[기초공학설계-C]

오목 프로젝트 보고서

과목명: 기초공학설계 01반

담당 교수님: 장형수 교수님

학번: 20181679

이름: 장서우

1. 주제

오목게임(omok)

1. 프로젝트 목적 및 설명

**<프로젝트 목적>**

-기초공학설계 과목에서 학습한 c언어의 변수, 반복문, 조건문 등의 문법적 요소들과 함수를 사용하여 프로그램을 모듈화하는 법을 체득한다.

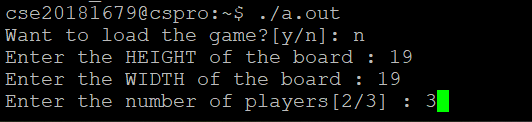
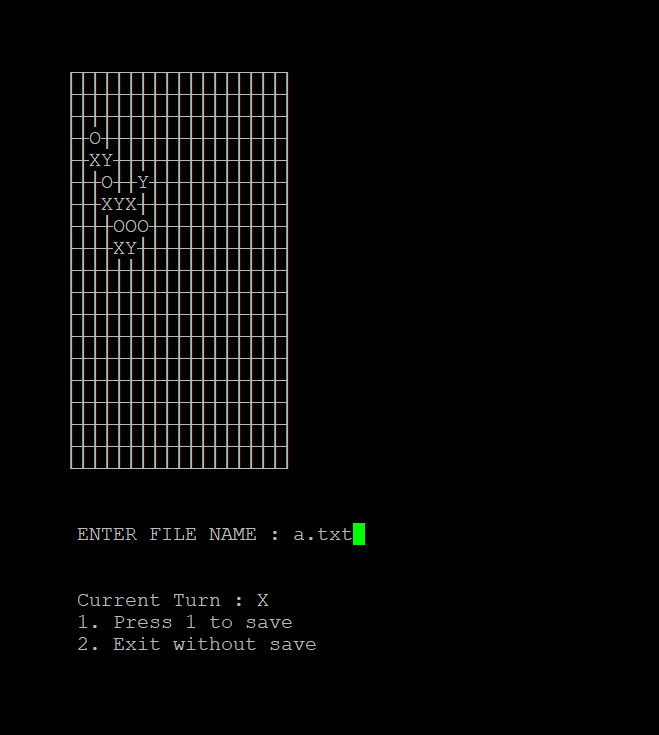
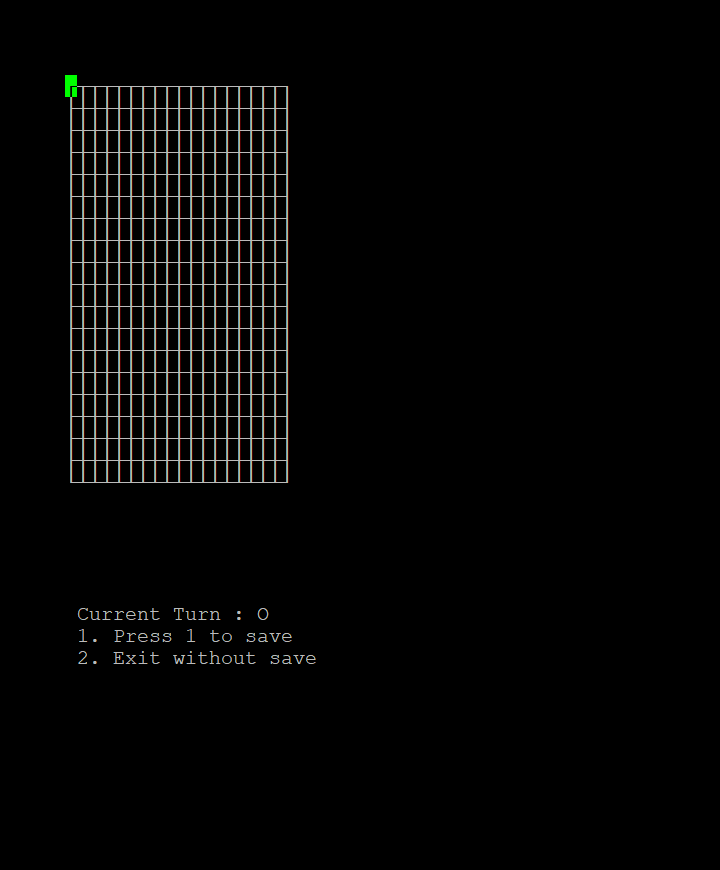
-UNIX기반 library인 ’ncurses.h’를 사용하기 위해 documentary를 찾아보고 이해하여 프로그램에 적용한다.

