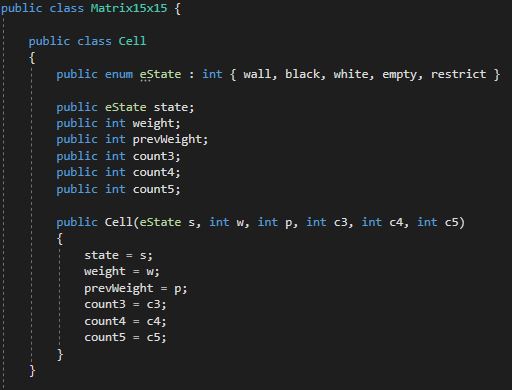
|  |  |
| --- | --- |
| **HOmok** | |
| C:\Users\asdjk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\2.png C:\Users\asdjk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\1.png | |
|  |  |
| **개발 기간** | 2015.04 ~ 2015.06 |
| **개발 인원** | 황인원 |
| **개발 스펙** | Unity [ C# ] |
| **개발 내역** | KMP 알고리즘을 이용한 렌주룰 구현  알파 베타 가지치기를 이용한 컴퓨터 대전 모드 구현 |
| **요 약** | 인공지능 수업 때 기말과제로 진행했던 오목 게임입니다. |
| **소스 코드** | <https://github.com/InwonHwang/HOmok> |
|  |  |
|  |  |

**표현**

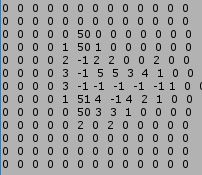
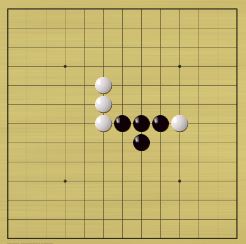
바둑판을 본떠서 2차원 배열을 갖고있는 Matrix15x15 클래스를 만들었습니다. 각 배열의 자료형은 Cell 클래스입니다. Cell 클래스 내부에는 가중치와 해당 셀에 돌을 두었을 때 3의 개수와 4의 개수 5가되는 개수를 저장합니다. 3이되는 위치와 4가되는 위치의 셀은 높은 가중치를 부여 받습니다.



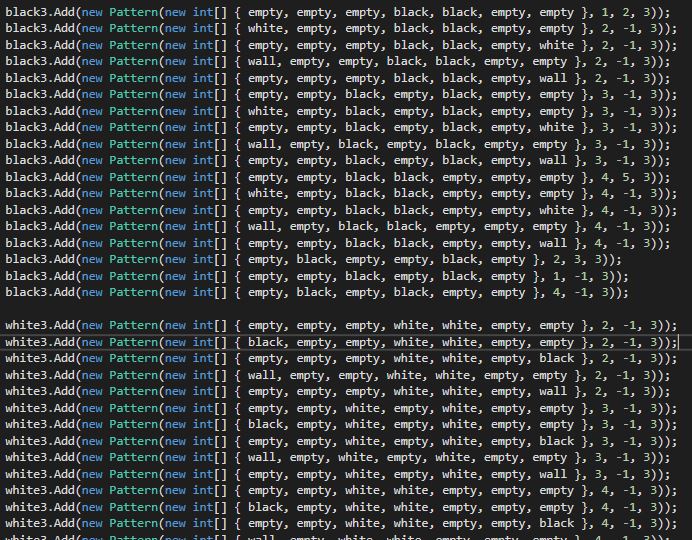
<Cell 클래스 일부>

**쌍삼와 쌍사**

흑의 유리함을 제한하는 렌주룰을 적용하였습니다. 렌주룰은 흑의 쌍삼과 쌍사가 되는 지역에 흑돌을 두는 것을 방지하는 것입니다. Matrix15x15을 최신화 시킬 때 패턴매칭 알고리즘을 사용하였습니다. 패턴의 모든경우를 고려하여 미리 만들어 두었습니다. Cell의 count3값이 2 이상이면 쌍삼이 되고 count4값이 2이상일 때 쌍사가 됩니다. 쌍삼과 쌍사가 되는 Cell의 상태는 restrict상태로 수를 놓는 것이 제한됩니다.



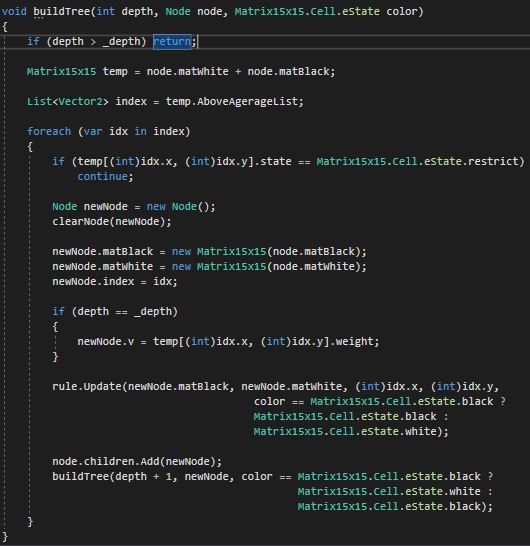
<가중치 표현>



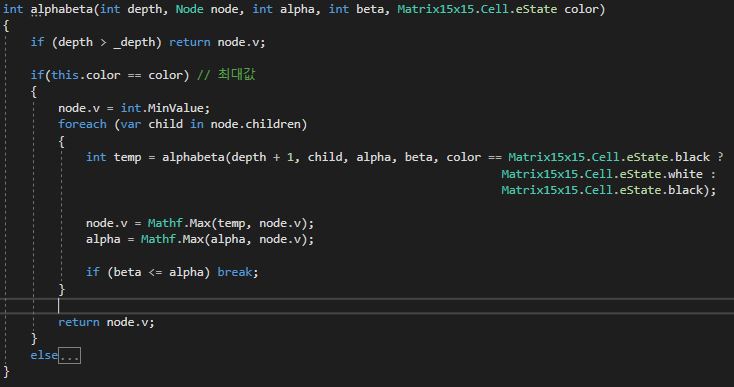
<패턴 일부>

**Alpha Beta Pruning**

컴퓨터와 대전모드를 구현하기위해 사용했던 알고리즘입니다. MinMax 알고리즘은 직접 구현하였고 alpha beta pruning은 wiki를 참고하였습니다. 기본알고리즘은 각 발생할 수 있는 상황별로 트리를 만들고 트리에서 자신에게 가장 유리한 상황이 되는 곳에 돌을 두는 것입니다. 몇수까지 예상 할지 depth를 지정하여 트리를 생성합니다.



<트리>



<alphabeta pruning>

**문제점**

트리를 만들 때 모든 경우의 수를 고려하면 트리의 가지 수가 수 없이 많이 늘어나는 것을 알게 되었습니다. 가중치가 높은 값에만 수를 두는 것을 알고 있기 때문에 가중치가 일정 값 이상인 경우만 트리에 추가하였습니다.

