CSED101. Programming & Problem solving
Fall, 2019

Programming Assignment #3
(70 points)

김태훈(taehoon.kim@postech.ac.kr)

■ ***Due***: **2019.11.17 23:59**

■ ***Development Environment***: **GNU C Compiler (GCC) and any Editor (Vi, or etc.)**

■ ***제출물***

- C Code file (assn3.c)
  - 프로그램의 소스 코드를 이해하기 쉽도록 반드시 **주석**을 붙일 것.
- **보고서 파일** (.docx, .hwp or .pdf) 예) assn3.docx, assn3.hwp 또는 assn3.pdf
  - AssnReadMe.pdf 를 참조하여 작성할 것.
  - **리눅스 서버에 접속하는 것부터 시작해서 프로그램 컴파일 및 실행하는 과정까지를 화면 캡쳐하여 보고서에 포함시키고 간단히 설명 할 것!!**
  - 명예서약(Honor code): 표지에 다음의 내용을 포함한다. "나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다." 보고서 표지에 명예서약이 없는 경우는 과제를 제출하지 않은 것으로 처리한다.
  - 소스코드와 보고서 파일을 LMS를 이용하여 제출한다.

■ ***주의사항***

- 각 문제에 해당하는 요구사항을 반드시 지킬 것.
- 모든 문제의 출력 형식은 아래의 예시들과 동일해야 하며, 같지 않을 시는 감점이 된다.
- 컴파일 & 실행이 안되면 무조건 0점 처리된다.
- **하루 late시 20%가 감점되며, 3일 이상 지나면 받지 않는다. (0점 처리)**
- 부정행위에 관한 규정은 POSTECH 전자컴퓨터공학부 학부위원회의 'POSTECH 전자컴퓨터공학부 부정행위 정의'를 따른다. (LMS의 과목 공지사항의 제목 [document about cheating]의 첨부파일인 disciplinary.pdf를 참조할 것.)
- 과제에서 제안하는 추가 기능을 구현하는 경우 추가 점수가 있습니다.

■ Problem : 'Dots and boxes' game

(목적)

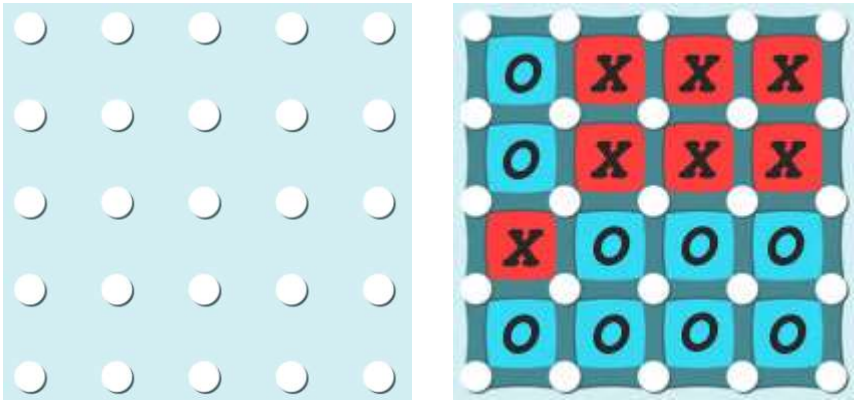
- 2차원 배열의 선언과 사용을 익힌다.
- 함수에서 2차원 배열을 매개변수로 사용하는 방법을 익힌다.
- 텍스트 파일 출력을 익힌다.

(주의사항)

- 전역변수, goto문, 구조체는 사용하지 않는다.
- 프로그램 구현 시, main() 함수를 호출하여 사용하지 않는다. 즉, 소스 코드 내에 main(); 이라고 호출하지 않는다.
- 적절히 사용자 정의 함수를 작성하여야 한다. (main() 함수 내에 모든 기능을 구현 시 감점)
- 과제에서 요구하는 사용자 정의 함수는 반드시 구현하여야 하며, 이외 필요한 함수는 적절히 정의해서 사용한다.
- 문제의 출력 형식은 채점을 위해 아래의 실행 예시와 최대한 비슷하게 작성해 주세요

(문제)

이번 과제에서는 4X4 크기의 'Dots and boxes' 게임을 구현합니다. 'Dots and boxes' 게임은 점으로 이루어진 게임 판에서 누가 더 많은 박스를 만드는지 겨루는 게임입니다.