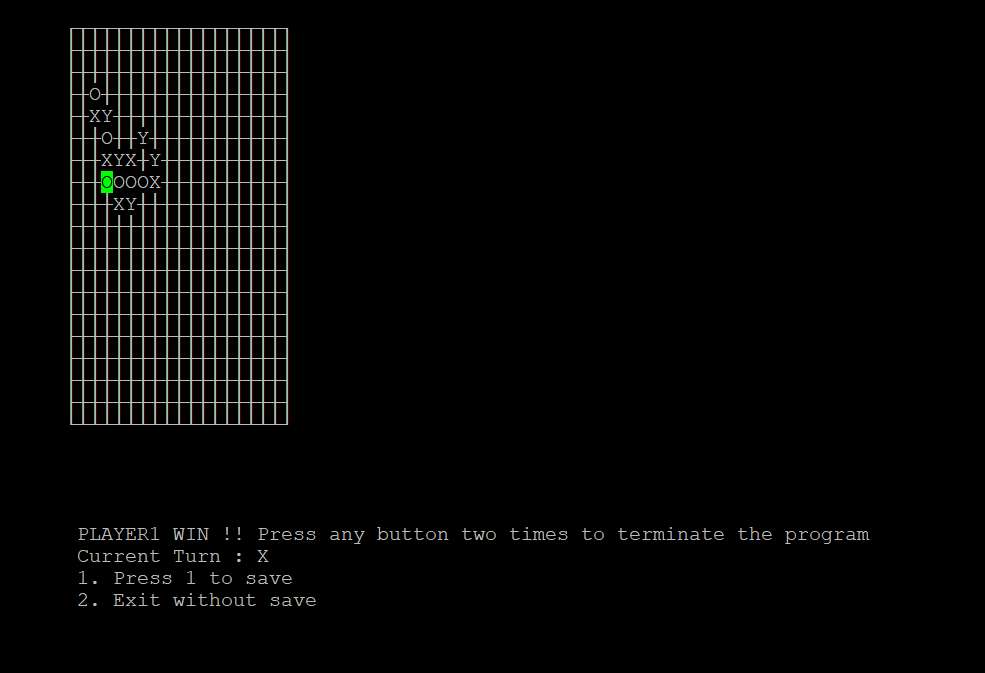
**<게임설명>**

-오목은 상대방보다 먼저 대각선, 가로, 세로 방향 중 하나로 5개의 연속된 돌을 놓으면 승리하는 게임이다.

-4목일 경우 4개의 연속된 돌을 놓으면 승리.

