CSED101. Programming & Problem solving
Fall 2022

Programming Assignment #3
(75 points)

유동준 (ydj2030@postech.ac.kr)

■ **제출 마감일: 2022.11.30 23:59**

■ **개발 환경: Windows Visual Studio 2019**

■ **제출물**

- C 소스 코드 (assn3.c)
  - ➢ 프로그램의 소스 코드에 채점자의 이해를 돕기 위한 주석을 반드시 붙여주세요.
- 보고서 파일 (.docx, .hwp 또는 .pdf; assn3.docx, assn3.hwp 또는 assn3.pdf)
  - ➢ 보고서는 AssnReadMe.pdf를 참조하여 작성하시면 됩니다.
  - ➢ 명예 서약 (Honor code): 표지에 다음의 서약을 기입하여 제출해 주세요: "나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다." 보고서 표지에 명예 서약이 기입되어 있지 않은 과제는 제출되지 않은 것으로 처리됩니다.
  - ➢ 작성한 소스 코드와 보고서 파일은 PLMS를 통해 제출해 주세요.

■ ***주의사항***

- 컴파일이나 실행이 되지 않는 과제는 0점으로 채점됩니다.
- 제출 기한보다 하루 늦게 제출된 과제는 최종 20%, 이틀 늦게 제출된 과제는 최종 40% 감점됩니다. 제출 기한보다 사흘 이상 늦으면 제출 받지 않습니다 (0점 처리).
- 각 문제의 제한 조건과 요구 사항을 반드시 지켜 주시기 바랍니다.
- 모든 문제의 출력 형식은 채점을 위해 아래에 제시된 예시들과 최대한 비슷하게 작성해 주세요.
- 부정행위에 관한 규정은 POSTECH 전자컴퓨터공학부 학부위원회의 "POSTECH 전자컴퓨터공학부 부정행위 정의"를 따릅니다 (PLMS의 본 과목 공지사항에 등록된 글 중, 제목이 [document about cheating]인 글에 첨부되어 있는 disciplinary.pdf를 참조하세요).
- 이번 과제에서는 추가 기능 구현에 대한 추가 점수는 없습니다.

■ Problem: 사다리 게임

(목적)

- 포인터와 동적 할당 및 해제 사용법을 익힌다.
- 텍스트 파일 입출력을 익힌다.

(주의사항)

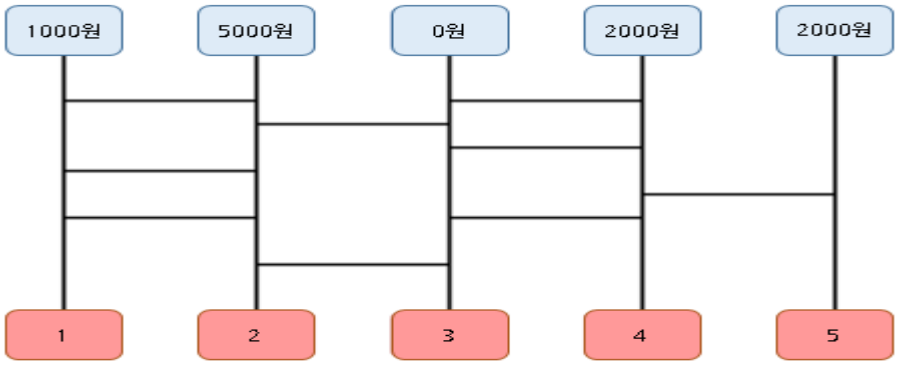
1. 전역변수, goto문, 그리고 구조체 등은 사용하지 않는다.
2. 문제의 출력 형식은 채점을 위해 아래의 실행 예시와 최대한 비슷하게 작성한다.
3. 명시된 예외 처리 외에는 고려하지 않아도 된다.
4. 프로그램에서 랜덤 시드는 프로그램 시작 시 main()에서 srand(time(NULL)); 함수를 한번만 호출하도록 하여 한번만 초기화한다.
5. 문서에 반드시 정의해서 사용해야 할 사용자 정의 함수가 설명되어 있으니 확인 후 구현하도록 합니다.
6. 프로그램 구현 시, main() 함수를 호출을 직접 하지 않습니다. 즉, 소스 코드 내에 main(); 이라고 호출하지 않는다.
7. 모든 기능을 main() 함수에서 구현 한 경우 감점 처리한다.
   기능적으로 독립됐거나 반복적으로 사용되는 기능은 사용자 함수를 정의해서 구현한다.
8. 프로그램 시작 시, 아래와 같이 main()함수에서 사다리 게임 보드 포인터를 선언 후 시작하도록 한다.

```
int **ladder_board = NULL;
```

