014136016 김선주

2014136041 노주영

목차

1. 서론

1) Term Project의 내용 및 목적

목적 : 픽셀게임의 돌을 두되 상대방이 4목을 만들면 자신이 이기는 변형된 알고리즘을 개발하는 것. 그동안 수업시간에 배웠던 여러 알고리즘들을 활용하여 변형된 4목 알고리즘을 개발하여 높은 점수를 기록하는 것.

픽셀 프로그램 소스 설명 :

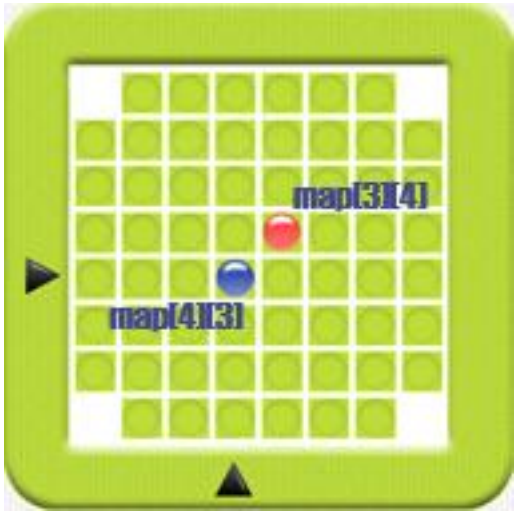
PixelTester.java	main 함수가 존재하는 Class
PixelBoard.java	그래픽 관련 처리 Class
Player.java	PixelPlayer를 위한 Abstract Class - currentPosition : 최근 자신이 둔 돌의 위치가 저장됨 - map : 현재 픽셀판의 정보가 담겨 있음 (0은 빈 공간, 1은 파란돌, 2는 빨간돌) - abstract Point nextPosition(Point currentPosition) : 상속 클래스에서 구현해야 하는 함수. 상대방이 가장 최근에 둔 돌의 위치를 입력으로 받아 현재 자신이 두고자 하는 돌의 위치를 리턴함.
PixelPlayer01.java	Player.java를 상속한 PixelPlayer 코드. 작성한 클래스 소스 - Player Class를 상속받음 - Point nextPosition() : 현재 주어진 map 정보에서 내가 이길 수 있는 가장 최적의 다음 돌 위치를 계산해 내는 알고리즘을 사용하여 구현함

Pixel 게임 규칙 :

1. 시작은 파란색(Player1) 돌의 위치인 map[4][3], 빨간색(Player2) 돌의 위치인 map[3][4]에 하나씩 두어진 상태로 시작한다. (map[4][3] = 1, map[3][4] = 2)

2. 슬라이더의 초기 좌표는 map[4][3]

3. 파란색 돌이 선공함



4. 본인의 4개의 돌이 일렬(가로, 세로, 대각선)로 먼저 놓아지면 패로 간주

5. nextPosition()이 반환하지 말아야 하는 위치
   - 이미 돌이 놓여진 위치
   - 행과 열이 동시에 바뀐 위치
   - 배열이 벗어나는 위치 (0~7 X 0~7 외부)
   - 코너, 즉 (0,0),(0,7),(7,0),(7,7) 위치

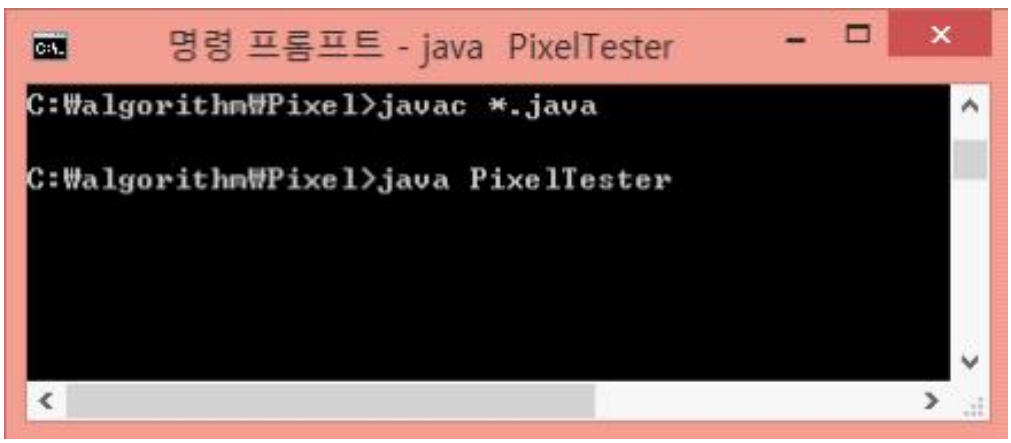
6. 위의 경우에 해당하는 위치가 반환될 시 패로 간주

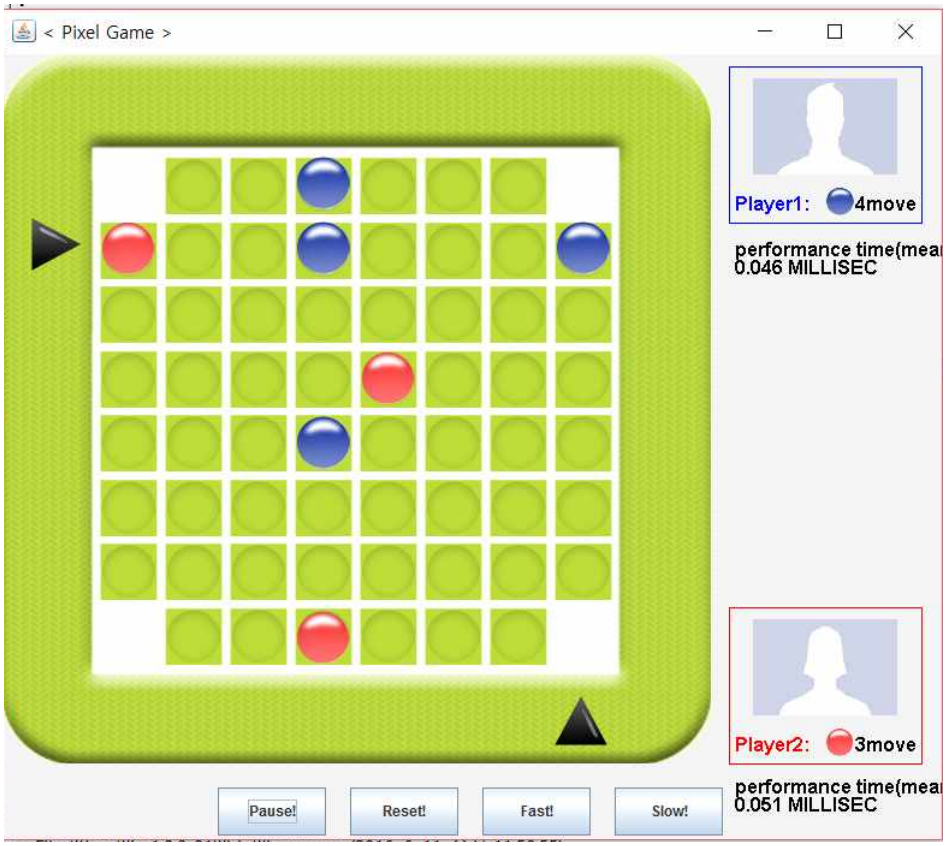
7. 무승부(Draw) 결과가 산출되는 경우
   - 맵 전체에 모든 돌이 놓여진 경우
   - 두 개의 슬라이더 중 어느 것을 움직여도 놓을 위치가 없는 경우