4X4 Dots and boxes Game

(게임 규칙)

- 게임 판은 점 스물다섯 개로 구성되어 있으며, 가로로 다섯 개, 세로로 다섯 개 크기로 배열되어 있습니다.
- 총 플레이어의 수는 2명이며, 각 플레이어가 번갈아 가로 또는 세로로 가까운 두 점을 잇는 선을 긋게 됩니다. 대각선 또는 멀리 떨어진 두 점을 연결할 수 없습니다.
- 플레이어가 선을 그어 닫힌 사각형을 만들면 1점을 득점합니다. 플레이어가 득점하면, 턴을 넘기지 않고 새로운 선을 그을 수 있습니다.
  (가로 및 세로의 길이가 1이라고 한다면, 넓이가 1인 사각형을 만들었을 때만 득점)
- 한번 그었던 선은 다시 그을 수 없으며, 한번 그은 선은 어떤 플레이어의 소유도 아닙니다.
- 본 게임 구현은 1인용(AI와 대결), 2인용, 그리고 AI 간 대결을 지원합니다.
- 게임이 진행될 때 게임의 진행과정을 파일로 출력하여 기록합니다.
- 예외 처리(Exception Handling): 아래 게임 설명에 명시된 예외 상황이 아닌 다른 예외 상황의 경우 채점 대상이 아닙니다. C에서는 assert() 함수나 조건문를 사용해서 예외를 처리할

수 있지만 이번 과제에서는 조건문만 활용하여 예외 처리 해주십시오.

- 게임 규칙이 이해가 잘 가지 않는다면, 직접 게임을 해보면서 룰을 익히실 수도 있습니다. (참고: https://www.coolmathgames.com/0-dots-and-boxes) 설명한 규칙과 위 주소의 게임 규칙이 일치하지 않을 때는 조교에게 문의해주시되, 이 문서에 우선하여 작성해주세요.

(1) 메뉴 화면

- 프로그램을 실행하면 아래와 같은 메뉴 화면이 출력된 후, 그 아래에 메뉴 입력을 대기한다. 이 상태에서 사용자는 1과 4 사이의 숫자를 입력해서 원하는 기능을 실행한다.

```
Welcome to Dots and boxes!
======================
1. Game: Player vs Player
2. Game: Player vs AI
3. Game: AI vs AI
4. Quit
======================
Select menu number:
```

메인 메뉴 화면

- 예외처리)
  메뉴에서 1~4 외의 숫자(정수)를 입력하면 올바른 입력이 아니라는 메시지와 함께 다시 메뉴를 출력 후, 입력을 받는다.

(2) PLAYER vs PLAYER

- 메뉴에서 1을 입력하여 선택하면, 아래와 같이 2명의 플레이어를 위한 게임 화면이 출력된다 (사용자 입력은 빨간색 밑줄 표시).
  상단에는 현재까지 플레이어들이 획득한 점수가, 그 아래에는 게임판이 표시된다. 게임판의 점은 '+'기호로 표시한다. 플레이어 간 대결에서는 누가 먼저 턴을 시작할지 결정하지 않고 플레이어 1이 먼저 선을 그을 자리의 좌표를 입력한다.

```
Welcome to Dots and boxes!
======================
1. Game: Player vs Player
2. Game: Player vs AI
3. Game: AI vs AI
4. Quit
======================
Select menu number: 1

======================
     0       vs       0
======================
  0  1  2  3  4  5  6  7  8
0 +     +     +     +     +
1
2 +     +     +     +     +
3
4 +     +     +     +     +
5
6 +     +     +     +     +
7
8 +     +     +     +     +
======================
Turn: Player 1
Select the position you want to draw.
```