   사다리 게임 보드는 반드시 2중 포인터를 사용하여 동적으로 할당한 뒤 사용 및 해제한다(5쪽 참고).
   - 동적 할당을 이용하지 않고 크기가 정해진 배열을 선언 후 사용시 0점 처리한다.

(설명)

이번 과제에서 구현할 사다리 타기는 제비 뽑기의 일종으로 일상 생활에서 간식 내기 등을 정할 때 많이 사용한다. 참여하는 인원만큼 세로줄을 긋고 한쪽 편에는 이름을 쓰고 반대쪽에는 내기 금액 등을 적당히 나눠서 쓴 뒤, 세로줄 사이에 가로줄을 겹치지 않게 무작위로 사다리 모양의 선을 그린다. 그리고 정해진 규칙에 따라 한 사람씩 자신이 선택한 출발지점에서 반대쪽 목적지로 움직여 목적지에 있는 금액만큼 지불하게 된다.



(규칙)

1. 세로선의 아래에서 위로 진행한다.
2. 세로선을 따라가다 가로선을 만나면 그 가로선을 따라 바로 옆의 세로선으로 이동하여 다시 위로 진행한다.

[0. 메뉴 출력]

프로그램이 실행되면 사다리 게임을 위한 메뉴를 출력한다. 메뉴 출력 화면은 다음과 같다.

```
[사다리 게임]
================================
1. 사다리 보드 생성
2. 사다리 타기 시작
3. 종료
================================
선택:
```

그림 1. 메뉴 화면

- 1을 입력할 시 사다리 게임을 위한 보드를 생성한다.
- 2를 입력할 시 사다리 타기 게임을 시작한다.
- 3을 입력할 시 프로그램을 종료한다.
- 1~3이외의 숫자와 정수 외의 입력 값에 대해서는 고려하지 않는다.

[1. 사다리 게임을 위한 판 생성]

본 단계에서는 사다리 게임을 위한 판 생성을 하여 텍스트 파일로 저장하게 된다.

아래와 같은 사다리 게임을 위한 판은 가로줄의 위치를 (높이, 왼쪽의 세로줄 번호)의 형태로 나타낼 수 있다. 예를 들면 아래의 경우 (1, 1), (1, 3), (2, 4), (3, 2), (4, 3), (5, 2), … 와 같이 나타낼 수 있다. 이를 이용하여 사다리 게임을 위한 판을 생성해 보자.