8. 무승부인 경우, nextPosition() 함수의 수(돌)당 평균 지연 시간을 비교하여 시간이 적게 걸린 Player를 “무승부승”으로 간주

9. 작성한 알고리즘이 에러를 발생시킬 경우, 패로 간주

실행 화면 :




< Pixel Game >
Player1: 4move
performance time(mea
0.046 MILLISEC
Player2: 3move
performance time(mea
0.051 MILLISEC
Pause!
Reset!
Fast!
Slow!

2. 본론

1) 픽셀 변형 알고리즘에 대한 전략, 절차, 방법

- 픽셀 변형 알고리즘에 대한 전략

1.둘 수 있는 곳이 없을 때 : 무승부
2.둘 수 있는 곳이 한곳일 때 : 무조건 그 곳에 둬야 한다.
3.둘 수 있는 곳이 여러 곳일 때 :
1) 근처에 내 돌 세 개가 없는 곳(= 뒀을 때 내가 4목이 되는 곳이 아닌 곳) // 배열에 3으로 저장
2) 상대방이 세 개가 있는 곳 빼고(= 상대방이 4목이 될 수 있는 곳이 아닌 곳)//배열에 4로 저장
-가장 외곽부터 둘 수 있도록
3) 근처에 내 돌 두개가 없는 곳 //배열에 5로 저장
-가장 외곽부터 둘 수 있도록
4) 남아있는 모든 곳의 근처에 같은 돌이 세 개 일 때는 어쩔 수 없지만 아무 곳에 넣는다(=지는 것)

- 픽셀 변형 알고리즘에 대한 전략

Greedy Algorithm을 이용하여 프로그램 설계

1. 내가 4목이 되지 않기 위한 최우선
선정절차: 근처에 내 돌이 세 개가 없을 만한 곳 선택
적정성 점검: 둘 수 있는 곳을 기준으로 가로, 세로, 상향대각선, 하향대각선에 각각 내 돌 세 개가 모여 있는지 확인
해답 점검: 근처에 없으면 그곳의 위치를 저장

2. 위의 1번을 만족한다면 상대방이 4목이 될 만한 곳은 두지 않도록
선정절차: 근처에 상대방 돌이 세 개가 없을 만한 곳 선택
적정성 점검: 둘 수 있는 곳을 기준으로 가로, 세로, 상향대각선, 하향대각선에 각각 상대방 돌 세 개가 모여 있는지 확인
해답 점검: 근처에 없으면 그곳의 위치를 저장

3. 위의 2번을 만족한다면 내가 최대한 안전할 수 있도록
선정절차: 근처에 내 돌이 두 개가 없을 만한 곳 선택
적정성 점검: 둘 수 있는 곳을 기준으로 가로, 세로, 상향대각선, 하향대각선에 각각 내돌 두 개가 모여 있는지 확인
해답 점검: 근처에 없으면 그곳의 위치를 저장

- 픽셀 변형 알고리즘에 대한 방법

먼저 상대방이 마지막으로 놓아둔 위치를 기준으로 내가 둘 수 있는 라인 두 개(가로, 세로)에 대한 배열을 만든다. 처음에 모두 -1로 지정해 놓고 for문을 이용해 검색하여 둘 수 있는 위치(빈 곳)를 파악해 0으로 바꾸어 놓는다. 그런 다음 0인 곳들 중에서 내가 4목이 되지 않는 곳을 다시 for문을 이용해 검색해 3으로 바꾸어 놓는다. 그리고 3인 곳들 중에서 상대방이 4목이 될 만한 위치가 아닌 곳을 for문을 이용해 검색해 4로 바꾸어 놓는다. 그 다음 4인 곳들 중에서 최대한 내 돌이 모여 있지 않도록 3개가 모이지 않을 곳들을 검색해 5로 저장해 둔다. 마지막으로 배열을 검색하여 해당 숫자가 있다면 5, 4, 3, 0의 순으로 그 위치에 내 돌을 놓는다.