플레이어들이 현재까지 획득한 점수
플레이어 1 vs 플레이어 2

게임 시작 화면

- 아래는 플레이어 1이 0, 1 좌표를 입력한 후 출력된 화면으로 0, 1 자리에 '---'이 출력됨을 볼 수 있다. 각 플레이어의 턴은 점수가 나지 않는 이상 기본적으로 번갈아 가면서 진행되므로, 다음으로 플레이어 2의 입력을 위해 대기한다.

```
Turn: Player 1
Select the position you want to draw.
0 1

======================
     0       vs       0
======================
  0  1  2  3  4  5  6  7  8
0 + --- +     +     +     +
1
2 +     +     +     +     +
3
4 +     +     +     +     +
5
6 +     +     +     +     +
7
8 +     +     +     +     +
======================
Turn: Player 2
Select the position you want to draw.
```

- 아래는 플레이어 1이 자신의 턴에서 닫힌 사각형을 만들어 1점을 획득한 화면으로, 만든 사각형 안에 대문자 'O'로 표시한다. (플레이어 2는 'X'로 표시)
- 현재, 플레이어 1이 득점을 했으므로 한번 더 선을 그을 입력을 하게 된다.
- 게임판에서 세로를 선택한 경우 출력은 'I' 로 표시한다. (OR 연산자(||) 입력시 사용하는 키)

```
Turn: Player 1
Select the position you want to draw.
2 3

======================
     1       vs       0
======================
  0  1  2  3  4  5  6  7  8
0 + --- + --- +     +     +
1       |  O  |
2 +     + --- +     +     +
3
4 +     +     +     +     +
5
6 +     +     +     +     +
7
8 +     +     +     +     +
======================
Turn: Player 1
Select the position you want to draw.
```

- 예외처리)
  유효하지 않은 좌표(범위를 벗어난 입력, 점의 위치, 이미 그은 선)를 입력하면 아래 예시처럼 올바른 입력이 아니라는 메시지를 출력하고 다시 입력을 받는다. 아래 예시는 유효하지 않은 좌표로 이미 그은 선의 좌표 2, 3을 선택하여 입력한 경우이다.

```
Turn: Player 1
Select the position you want to draw.
2 3
Impossible: Wrong position (Dot or box). Try again.
Select the position you want to draw.
```

- **게임 종료**)
  게임은 모든 라인이 그어졌을 때 종료된다. 게임이 종료될 때에는 아래의 예시처럼, 양 플레이어 또는 AI의 최종 점수 및 게임판을 보여주고 승자를 표시하는 메시지를 출력한다. 아래 화면에서 엔터를 입력하면 초기 화면 메뉴로 돌아간다. (엔터 외의 입력은 고려할 필요 없음.)

```
Turn: Player 1
Select the position you want to draw.
2 1

======================
    10       vs        6
======================
  0  1  2  3  4  5  6  7  8
0 + --- + --- + --- + --- +
1 |  X  |  O  |  O  |  X  |
2 + --- + --- + --- + --- +
3 |  O  |  O  |  O  |  O  |
4 + --- + --- + --- + --- +
5 |  O  |  O  |  O  |  O  |
6 + --- + --- + --- + --- +
7 |  X  |  X  |  X  |  X  |
8 + --- + --- + --- + --- +
======================
Player 1 Wins!

Press Enter key to return to the Main Menu.
```

게임 종료 예시 (플레이어 1 승리)

```
Turn: Player 1
Select the position you want to draw.
2 7

======================
     8       vs        8
======================
  0  1  2  3  4  5  6  7  8
0 + --- + --- + --- + --- +
1 |  X  |  O  |  O  |  O  |
2 + --- + --- + --- + --- +
3 |  X  |  O  |  O  |  O  |
4 + --- + --- + --- + --- +
5 |  X  |  O  |  X  |  X  |
6 + --- + --- + --- + --- +
7 |  X  |  O  |  X  |  X  |
8 + --- + --- + --- + --- +
======================
DRAW!

Press Enter key to return to the Main Menu.
```