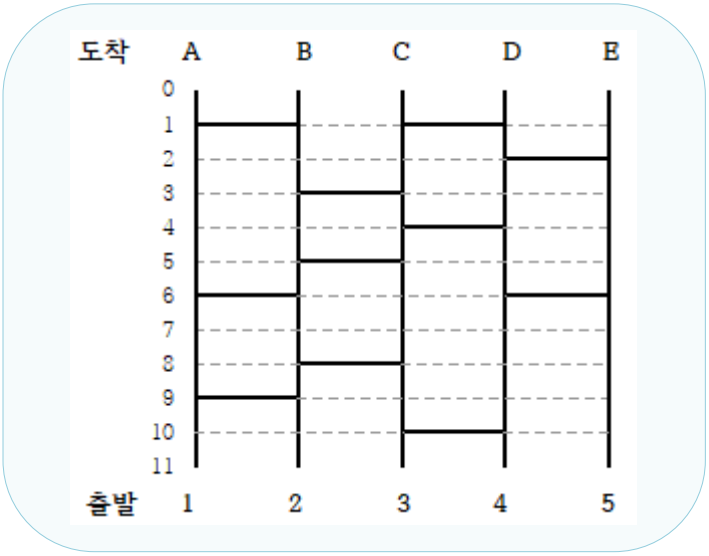


그림 2. 참여 인원수 5명, 사다리 높이 12개, 가로줄 개수 11개인 경우의 예제

메뉴 화면에서 1을 입력하면, 사다리 게임을 위한 판 생성을 위하여 아래의 예시와 같이 참여 인원수, 사다리 높이, 가로줄 개수 그리고 파일이름을 순서대로 입력 받는다.
(예시의 빨간색 밑줄은 사용자 입력에 해당)

```
선택: 1

참여 인원수: 5
사다리 높이: 12
가로줄 개수: 11
파일이름: map1.txt
```

그림 3. 보드 생성을 위한 입력 예시

(가정)

- 참여 인원수 입력 범위: 2명 ~ 26명
- 사다리의 높이는 5개 이상(위의 그림의 경우 0번부터 11번까지 사다리 높이는 12개 임)
- 가로줄의 개수의 입력 범위: 0 ~ (가로줄 전체 개수의 40%)
  단, 소수점 이하는 버린다.
- 파일 이름은 20자를 넘지 않는다고 가정한다.
  생성한 게임 판을, 입력 받은 파일명으로 저장하도록 한다.

사용자로부터 지정 범위내의 참여인원수와 가로줄 개수를 입력 받았으면 사다리 게임 판을 생성한 후 파일에 저장하도록 한다. 이 때, 잘못된 범위의 입력 등은 없다고 가정한다.