2) 프로그램 코드

```
import java.awt.Point;

/* 1.둘 수 있는 곳이 없을 때 : 무승부
 * 2.둘 수 있는 곳이 한곳일 때 : 무조건 그 곳에 둬야 한다
 * 3.둘 수 있는 곳이 여러곳일 때 :
 *   1) 근처에 내 돌 세개가 없는 곳(= 뒀을 때 내가 4목이 되는 곳이 아닌 곳) // 배열에 3으로 저장
 *   2) 상대방이 세개가 있는 곳 빼고(= 상대방이 4목이 될 수 있는 곳이 아닌 곳)//배열에 4로 저장
 *              -가장 외곽부터 둘 수 있도록
 *   3) 근처에 내 돌 두개가 없는 곳 //배열에 5로 저장
 *              -가장 외곽부터 둘 수 있도록
 *   4) 남아있는 모든 곳의 근처에 같은 돌이 세개 일때는 어쩔 수 없지만 아무 곳에 넣는다(=지는 것)
 */

public class PixelPlayer01 extends Player {
        PixelPlayer01(int[][] map) {
                super(map);
        }

        public Point nextPosition(Point lastPosition) {
                int x = (int)lastPosition.getX(), y = (int)lastPosition.getY(); //상대방이 마지막으로 둔위치 x,y
                int myNum = map[(int)currentPosition.getX()][(int)currentPosition.getY()];//currentPosition은 최근 자기가 둔거 위치 저장
                int yourNum = map[(int)lastPosition.getX()][(int)lastPosition.getY()];//상대방의 돌 색깔을 나타내는 숫자

                Point nextPosition;//반환할 nextPosition변수 선언
                int[] xPosition=new int[PixelTester.SIZE_OF_BOARD]; //세로로 어디가 비었는지, 좋은 자리인지의 정보를 넣은 배열
                int[] yPosition=new int[PixelTester.SIZE_OF_BOARD]; //가로로 어디가 비었는지, 좋은 자리인지의 정보를 넣은 배열
```

```
		for(int i = 0; i < PixelTester.SIZE_OF_BOARD ; i++){// 배열 -1(= 둘 수 없는 곳)로 초기화
			xPosition[i] = -1;
			yPosition[i] = -1;
		}

		int count = 0; //빈 곳이 몇 개 남았는지 세는 변수
		//둘 수 있는 곳(= 0)이 얼마나 있는지 확인
		for(int i = 0; i < PixelTester.SIZE_OF_BOARD ; i++){//세로로 어디가 비었는지 확인
			if(map[i][y] == 0){
				count++;
				xPosition[i] = 0; //그 위치들 기억
			}
		}
		for(int j = 0; j < PixelTester.SIZE_OF_BOARD ; j++){//가로로 어디가 비었는지 확인
			if(map[x][j] == 0){
				count++;
				yPosition[j] = 0; // 그 위치들 기억
			}
		}

		//판단해서 돌 놓기
		if(count == 0){ //둘 곳 없을 때는 그냥 똑같은 거 반환해서 무승부 알려줌
			nextPosition = new Point(x, y);
			return nextPosition;
		}
		else if(count == 1){ //둘 곳 한 곳 남았을 때는 무조건 그곳에 둘 수 있도록
			for(int i = 0; i < PixelTester.SIZE_OF_BOARD ; i ++){
				if(xPosition[i] == 0){ //세로라인부터 남은 곳 확인
					nextPosition = new Point(i, y);
					return nextPosition;
				}
				if(yPosition[i] == 0){ //가로라인 남은 곳 확인해서 입력
					nextPosition = new Point(x, i);
					return nextPosition;
				}
			}
		}
		else{ //둘 수 있는 곳이 여러 곳 남았을 때
			for(int i = 0; i < PixelTester.SIZE_OF_BOARD ; i ++){
				boolean isBadPlace = false; //안좋은 자리 있는지 확인하는 변수(근처에 내 돌이 세개 있을 때)

				//BadPlace(근처에 내 것이 세개 있는 위치)확인
```

```
                        //세로선먼저 확인하자!
                        if(xPosition[i] == 0){
                                //세로줄의 상향 대각선
                                //첫번째 위치에 놓일 때 확인
                                if (i >= 0 && i <= 4 && y >= 3 && y <= 7 && map[i + 1][y -
1] == myNum
                                                && map[i + 2][y - 2] == myNum && map[i +
3][y - 3] == myNum) {
                                        isBadPlace = true;
                                }
                                // 두번째 위치에 놓일 때 확인
                                else if (i >= 1 && i <= 5 && y >= 2 && y <= 6 && map[i -
1][y + 1] == myNum
                                                && map[i + 1][y - 1] == myNum && map[i +
2][y - 2] == myNum) {
                                        isBadPlace = true;
                                }
                                // 세번째 위치에 놓일 때 확인
                                else if (i >= 2 && i <= 6 && y >= 1 && y <= 5 && map[i -
2][y + 2] == myNum
                                                && map[i - 1][y + 1] == myNum && map[i +
1][y - 1] == myNum) {
                                        isBadPlace = true;
                                }
                                // 네번째 위치에 놓일 때 확인
                                else if (i >= 3 && i <= 7 && y >= 0 && y <= 4 && map[i -
3][y + 3] == myNum
                                                && map[i - 2][y + 2] == myNum && map[i -
1][y + 1] == myNum) {
                                        isBadPlace = true;
                                }
                                // 세로줄의 하향대각선 확인
                                // 첫번째 위치에 놓일 때 확인
                                if (i >= 3 && i <= 7 && y >= 3 && y <= 7 && map[i - 1][y -
1] == myNum
                                                && map[i - 2][y - 2] == myNum && map[i -
3][y - 3] == myNum) {
                                        isBadPlace = true;
                                }
                                // 두번째 위치에 놓일 때 확인
                                else if (i >= 2 && i <= 6 && y >= 2 && y <= 6 && map[i -
2][y - 2] == myNum
                                                && map[i - 1][y - 1] == myNum && map[i +
1][y + 1] == myNum) {
```

```
                                        isBadPlace = true;
                                    }
                                    // 세번째 위치에 놓일 때 확인
                                    else if (i >= 1 && i <= 5 && y >= 1 && y <= 5 && map[i -
1][y - 1] == myNum
                                            && map[i + 1][y + 1] == myNum && map[i +
2][y + 2] == myNum) {
                                        isBadPlace = true;
                                    }
                                    // 네번째 위치에 놓일 때 확인
                                    else if (i >= 0 && i <= 4 && y >= 0 && y <= 4 && map[i +
1][y + 1] == myNum
                                            && map[i + 2][y + 2] == myNum && map[i +
3][y + 3] == myNum) {
                                        isBadPlace = true;
                                    }
                                    //세로줄의 세로 확인
                                    if(i >= 0 && i <= 4 && map[i+1][y]==myNum &&
map[i+2][y]==myNum && map[i+3][y]==myNum){//첫번째
                                        isBadPlace= true;
                                    }
                                    else if(i >= 1 && i <= 5 && map[i-1][y]==myNum &&
map[i+1][y]==myNum && map[i+2][y]==myNum){//두번째
                                        isBadPlace= true;
                                    }
                                    else if(i >= 2 && i <= 6 && map[i-2][y]==myNum &&
map[i-1][y]==myNum && map[i+1][y]==myNum){//세번째
                                        isBadPlace= true;
                                    }
                                    else if(i >= 3 && i <= 7 && map[i-3][y]==myNum &&
map[i-2][y]==myNum && map[i-1][y]==myNum){//네번째
                                        isBadPlace= true;
                                    }
                                    //세로줄의 가로 확인
                                    if(y >= 0 && y <= 4 && map[i][y+1]==myNum &&
map[i][y+2]==myNum && map[i][y+3]==myNum){//첫번째
                                        isBadPlace= true;
                                    }
                                    else if(y >= 1 && y <= 5 && map[i][y-1]==myNum &&
map[i][y+1]==myNum && map[i][y+2]==myNum){//두번째
                                        isBadPlace= true;
                                    }
                                    else if(y >= 2 && y <= 6 && map[i][y-2]==myNum &&
map[i][y-1]==myNum && map[i][y+1]==myNum){//세번째
```

```
                                        isBadPlace= true;
                                }
                                else if(y >= 3 && y <= 7 && map[i][y-3]==myNum &&
map[i][y-2]==myNum && map[i][y-1]==myNum){//네번째
                                        isBadPlace= true;
                                }
                                //세로줄 확인 끝나면
                                if(isBadPlace == false){
                                        xPosition[i] = 3;//나한테 불리한 곳 아니면 3을 넣는다.
                                }
                        }

                        //이제 가로선 확인시작!
                        if(yPosition[i] == 0){
                                //가로줄의 상향 대각선
                                //첫번째 위치에 놓일 때 확인
                                if (x >= 0 && x <= 4 && i >= 3 && i <= 7 && map[x + 1][i -
1] == myNum
                                                && map[x + 2][i - 2] == myNum && map[x +
3][i - 3] == myNum) {
                                        isBadPlace = true;
                                }
                                // 두번째 위치에 놓일 때 확인