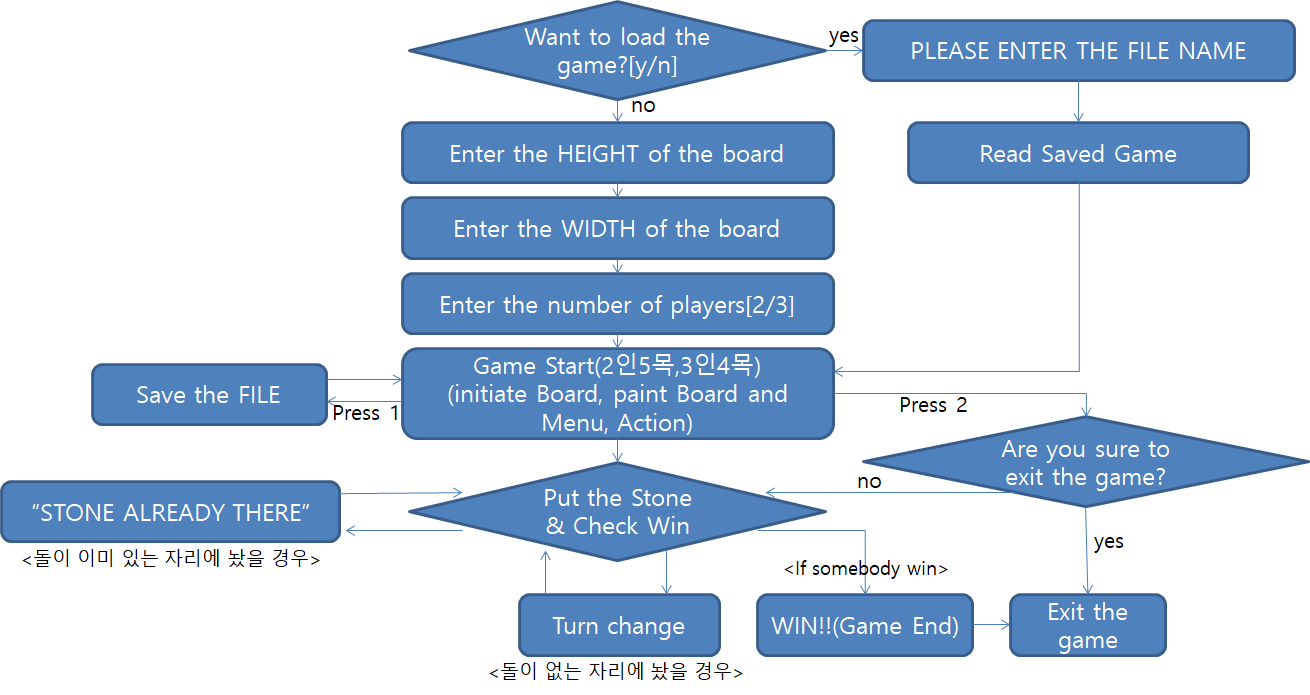
**<프로그램 동작 결과>**

**  **

****

<왼쪽 위부터 그림1,2,3,4,5. 프로그램 동작 결과>

1. 프로그램 실행흐름도(Flowchart)



<그림6. 실행흐름도>

1. 저장된 게임을 불러오려면 y를 누르고 파일이름을 입력한 후 불러온 게임으로 게임을 다시 진행한다.

2. 새 게임을 하려면 n을 누르고 오목판의 세로길이와 가로길이를 설정해준 후 플레이어 수를 입력해준다. (이때 플레이어 수가 2명이면 오목, 3명이면 4목이 진행된다.)

3. 각 사용자가 돌을 번갈아 가며 둔다. (※돌이 있는 위치에는 돌을 둘 수 없다.)

4. 상대방보다 먼저 대각선, 가로, 세로 방향 중 하나로 5개의 연속된 돌을 놓으면 승리한다. (4목일 경우 4개의 연속된 돌을 놓으면 승리.)

5. 게임 도중 1을 누르면 중간에 파일저장이 가능하다.

6. 게임 도중 2를 누르면 게임을 저장하지 않고 나갈 수 있다.

7. 누군가 한 명이 이길 경우 1,2 키는 적용되지 않으며 아무 키나 '두 번' 누르면 빠져나간다.