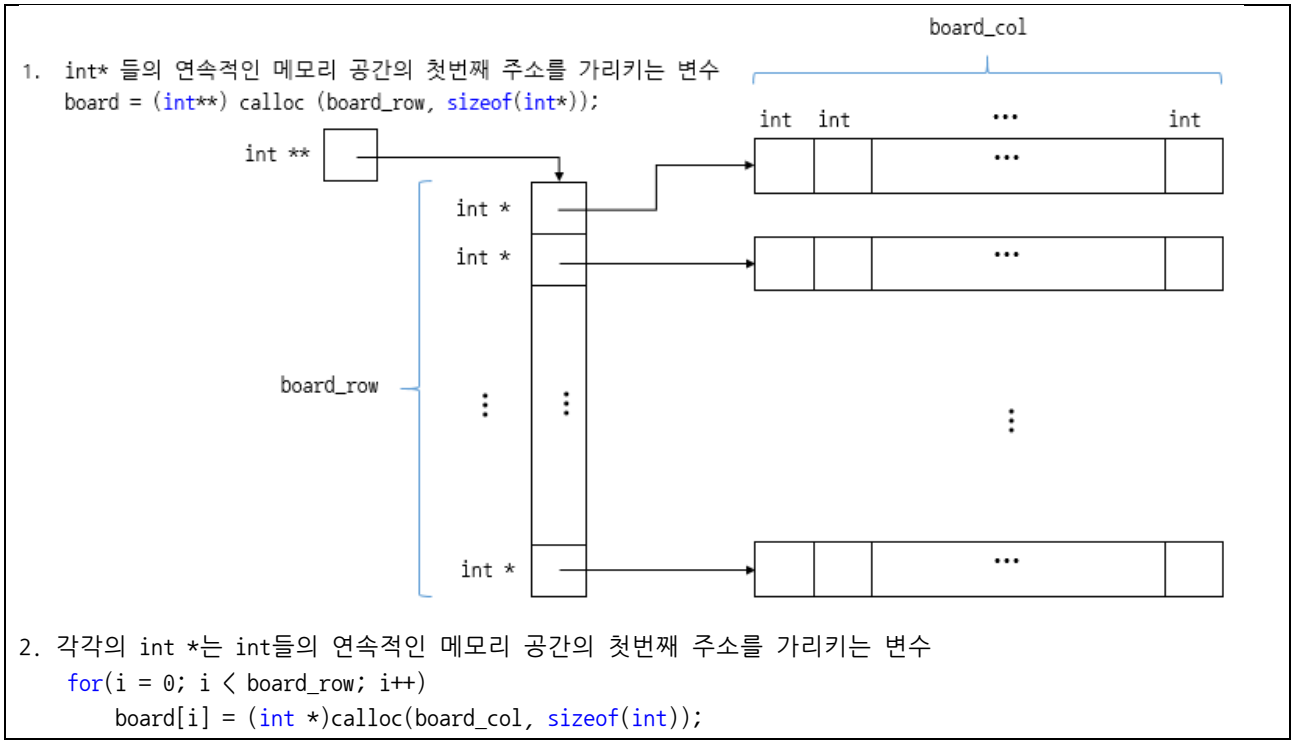
1) 게임 판 생성

게임 판 생성을 위해, int **board의 형태로 선언한 뒤 2차원 동적 할당을 받아 사용하도록 한다.

- 게임 판은 보드 생성할 때마다 크기가 달라지기 때문에 동적 할당을 이용하도록 한다.

```
int ** board;
board = (int **)calloc(board_row, sizeof(int *));
for(i = 0; i < board_row; i++)
   board[i] = (int *)calloc(board_col, sizeof(int));
```

- 2차원 배열에 대한 모식도



2) 가로 줄 채우기

위와 같이 동적 할당을 한 후, 랜덤하게 가로줄을 채운다.

- 가로줄은 0번과 (사다리 높이 - 1)번의 높이에는 넣을 수 없다.
  사다리 높이가 12인 그림 2의 예시로 설명을 하면, 0번과 11번의 높이에는 가로줄을 넣을 수 없다. 즉 (0, 1), (11, 1) 등은 있을 수 없다.
- 인접한 가로줄은 넣을 수 없다. (가로줄 (3, 2)의 인접한 가로줄은 (3, 1)과 (3, 3)이다.)
- 프로그램을 실행할 때마다 랜덤하게 가로줄을 채우도록 구현한다.

- 힌트) 게임 판을 모두 0으로 저장 후, 적절한 가로줄 위치를 찾게 되면 1로 저장한다.

3) 파일에 저장

판을 성공적으로 생성했으면 입력 받은 파일 이름(map1.txt)으로 파일에 저장하도록 한다.

- 파일의 첫 줄에는 참여 인원수, 사다리 높이, 가로줄 개수를 순서대로 출력한다.
- 그 다음 줄부터는 가로줄의 위치를 파일로 출력한다. 가로줄의 위치는 (높이, 왼쪽의 세로줄 번호)의 형태로 출력한다. (가로줄이 하나도 없는 열이 존재할 수 있다.)

map1.txt 예시
5 12 11 **1 1** **1 3** **2 4** **3 2** **4 3** **5 2** **6 1** **6 4** **8 2** **9 1** **10 3**

파일 저장이 끝나면,

- 해당 게임 판은 더 이상 사용하지 않으므로, 동적 할당해준 2차원 배열을 할당 해제한다.
- 게임 판 작성이 완료되었으므로, 그림 1의 메뉴 화면을 출력하여 사용자 입력을 기다린다.

(필수 구현 함수)

반드시 작성해야 하는 함수는 다음과 같다. 함수 기능 외에 함수 이름, 매개 변수, 반환 자료형은 자유롭게 변경이 가능하다. 매개 변수를 추가하는 것 외에 해당하는 변경 사항은 보고서에 기록한다.

이 함수들 말고도 자유롭게 추가로 함수를 정의해서 사용하도록 한다.

- **void generate_ladder(int **board, int num_line, …)**
  사다리의 적절한 위치에 해당 개수만큼 가로줄을 랜덤하게 생성하여 긋는 함수

- **void save_ladder(char filename[], int **board, …)**
  매개변수로 전달받은 파일 이름으로 텍스트 파일을 만들고 생성한 사다리 정보를 저장하는 함수
- **void free_ladder(int **board, …)**
  동적 할당 받은 메모리를 할당 해제하는 함수

[2. 사다리 타기 게임 시작]

본 단계에서는 사다리 타기를 실제로 진행한다. 사다리 게임을 위한 판 구성은 선택 메뉴 1을 통해 생성한 사다리 데이터 파일을 읽어서 구성하도록 한다. 이 때 예시 외에도 규칙에 맞는 모든 데이터 파일에 대해, 정상적으로 사다리가 구성되어야 한다.