                                else if (x >= 1 && x <= 5 && i >= 2 && i <= 6 && map[x -
1][i + 1] == myNum
                                                && map[x + 1][i - 1] == myNum && map[x +
2][i - 2] == myNum) {
                                        isBadPlace = true;
                                }
                                //세번째 위치에 놓일 때 확인
                                else if (x >= 2 && x <= 6 && i >= 1 && i <= 5 && map[x -
2][i + 2] == myNum
                                                && map[x - 1][i + 1] == myNum && map[x +
1][i - 1] == myNum) {
                                        isBadPlace = true;
                                }
                                //네번째 위치에 놓일 때 확인
                                else if (x >= 3 && x <= 7 && i >= 0 && i <= 4 && map[x -
3][i + 3] == myNum
                                                && map[x - 2][i + 2] == myNum && map[x -
1][i + 1] == myNum) {
                                        isBadPlace = true;
                                }
                                //가로줄의  햐향대각선 확인
```

```
                                        //네번째 위치에 놓일 때 확인
                                        if (x >= 0 && x <= 4 && i >= 0 && i <= 4 && map[x + 1][i +
1] == myNum && map[x + 2][i + 2] == myNum
                                                && map[x + 3][i + 3] == myNum) {

                                            isBadPlace = true;
                                        }
                                        //세번째 위치에 놓일 때 확인
                                        else if (x >= 1 && x <= 5 && i >= 1 && i <= 5 && map[x -
1][i - 1] == myNum
                                                && map[x + 1][i + 1] == myNum && map[x +
2][i + 2] == myNum) {
                                            isBadPlace = true;
                                        }
                                        //두번째 위치에 놓일 때 확인
                                        else if (x >= 2 && x <= 6 && i >= 2 && i <= 6 && map[x -
2][i - 2] == myNum
                                                && map[x - 1][i - 1] == myNum && map[x +
1][i + 1] == myNum) {
                                            isBadPlace = true;
                                        }
                                        //첫번째 위치에 놓일 때 확인
                                        else if (x >= 3 && x <= 7 && i >= 3 && i <= 7 && map[x -
3][i - 3] == myNum
                                                && map[x - 2][i - 2] == myNum && map[x -
1][i - 1] == myNum) {
                                            isBadPlace = true;
                                        }
                                        //가로줄의 세로 확인하고
                                        //첫번째 위치에 놓일 때 확인
                                        if(x >= 0 && x <= 4 && map[x+1][i]==myNum &&
map[x+2][i]==myNum && map[x+3][i]==myNum){
                                            isBadPlace= true;
                                        }
                                        //두번째 위치에 놓일 때 확인
                                        else if(x >= 1 && x <= 5 && map[x-1][i]==myNum &&
map[x+1][i]==myNum && map[x+2][i]==myNum){
                                            isBadPlace= true;
                                        }
                                        //세번째 위치에 놓일 때 확인
                                        else if(x >= 2 && x <= 6 && map[x-2][i]==myNum &&
map[x-1][i]==myNum && map[x+1][i]==myNum){
                                            isBadPlace= true;
                                        }
```

```
                                        //네번째 위치에 놓일 때 확인
                                        else if(x >= 3 && x <= 7 && map[x-3][i]==myNum &&
map[x-2][i]==myNum && map[x-1][i]==myNum){
                                                isBadPlace= true;
                                        }
                                        //가로줄의 가로 확인하고
                                        //첫번째 위치에 놓일 때 확인
                                        if(i >= 0 && i <= 4 && map[x][i+1]==myNum &&
map[x][i+2]==myNum && map[x][i+3]==myNum){
                                                isBadPlace= true;
                                        }
                                        //두번째 위치에 놓일 때 확인
                                        else if(i >= 1 && i <= 5 && map[x][i-1]==myNum &&
map[x][i+1]==myNum && map[x][i+2]==myNum){
                                                isBadPlace= true;
                                        }
                                        //세번째 위치에 놓일 때 확인
                                        else if(i >= 2 && i <= 6 && map[x][i-2]==myNum &&
map[x][i-1]==myNum && map[x][i+1]==myNum){
                                                isBadPlace= true;
                                        }
                                        //네번째 위치에 놓일 때 확인
                                        else if(i >= 3 && i <= 7 && map[x][i-3]==myNum &&
map[x][i-2]==myNum && map[x][i-1]==myNum){//네번째
                                                isBadPlace= true;
                                        }
                                        //가로줄 확인 끝나면
                                        if(isBadPlace == false){
                                                yPosition[i] = 3; //나한테 불리한 곳이 아니라는 뜻의 3
넣음
                                        }
                                }
                        }//BadPlace의 위치를 확인하기 위한 for문 끝

                        //BadPlace가 얼마나 있는지 확인하는 구간
                        int find_3 = 0; //BadPlace가 아닌 곳(3)을 세기 위한 count 변수
                        for(int i = 0; i < PixelTester.SIZE_OF_BOARD; i++ ){
                                if(xPosition[i] == 3){
                                        find_3++;
                                }
                                if(yPosition[i] == 3){
                                        find_3++;
                                }
                        }
```

```
                        // BadPlace가 얼마나 있는지 확인하는 구간 끝

                        if(find_3 == 0){//모두 BadPlace일때 (3이 하나도 없을 때) 빈 곳에 차례로 넣음
                                for(int i = 0; i < PixelTester.SIZE_OF_BOARD ; i ++){
                                        if(xPosition[i] == 0){ //세로부터 남은 곳 확인
                                                nextPosition = new Point(i, y);
                                                return nextPosition;
                                        }
                                        if(yPosition[i] == 0){ //가로 남은 곳 확인해서 입력
                                                nextPosition = new Point(x, i);
                                                return nextPosition;
                                        }
                                }
                        }
                        else if(find_3 != 0){//BadPlace가 아닌 구간 있으면
                                for(int i = 0; i< PixelTester.SIZE_OF_BOARD; i++){
                                        boolean isGoodPlace = false; // 상대방을 공격하기 위한 좋은
자리 = 상대방이 놓았을 때 사목이 되는 자리

                                        //GoodPlace(근처에 상대방 것이 세개 있는 위치)확인
                                        //세로선먼저 확인
                                        if(xPosition[i] == 3){
                                                //상향 대각선
                                                //첫번째 위치에 놓일 때 확인
                                                if (i >= 0 && i <= 4 && y >= 3 && y <= 7 && map[i
+ 1][y - 1] == yourNum
                                                                && map[i + 2][y - 2] == yourNum &&
map[i + 3][y - 3] == yourNum) {
                                                        isGoodPlace = true;
                                                }
                                                // 두번째 위치에 놓일 때 확인
                                                else if (i >= 1 && i <= 5 && y >= 2 && y <= 6 &&
map[i - 1][y + 1] == yourNum
                                                                && map[i + 1][y - 1] == yourNum &&
map[i + 2][y - 2] == yourNum) {
                                                        isGoodPlace = true;
                                                }