게임 종료 예시 (비긴 경우)

(3) PLAYER vs AI

- 메뉴에서 2를 선택하면, 아래와 같이 AI의 난이도를 선택할 수 있다.

```
Welcome to Dots and boxes!
======================
1. Game: Player vs Player
2. Game: Player vs AI
3. Game: AI vs AI
4. Quit
======================
Select menu number: 2

Which difficulty do you want?
======================
1. Easy
2. Normal
3. Hard
4. Quit
======================
Select menu number:
```

AI 난이도 선택 예시

- 혼자서도 게임을 즐길 수 있도록 간단한 AI를 구현하도록 한다.
  - ➢ Easy AI: 컴퓨터는 가능한 무작위 자리에 라인을 긋는다.
  - ➢ Normal AI: 컴퓨터는 사각형을 만들어 점수를 낼 수 있는 라인을 우선적으로 긋는다. (한 개의 선만 더 그으면 사각형이 되는 자리) 그런 자리가 없다면, 무작위 자리에 라인을 긋는다.
    ※ 프로그램 **실행할 때마다 무작위**로 라인을 선택하도록 작성할 것.
  - ➢ **(추가기능)** Hard AI: Normal AI보다 더 좋은 AI를 구현해 주세요!
    추가기능 구현 시 주석에 명시해주시고 보고서에 자세한 사항을 기록해 주세요.

- 예외처리) 난이도 선택 메뉴에서 1~4 외의 숫자(정수)를 입력하면 올바른 입력이 아니라는 메시지와 함께 다시 메뉴를 출력 후, 입력을 받는다. Hard를 구현하지 않은 경우에 3을 선택하면, 구현하지 않았다는 적절한 메시지 출력과 함께 메뉴 출력 후 다시 입력을 받는다.

- 난이도 선택 메뉴에서 4를 입력하여 Quit을 선택하면, 메인 메뉴로 돌아가도록 구현한다.

- 난이도 선택 후, 플레이어와 AI 대결에서는 플레이어가 먼저 선을 긋는 것으로 게임을 시작하도록 한다. 플레이어와 AI 대결에서는 플레이어의 박스는 O로, AI의 박스는 X로 표시한다.

```
1. Easy
2. Normal
3. Hard
4. Quit
======================
Select menu number: 2

======================
     0       vs        0
======================
  0  1  2  3  4  5  6  7  8
0 +     +     +     +     +
1
2 +     +     +     +     +
3
4 +     +     +     +     +
5
6 +     +     +     +     +
7
8 +     +     +     +     +
======================
Turn: Player
Select the position you want to draw.
```

Player vs AI

게임 시작 화면 (PLAYER vs AI)

- 플레이어가 5, 0 좌표를 입력하여 선을 그은 화면이 출력된 후 1초 뒤에 컴퓨터가 선택한 좌표와 함께 게임판이 출력되도록 한다.
  1초 지연 효과는 *Tips-지연 효과 주기(마지막 쪽)*을 참고하여 구현한다.

```
Turn: Player
Select the position you want to draw.
5 0

======================
     0       vs       0
======================
  0  1  2  3  4  5  6  7  8
0 +     +     +     +     +
1
2 +     +     +     +     +
3
4 +     +     +     +     +
5 ¦
6 +     +     +     +     +
7
8 +     +     +     +     +
======================
Turn: AI
The selected position by AI is 6 3

======================
     0       vs       0
======================
  0  1  2  3  4  5  6  7  8
0 +     +     +     +     +
1
2 +     +     +     +     +
3
4 +     +     +     +     +
5 ¦
6 +     + --- +     +     +
7
8 +     +     +     +     +
======================
Turn: Player
Select the position you want to draw.
```

게임 진행 화면 (PLAYER vs AI)