1. 프로그램의 각 함수 설명
   1. int main()
      1. 게임 시작 시 실행되며, 저장된 게임을 불러오려면 y를 누르고 파일이름을 입력하고 저장된 게임이 실행되도록 한다.
      2. 새로운 게임을 하려면 오목판의 크기를 입력 받고 플레이어 수를 입력 받는다.
      3. 새로운 윈도우’WINDOW \*win=newwin()’를 불러올 뿐만 아니라, 터미널을 원상 복귀 시키기 위한 함수’endwin()’를 호출하여 종료시킨다.
   2. int\*\* initBoard(int \*\*board, int \*row, int \*col, int \*turn, int load)
      1. 오목판을 초기화 시켜준다.
   3. int gameStart(WINDOW \*win, int load, int players)
      1. 오목판을 초기화 시키는 함수를 호출한다.
      2. 게임 진행에 요구되는 사항들(오목판 초기화, 오목판 그리기, 메뉴 그리기, 입력 받기, 입력에 대한 적절한 행동하기)을 순서대로 실행한다.
      3. 사용자의 입력(keyin)이 2일 경우, 게임을 종료해준다. 이때 게임을 종료할건지 한 번 더 질문하여 y/n로 게임을 종료하거나 continue해준다.
   4. Int paintBoard(int \*\*board, WINDOW \*win, int row, int col)
      1. 오목판과 올려진 돌을 화면에 출력시키는 역할을 한다.
   5. void paintMenu(int players, int turn, char cturn, WINDOW \*win, int row, int col)
      1. menu를 화면에 출력시키는 역할을 한다.
      2. menu에 current turn을 나타내는 구간이 있으므로 turn과 cturn을 이용하여 current turn을 계속 반복하여 바꿔준다.
      3. 2인 오목인 경우 player1의 돌은 ‘O’, player2의 돌은 ‘X’로 표시되도록 하며 turn은 1->2->1->2->1->…순으로 바꿔준다.
      4. 3인 4목인 경우 player1의 돌은 ‘O’, player2의 돌은 ‘X’, player3의 돌은 ‘Y’로 표시되도록 하며 turn은 1->2->3->1->2->3->1->…순으로 바꿔준다.
   6. int Action(WINDOW \*win, int \*\*board, int keyin, int \*row, int \*col, int \*turn, int players)
      1. 사용자의 입력(keyin)이 방향키일 경우 방향키의 방향대로 커서가 이동하며 오목판을 벗어날 수는 없다.
      2. 사용자의 입력(keyin)이 Enter키 또는 Space키 일 경우 바둑돌을 둘 수 있으며, 이미 돌이 놓여진 자리에는 돌을 놓을 수 없도록 하였다.
      3. 사용자의 입력(keyin)이 1일 경우, 게임을 저장해준다.
      4. 사용자의 입력(keyin)이 2일 경우, 2를 return 해준다.->gameStart함수로 이어짐
      5. 한 사용자가 이겨서 게임이 끝날 경우 1을 return 하고 이외엔 0을 return 한다. (사용자의 입력(keyin)이 Enter키 또는 Space키일 경우마다(돌을 놓을 때마다) checkWin함수가 돌아가도록 해준다.)
   7. int checkWin(int \*\*board, int turn, int players)
      1. players가 2일 경우와 3일 경우를 나누어서 게임이 끝났는지를 체크한다.
      2. players가 2일 경우는 가로, 세로, 대각선으로 돌이 연속해서 5개가 있는 지를 체크하고 어느 플레이어가 이겼는지 화면에 나타내준다.
      3. players가 3일 경우는 가로, 세로, 대각선으로 돌이 연속해서 4개가 있는 지를 체크하고 어느 플레이어가 이겼는지 화면에 나타내준다.
      4. 한 사용자가 이겨서 게임이 끝날 경우 1을 return하고 이외엔 0을 return해준다.
   8. void saveGame(int \*\*board, int players, int row, int col, int turn, char \*filename)
2. 게임을 입력 받은 파일에 저장한다.
3. 플레이어 수, 커서의 위치, 차례, 오목판의 크기를 저장해준다.
   1. int\*\* readSavedGame(int \*\*board, int \*players, int \*row, int \*col, int \*turn, char \*filename) / void readSavedHW(char \*filename)
4. 저장했던 게임의 파일 이름을 읽어 들여서 board를 return 하여 돌의 위치를 저장했던 그대로 불러올 수 있도록 한다.
5. 참고자료

-http://www.cs.ukzn.ac.za/~hughm/os/notes/ncurses.html