                                                // 세번째 위치에 놓일 때 확인
                                                else if (i >= 2 && i <= 6 && y >= 1 && y <= 5 &&
map[i - 2][y + 2] == yourNum
                                                                && map[i - 1][y + 1] == yourNum &&
map[i + 1][y - 1] == yourNum) {
                                                        isGoodPlace = true;
                                                }
```

```
                                        // 네번째 위치에 놓일 때 확인
                                        else if (i >= 3 && i <= 7 && y >= 0 && y <= 4 &&
map[i - 3][y + 3] == yourNum
                                                && map[i - 2][y + 2] == yourNum &&
map[i - 1][y + 1] == yourNum) {
                                            isGoodPlace = true;
                                        }
                                        //세로줄의  햐향대각선 확인
                                        //첫번째 위치에 놓일 때 확인
                                        if (i >= 3 && i <= 7 && y >= 3 && y <= 7 && map[i
- 3][y - 3] == yourNum
                                                && map[i - 2][y - 2] == yourNum &&
map[i - 1][y - 1] == yourNum) {
                                            isGoodPlace = true;
                                        }
                                        // 두번째 위치에 놓일 때 확인
                                        else if (i >= 2 && i <= 6 && y >= 2 && y <= 6 &&
map[i - 2][y - 2] == yourNum
                                                && map[i - 1][y - 1] == yourNum &&
map[i + 1][y + 1] == yourNum) {
                                            isGoodPlace = true;
                                        }
                                        // 세번째 위치에 놓일 때 확인
                                        else if (i >= 1 && i <= 5 && y >= 1 && y <= 5 &&
map[i - 1][y - 1] == yourNum
                                                && map[i + 1][y + 1] == yourNum &&
map[i + 2][y + 2] == yourNum) {
                                            isGoodPlace = true;
                                        }
                                        // 네번째 위치에 놓일 때 확인
                                        else if (i >= 0 && i <= 4 && y >= 0 && y <= 4 &&
map[i + 1][y + 1] == yourNum
                                                && map[i + 2][y + 2] == yourNum &&
map[i + 3][y + 3] == yourNum) {
                                            isGoodPlace = true;
                                        }
                                        //세로줄의 세로
                                        //첫번째 위치에 놓일 때 확인
                                        if(i >= 0 && i <= 4 && map[i+1][y]==yourNum &&
map[i+2][y]==yourNum && map[i+3][y]==yourNum){
                                            isGoodPlace= true;
                                        }
                                        //두번째 위치에 놓일 때 확인
                                        else if(i >= 1 && i <= 5 && map[i-1][y]==yourNum &&
```

```
map[i+1][y]==yourNum && map[i+2][y]==yourNum){
                                                isGoodPlace= true;
                                        }
                                        //세번째 위치에 놓일 때 확인
                                        else if(i >= 2 && i <= 6 && map[i-2][y]==yourNum &&
map[i-1][y]==yourNum && map[i+1][y]==yourNum){
                                                isGoodPlace= true;
                                        }
                                        //네번째 위치에 놓일 때 확인
                                        else if(i >= 3 && i <= 7 && map[i-3][y]==yourNum &&
map[i-2][y]==yourNum && map[i-1][y]==yourNum){
                                                isGoodPlace= true;
                                        }
                                        //세로줄의 가로 확인
                                        //첫번째 위치에 놓일 때 확인
                                        if(y >= 0 && y <= 4 && map[i][y+1]==yourNum &&
map[i][y+2]==yourNum && map[i][y+3]==yourNum){
                                                isGoodPlace= true;
                                        }
                                        //두번째 위치에 놓일 때 확인
                                        else if(y >= 1 && y <= 5 && map[i][y-1]==yourNum
&& map[i][y+1]==yourNum && map[i][y+2]==yourNum){
                                                isGoodPlace= true;
                                        }
                                        //세번째 위치에 놓일 때 확인
                                        else if(y >= 2 && y <= 6 && map[i][y-2]==yourNum
&& map[i][y-1]==yourNum && map[i][y+1]==yourNum){
                                                isGoodPlace= true;
                                        }
                                        //네번째 위치에 놓일 때 확인
                                        else if(y >= 3 && y <= 7 && map[i][y-3]==yourNum
&& map[i][y-2]==yourNum && map[i][y-1]==yourNum){
                                                isGoodPlace= true;
                                        }
                                        //세로줄 확인 끝나면
                                        if(isGoodPlace == false){ //좋은 자리 아니면
                                                xPosition[i] = 4; //GoodPlace아닌 자리
최우선으로 놓아야 할 자리
                                        }
                                }
                                //GoodPlace(근처에 상대방 것이 세개 있는 위치)세로줄 확인 끝

                                //이제 가로선 확인시작!
                                if(yPosition[i] == 3){
```

```
                                                    //상향 대각선
                                                    //첫번째 위치에 놓일 때 확인
                                                    if (x >= 0 && x <= 4 && i >= 3 && i <= 7 && map[x
+ 1][i - 1] == yourNum
                                                                && map[x + 2][i - 2] == yourNum &&
map[x + 3][i - 3] == yourNum) {
                                                        isGoodPlace = true;
                                                    }
                                                    // 두번째 위치에 놓일 때 확인
                                                    else if (x >= 1 && x <= 5 && i >= 2 && i <= 6 &&
map[x - 1][i + 1] == yourNum
                                                                && map[x + 1][i - 1] == yourNum &&
map[x + 2][i - 2] == yourNum) {
                                                        isGoodPlace = true;
                                                    }
                                                    // 세번째 위치에 놓일 때 확인
                                                    else if (x >= 2 && x <= 6 && i >= 1 && i <= 5 &&
map[x - 2][i + 2] == yourNum
                                                                && map[x - 1][i + 1] == yourNum &&
map[x + 1][i - 1] == yourNum) {
                                                        isGoodPlace = true;
                                                    }
                                                    // 네번째 위치에 놓일 때 확인
                                                    else if (x >= 3 && x <= 7 && i >= 0 && i <= 4 &&
map[x - 3][i + 3] == yourNum
                                                                && map[x - 2][i + 2] == yourNum &&
map[x - 1][i + 1] == yourNum) {
                                                        isGoodPlace = true;
                                                    }
                                                    // 가로 줄의 햐향대각선 확인
                                                    // 네번째 위치에 놓일 때 확인
                                                    if (x >= 0 && x <= 4 && i >= 0 && i <= 4 && map[x
+ 1][i + 1] == yourNum
                                                                && map[x + 2][i + 2] == yourNum &&
map[x + 3][i + 3] == yourNum) {
                                                        isGoodPlace = true;
                                                    }
                                                    // 세번째 위치에 놓일 때 확인
                                                    else if (x >= 1 && x <= 5 && i >= 1 && i <= 5 &&
map[x - 1][i - 1] == yourNum
                                                                && map[x + 1][i + 1] == yourNum &&