- 다음은 플레이어가 6점, AI가 10점을 획득하여 AI가 승리한 예시 화면으로 엔터를 입력하면 메인 메뉴로 돌아간다.

```
Turn: AI
The selected position by AI is 3 0

======================
     6       vs       10
======================
  0  1  2  3  4  5  6  7  8
0 + --- + --- + --- + --- +
1 ¦  X  ¦  X  ¦  X  ¦  X  ¦
2 + --- + --- + --- + --- +
3 ¦  X  ¦  X  ¦  X  ¦  X  ¦
4 + --- + --- + --- + --- +
5 ¦  X  ¦  O  ¦  O  ¦  O  ¦
6 + --- + --- + --- + --- +
7 ¦  X  ¦  O  ¦  O  ¦  O  ¦
8 + --- + --- + --- + --- +
======================
AI Wins!

Press Enter key to return to the Main Menu.
```

게임 종료 예시 (AI 승리)

(4) AI vs AI

- 메뉴에서 3을 선택하면, 아래와 같이 AI의 난이도를 2차례에 걸쳐 선택하여 AI간 대결을 하도록 한다. AI의 난이도는 (3)에서 구현한 AI들을 사용한다.
- AI 대결에서는 AI 1이 먼저 선을 긋는 것으로 게임을 시작하도록 하며 AI 1의 박스는 O로, AI 2의 박스는 X로 표시한다.
- 플레이어의 입력이 없으므로, AI간의 대결은 사용자가 진행상황을 확인하기 편리하도록 sleep 함수를 사용하여 0.5초의 간격으로 출력되도록 한다. (*Tips-지연 효과 주기(마지막 쪽) 참고*)

```
Which difficulty do you want for AI 1?
======================
1. Easy
2. Normal
3. Hard
4. Quit
======================
Select menu number: 1

Which difficulty do you want for AI 2?
======================
1. Easy
2. Normal
3. Hard
4. Quit
======================
Select menu number: 1

======================
     0       vs        0
======================
  0  1  2  3  4  5  6  7  8
0 +     +     +     +     +
1
2 +     +     +     +     +
3
4 +     +     +     +     +
5
6 +     +     +     +     +
7
8 +     +     +     +     +
======================
Turn: AI 1
The selected position by AI 1 is 1 2
```

AI 1 vs AI 2

- 난이도 선택에서, 4를 입력하여 Quit을 선택하면 메인 메뉴로 돌아가도록 구현한다.
- 예외처리) 난이도 선택 메뉴에서 1~4 외의 숫자(정수)를 입력하면 올바른 입력이 아니라는 메시지와 함께 다시 메뉴를 출력 후, 입력을 받는다. 구현하지 않은 난이도를 선택하면, 구현하지 않았다는 적절한 메시지 출력과 함께 메뉴 출력 후 다시 입력을 받는다.

(5) 게임 종료

- 메인 메뉴에서 4를 선택하면 아래 예시처럼 종료 메시지와 함께 프로그램을 종료한다.

```
Welcome to Dots and boxes!
======================
1. Game: Player vs Player
2. Game: Player vs AI
3. Game: AI vs AI
4. Quit
======================
Select menu number: 4
Thank you for playing Dots and Boxes!
[gth1919@programming2 assn3]$
```

프로그램 종료 화면

(6) 게임 기록

- 게임의 기보를 파일에 기록한다. 즉 한번의 게임 진행 시, 그은 선의 위치를 순서대로 기록한다. (단, 유효한 선의 위치만 기록할 것)
- 게임의 기록 형식은 아래와 같다.

```
1 6
7 6
0 1

(중략)

8 3
```

게임 기록 예시

- 저장할 파일 이름은 게임의 종류(1인, 2인, AI대결)에 따라 달라지며, 다음과 같다.
  - `Player vs Player : PvP.txt`
  - `Player vs AI : PvA.txt`
  - `AI vs AI : AvA.txt`
- 종류가 같은 게임을 2번 이상한 경우에는 마지막 게임의 진행과정이 기록으로 남아 있으면 된다.