map[x + 2][i + 2] == yourNum) {
                                                        isGoodPlace = true;
                                                    }
```

```
                                        // 두번째 위치에 놓일 때 확인
                                        else if (x >= 2 && x <= 6 && i >= 2 && i <= 6 &&
map[x - 2][i - 2] == yourNum
                                                && map[x - 1][i - 1] == yourNum &&
map[x + 1][i + 1] == yourNum) {
                                            isGoodPlace = true;
                                        }
                                        // 첫번째 위치에 놓일 때 확인
                                        else if (x >= 3 && x <= 7 && i >= 3 && i <= 7 &&
map[x - 3][i - 3] == yourNum
                                                && map[x - 2][i - 2] == yourNum &&
map[x - 1][i - 1] == yourNum) {
                                            isGoodPlace = true;
                                        }
                                        //가로줄의 세로 확인
                                        //첫번째 위치에 놓일 때 확인
                                        if(x >= 0 && x <= 4 && map[x+1][i]==yourNum &&
map[x+2][i]==yourNum && map[x+3][i]==yourNum){
                                            isGoodPlace= true;
                                        }
                                        //두번째 위치에 놓일 때 확인
                                        else if(x >= 1 && x <= 5 && map[x-1][i]==yourNum
&& map[x+1][i]==yourNum && map[x+2][i]==yourNum){
                                            isGoodPlace= true;
                                        }
                                        //세번째 위치에 놓일 때 확인
                                        else if(x >= 2 && x <= 6 && map[x-2][i]==yourNum
&& map[x-1][i]==yourNum && map[x+1][i]==yourNum){
                                            isGoodPlace= true;
                                        }
                                        //네번째 위치에 놓일 때 확인
                                        else if(x >= 3 && x <= 7 && map[x-3][i]==yourNum
&& map[x-2][i]==yourNum && map[x-1][i]==yourNum){
                                            isGoodPlace= true;
                                        }
                                        //가로줄의 가로 확인
                                        //첫번째 위치에 놓일 때 확인
                                        if(i >= 0 && i <= 4 && map[x][i+1]==yourNum &&
map[x][i+2]==yourNum && map[x][i+3]==yourNum){
                                            isGoodPlace= true;
                                        }
                                        //두번째 위치에 놓일 때 확인
                                        else if(i >= 1 && i <= 5 && map[x][i-1]==yourNum &&
map[x][i+1]==yourNum && map[x][i+2]==yourNum){
```

```
                                        isGoodPlace= true;
                                    }
                                    //세번째 위치에 놓일 때 확인
                                    else if(i >= 2 && i <= 6 && map[x][i-2]==yourNum &&
map[x][i-1]==yourNum && map[x][i+1]==yourNum){
                                        isGoodPlace= true;
                                    }
                                    //네번째 위치에 놓일 때 확인
                                    else if(i >= 3 && i <= 7 && map[x][i-3]==yourNum &&
map[x][i-2]==yourNum && map[x][i-1]==yourNum){//네번째
                                        isGoodPlace= true;
                                    }
                                    //가로줄 확인 끝나면
                                    if(isGoodPlace == false){ //좋은 자리 아니면
                                        yPosition[i] = 4; //GoodPlace아닌 자리
                                    }
                                }
                                //GoodPlace(근처에 상대방 것이 세개 있는 위치) 가로줄 확인 끝
                            }//GoodPlace검색하는 for문 끝



                            //여기 부터는 GoodPlace가 얼마나 있는지 확인하는 구간
                            int find_4 = 0; //GoodPlace가 아닌 곳(=4)이 몇개인지 확이하는
count변수

                            for(int i = 0; i < PixelTester.SIZE_OF_BOARD; i++ ){
                                if(xPosition[i] == 4){
                                    find_4++;
                                }
                                if(yPosition[i] == 4){
                                    find_4++;
                                }
                            }
                            //GoodPlace가 얼마나 있는지 확인하는 구간 끝
                            if(find_4 == 0){//모두 GoodPlace일 때 (=4가 하나도 없을 때)차례로 넣음
                                for(int i = 0; i < PixelTester.SIZE_OF_BOARD ; i ++){
                                    if(xPosition[i] == 3){ //세로부터 남은 곳 확인
                                        nextPosition = new Point(i, y);
                                        return nextPosition;
                                    }
                                    if(yPosition[i] == 3){ //가로 남은 곳 확인해서 입력
                                        nextPosition = new Point(x, i);
                                        return nextPosition;
                                    }
```

```
                        }
                    }
                    else if (find_4 != 0) { // GoodPlace가 아닌 구간 있으면
                        for (int i = 0; i < PixelTester.SIZE_OF_BOARD; i++) {
                            boolean isNotAtrractivePlace = false; // 근처에 내것이
두개 있는 위치
                            // NotAtrractivePlace(근처에 내 것이 두개 있는
위치)확인
                            // 세로선먼저 확인
                            if (xPosition[i] == 4) {
                                // 상향 대각선
                                // 첫번째 위치에 놓일 때 확인
                                if (i >= 0 && i <= 5 && y >= 2 && y <= 7
&& map[i + 1][y - 1] == myNum
                                        && map[i + 2][y - 2] ==
myNum) {
                                    isNotAtrractivePlace = true;
                                }
                                // 두번째 위치에 놓일 때 확인
                                else if (i >= 1 && i <= 6 && y >= 1 && y <=
6 && map[i - 1][y + 1] == myNum
                                        && map[i + 1][y - 1] ==
myNum) {
                                    isNotAtrractivePlace = true;
                                }
                                // 세번째 위치에 놓일 때 확인
                                else if (i >= 2 && i <= 7 && y >= 0 && y <=
5 && map[i - 2][y + 2] == myNum
                                        && map[i - 1][y + 1] ==
myNum) {
                                    isNotAtrractivePlace = true;
                                }
                                // 세로줄의 햐향대각선 확인
                                // 첫번째 위치에 놓일 때 확인
                                if (i >= 2 && i <= 7 && y >= 2 && y <= 7
&& map[i - 2][y - 2] == myNum
                                        && map[i - 1][y - 1] ==
myNum) {
                                    isNotAtrractivePlace = true;
                                }
                                // 두번째 위치에 놓일 때 확인
                                else if (i >= 1 && i <= 6 && y >= 1 && y <=
6 && map[i - 1][y - 1] == myNum
                                        && map[i + 1][y + 1] ==
```

```
myNum) {
                                                    isNotAtrractivePlace = true;
                                                }
                                                // 세번째 위치에 놓일 때 확인
                                                else if (i >= 0 && i <= 5 && y >= 0 && y <=
5 && map[i + 1][y + 1] == myNum
                                                                && map[i + 2][y + 2] ==
myNum) {
                                                    isNotAtrractivePlace = true;
                                                }
                                                // 세로줄의 세로
                                                // 첫번째 위치에 놓일 때 확인
                                                if (i >= 0 && i <= 5 && map[i + 1][y] ==
myNum && map[i + 2][y] == myNum) {
                                                    isNotAtrractivePlace = true;
                                                }
                                                // 두번째 위치에 놓일 때 확인
                                                else if (i >= 1 && i <= 6 && map[i - 1][y] ==
myNum && map[i + 1][y] == myNum) {
                                                    isNotAtrractivePlace = true;
                                                }
                                                // 세번째 위치에 놓일 때 확인
                                                else if (i >= 2 && i <= 7 && map[i - 2][y] ==
myNum && map[i - 1][y] == myNum) {
                                                    isNotAtrractivePlace = true;
                                                }
                                                // 세로줄의 가로 확인
                                                // 세번째 위치에 놓일 때 확인
                                                if (y >= 0 && y <= 5 && map[i][y + 1] ==
myNum && map[i][y + 2] == myNum) {
                                                    isNotAtrractivePlace = true;
                                                }
                                                // 두번째 위치에 놓일 때 확인
                                                else if (y >= 1 && y <= 6 && map[i][y - 1] ==
myNum && map[i][y + 1] == myNum) {
                                                    isNotAtrractivePlace = true;
                                                }
                                                // 첫번째 위치에 놓일 때 확인
                                                else if (y >= 2 && y <= 7 && map[i][y - 2] ==
myNum && map[i][y - 1] == myNum) {
                                                    isNotAtrractivePlace = true;
                                                }
                                                // 세로줄 확인 끝나면
                                                if (isNotAtrractivePlace == false) { // 내거
```

```
두개 있는 자리 아니면
                                                    xPosition[i] = 5; //
NotAtrractivePlace아닌 자리// 최우선으로 놓아야 할 자리
                                                }
                                            } // NotAtrractivePlace(근처에 상대방 것이 세개 있는
위치)세로줄 확인 끝

                                            // 이제 가로선 확인시작!
                                            if (yPosition[i] == 4) {
                                                    // 상향 대각선
                                                    // 첫번째 위치에 놓일 때 확인
                                                    if (x >= 0 && x <= 5 && i >= 2 && i <= 7
&& map[x + 1][i - 1] == myNum
                                                                    && map[x + 2][i - 2] ==
myNum) {
                                                            isNotAtrractivePlace = true;
                                                    }
                                                    // 두번째 위치에 놓일 때 확인
                                                    else if (x >= 1 && x <= 6 && i >= 1 && i <=
6 && map[x - 1][i + 1] == myNum
                                                                    && map[x + 1][i - 1] ==
myNum) {
                                                            isNotAtrractivePlace = true;
                                                    }
                                                    // 세번째 위치에 놓일 때 확인
                                                    else if (x >= 2 && x <= 7 && i >= 0 && i <=
5 && map[x - 2][i + 2] == myNum
                                                                    && map[x - 1][i + 1] ==
myNum) {
                                                            isNotAtrractivePlace = true;
                                                    }
                                                    // 가로줄의 햐향대각선 확인
                                                    // 세번째 위치에 놓일 때 확인
                                                    if (x >= 0 && x <= 5 && i >= 0 && i <= 5
&& map[x + 1][i + 1] == myNum
                                                                    && map[x + 2][i + 2] ==
myNum) {
                                                            isNotAtrractivePlace = true;
                                                    }
                                                    // 두번째 위치에 놓일 때 확인
                                                    else if (x >= 1 && x <= 6 && i >= 1 && i <=
6 && map[x - 1][i - 1] == myNum
                                                                    && map[x + 1][i + 1] ==
myNum) {
```

```
                isNotAtrractivePlace = true;
            }
            // 첫번째 위치에 놓일 때 확인
            else if (x >= 2 && x <= 7 && i >= 2 && i <=
7 && map[x - 2][i - 2] == myNum
                        && map[x - 1][i - 1] ==
myNum) {
                isNotAtrractivePlace = true;
            }
            // 가로줄의 세로 확인
            // 첫번째 위치에 놓일 때 확인
            if (x >= 0 && x <= 5 && map[x + 1][i] ==
myNum && map[x + 2][i] == myNum) {
                isNotAtrractivePlace = true;
            }
            // 두번째 위치에 놓일 때 확인
            else if (x >= 1 && x <= 6 && map[x - 1][i] ==
myNum && map[x + 1][i] == myNum) {
                isNotAtrractivePlace = true;
            }
            // 세번째 위치에 놓일 때 확인
            else if (x >= 2 && x <= 7 && map[x - 2][i] ==
myNum && map[x - 1][i] == myNum) {
                isNotAtrractivePlace = true;
            }
            // 가로줄의 가로 확인
            // 세번째 위치에 놓일 때 확인
            if (i >= 0 && i <= 5 && map[x][i + 1] ==
myNum && map[x][i + 2] == myNum) {
                isNotAtrractivePlace = true;
            }
            // 두번째 위치에 놓일 때 확인
            else if (i >= 1 && i <= 6 && map[x][i - 1] ==
myNum && map[x][i + 1] == myNum) {
                isNotAtrractivePlace = true;
            }
            // 첫번째 위치에 놓일 때 확인
            else if (i >= 2 && i <= 7 && map[x][i - 2] ==
myNum && map[x][i - 1] == myNum) {
                isNotAtrractivePlace = true;
            }
            // 가로줄 확인 끝나면
            if (isNotAtrractivePlace == false) { // 내거
두개 있는 자리 아니면
```

```
                                                yPosition[i] = 5; //
NotAtrractivePlace아닌 자리// 최우선으로 놓아야 할 자리
                                                }
                                        } // NotAtrractivePlace(근처에 내 것이 두개 있는 위치)
가로줄 확인 끝
                                } // NotAtrractivePlace검색하는 for문 끝

                                //NotAtrractivePlace가 아닌 곳(5) 있으면 거기 넣기
                                int key = 0; // 단순히 배열의 인덱스를 나타내는 변수명
                                for(int k = 0; k < PixelTester.SIZE_OF_BOARD; k++ ){
                                        if(xPosition[key] == 5){
                                                nextPosition = new Point(key, y);
                                                return nextPosition;
                                        }
                                        if(yPosition[key] == 5){
                                                nextPosition = new Point(x, key);
                                                return nextPosition;
                                        }
                                        key = key + (7 - k)* (int)Math.pow(-1,k );//빈 곳이
있다면 양쪽 끝부터 돌을 배치
                                }

                                // GoodPlace 없으면 나머지에서 아무거나 넣음
                                int key2 = 0;// 단순히 배열의 인덱스를 나타내는 변수명
                                for(int k = 0; k < PixelTester.SIZE_OF_BOARD; k++ ){
                                        if(xPosition[key2] == 4){
                                                nextPosition = new Point(key2, y);
                                                return nextPosition;
                                        }
                                        if(yPosition[key2] == 4){
                                                nextPosition = new Point(x, key2);
                                                return nextPosition;
                                        }
                                        key2 = key2 + (7 - k)* (int)Math.pow(-1,k );//빈 곳이
있다면 양쪽 끝부터 돌을 배치
                                }
                        }//GoodPlace 아닌 구간 있을 때 확인한 거 끝
                }//BadPlace 아닌 구간 있을 때 확인한 거 끝
        }//둘 수 있는 곳 여러곳남았을 때의 else문 끝
        nextPosition = new Point(x, y);
        return nextPosition;

    }//lastPoint 끝
}
```