(코드 작성)

반드시 작성해야 하는 함수는 다음과 같습니다. 이 함수들 말고도 자유롭게 추가로 함수를 정의해서 사용하도록 하세요. 아래 명시된 함수 이름은 바꾸지 말아주세요. 함수 이름과 변수 이름의 대소문자나 철자에 유의해주시기 바라겠습니다. 적절한 주석은 필수입니다. (매개변수는 자유롭게 구성하여 구현해주세요.)

■ int main()

게임판을 나타내는 적절한 2차원 배열을 main 함수에서 선언 후 사용해야 합니다.

■ void init_board(…)

전달 받은 게임판(2차원 배열)을 게임 시작을 위하여 적절한 값으로 채워 넣는 역할을 합니다.

■ 콘솔 출력 관련: void view_board(…)

게임이 진행 중일 때 게임판을 콘솔에 출력하는 역할을 합니다. 해당 함수는 2차원 배열(게임판)을 매개변수로 포함하고 있어야 합니다.

■ Turn 관련

게임의 Turn과 관련하여 적절한 이름의 사용자 정의 함수를 정의하여 구현해야 합니다. Turn 관련 사용자 함수는 주석을 통해서 명시해주시고, 정의한 함수가 어떤 기능을 하는지 적어주십시오.

■ 게임 AI 관련: void AI_function(int *px, int *py, …)

이 함수는 게임의 AI를 구현하는 부분으로 레벨(Easy, Normal, Hard)에 따라 적절하게 좌표를 선택한 후, 포인터 변수 px, py를 통하여 전달합니다. 해당 함수는 2차원 배열을 매개변수로 포함하고 있어야 합니다.

■ 파일 출력 관련: void print_record(…)

진행되는 게임의 기보를 파일에 기록합니다.

Tips.

■ 2차원 배열 사용

- 배열의 크기를 선언할 때 변수를 사용하면 안 됩니다.
- 2차원 배열을 함수의 매개변수로 넘겨주기 위해서는 ary[][3]과 같은 표현을 사용합니다.

```
#define MAXROW 3
#define MAXCOLUMN 3

int ArrayTestFunc(int ary[][MAXCOLUMN], int testvalue);
int main()
{
    int ary[MAXROW][MAXCOLUMN] = {0, 1, 2, 3, 4, 5, 6, 7, 8};
    ArrayTestFunc(ary, MAXROW);
    return 0;
}
```

■ 지연 효과 주기

지연 효과는 unistd.h 헤더 파일을 포함 시킨 후 sleep()함수를 통해 구현 가능합니다. 해당 함수는 인자로 지연시간을 second 단위로 받으며 사용 예는 다음과 같습니다.

```
#include <stdio.h>
#include <unistd.h>
int main()
{
    printf("첫 번째 메시지 입니다\n");
    sleep(1);
    printf("이 메시지는 1초뒤에 출력됩니다\n");
    sleep(0.5);
    printf("이 메시지는 0.5초뒤에 출력됩니다\n");
    return 0;
}
```

■ 어싸인 팁

1. 프로그램 작성 중 무한 루프에 의해 프로그램이 끝나지 않을 경우, Ctrl+C를 누르세요.

2. 이번 과제의 개발 환경은 리눅스입니다. 제출 전, 반드시 Linux 에서 제대로 컴파일 되고 동작되는지 확인하십시오. Linux 환경에서 실행되지 않는 프로그램은 채점이 불가합니다.

3. 코딩부터 시작하지 마십시오. 실행 할 프로그램에 대한 구조와 알고리즘을 먼저 설계한 후, 컴퓨터 앞에 앉아서 코딩을 시작하세요. 복잡하고 어렵게 코드를 작성하는 사람이 좋은 프로그래머가 아닙니다. 누가 보더라도 쉽게 프로그램의 흐름을 이해할 수 있는 가독성 좋은 쉬운 코드를 작성해 주기 바랍니다.

4. **일찍 시작하세요.** 아무리 완벽하게 계획하고 코딩한다고 해도 중간 과정에서 등장하는 수많은 오류와 버그가 발목을 붙잡을 것입니다!