3. 결론

1) 고찰

김선주 :

리그전에 나가게 될 게임 알고리즘을 개발해보는 것은 처음이라서 많이 걱정도 되고 어렵기도 했다. 여러 가지 방법을 생각해보고, 직접 종이에 변형된 4목 게임을 해보면서 어떻게 코드를 만들어야 할지 고민했다. 코드를 만들면서 자바 문법에 대해서 자연스럽게 공부하게 되는 것은 물론이고, 수업시간에 배웠던 greedy 알고리즘이나 branch and bound 알고리즘 등에 대해서도 복습을 할 수 있었다. 책에 나와 있는 예제를 연습하는 것이 아니라 직접 게임에 쓰이게 될 코드를 만들어보았기 때문에 학습 효과가 더 컸다. 리그전에서 어떤 결과가 나올지 매우 궁금하다. 15일에 있을 텀 프로젝트 발표에서 다른 조들의 알고리즘에 대해서도 알아보고 싶고, 어떤 방법으로 나와 다른 생각을 했는지도 궁금하다.

노주영 :

처음에는 막막하기만 했었는데 과제와 함께 예시로 나온 100, 101번의 코드 분석을 가지고 어떻게 표현해야 할지를 알아내었고, 나머지 전략적인 부분은 직접 종이에 게임을 해보면서 팀 멤버와 함께 분석해 나가는 것이 생각보다 재밌었다. 밤을 새워가며 서로 코드를 보고 개선해야 할 점이나 잘한 점을 배워 좀 더 나은 코드를 만들어 내는 것이 힘들었지만 즐거웠다. 학교 다니면서 가장 열심히 짠 코드였기 때문에 아직 결과를 모르는 지금은 뿌듯하다.