메뉴에서 2를 선택하면, 아래와 같이 사용자로부터 파일 이름을 입력 받아, 해당 파일을 읽어 화면을 모두 지우고 그림 5와 같이 출력한다.

- 화면을 지우는 방법은 **Tips-화면 지우기**(마지막 쪽)을 참고하여 구현한다.

```
선택: 2

파일 이름: map1.txt
```

그림 4. 메뉴 2를 선택한 경우

(예외처리) 입력한 파일이 없는 경우 파일이 존재하지 않는다 정도의 적절한 에러 메시지 출력 후, 그림 1의 메뉴 선택화면으로 돌아간다.

아래의 실행 예시는 map1.txt 파일을 읽어 출력한 화면이다.

map1.txt 예시

```
5 12 11
1 1
1 3
2 4
3 2
4 3
5 2
6 1
6 4
8 2
9 1
10 3
```

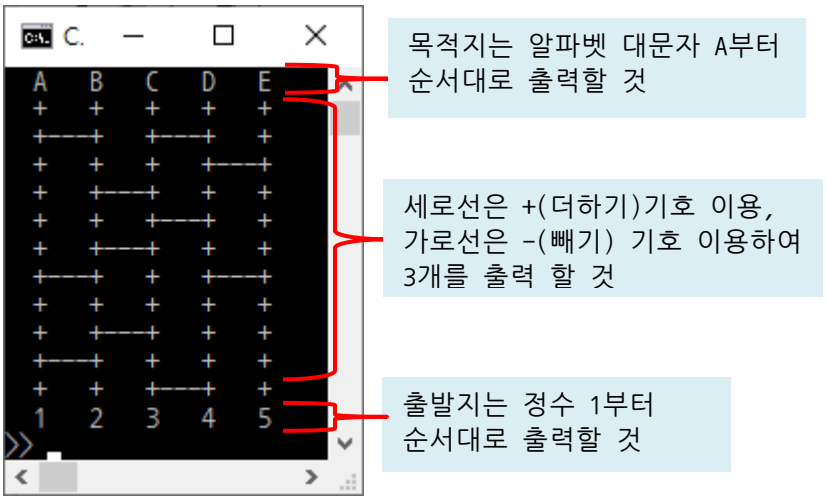


그림 5. 사다리 타기 게임 초기 상태

map1.txt를 실제 사다리로 나타내면 그림 6과 같이 구성된다.

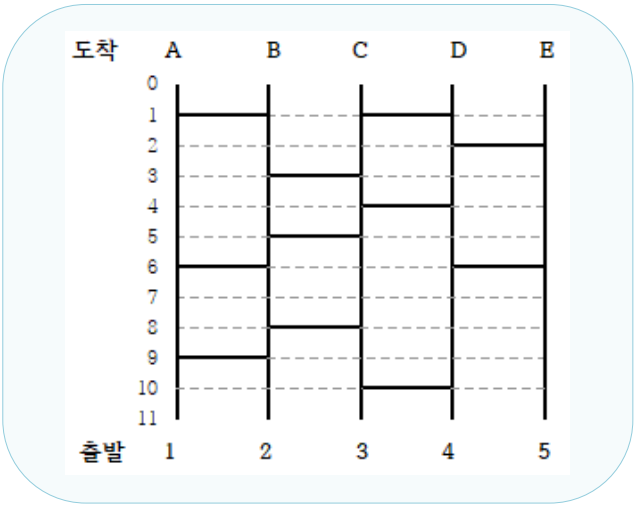


그림 6. 사다리 예시

	0	1	2	3	4	5	6	7	8
0	1		1		1		1		1
1	1	1	1		1	1	1		1
2	1		1		1		1	1	1
3	1		1	1	1		1		1
4	1		1		1	1	1		1
5	1		1	1	1		1		1
6	1	1	1		1		1	1	1
7	1		1		1		1		1
8	1		1	1	1		1		1
9	1	1	1		1		1		1
10	1		1		1	1	1		1
11	1		1		1		1		1

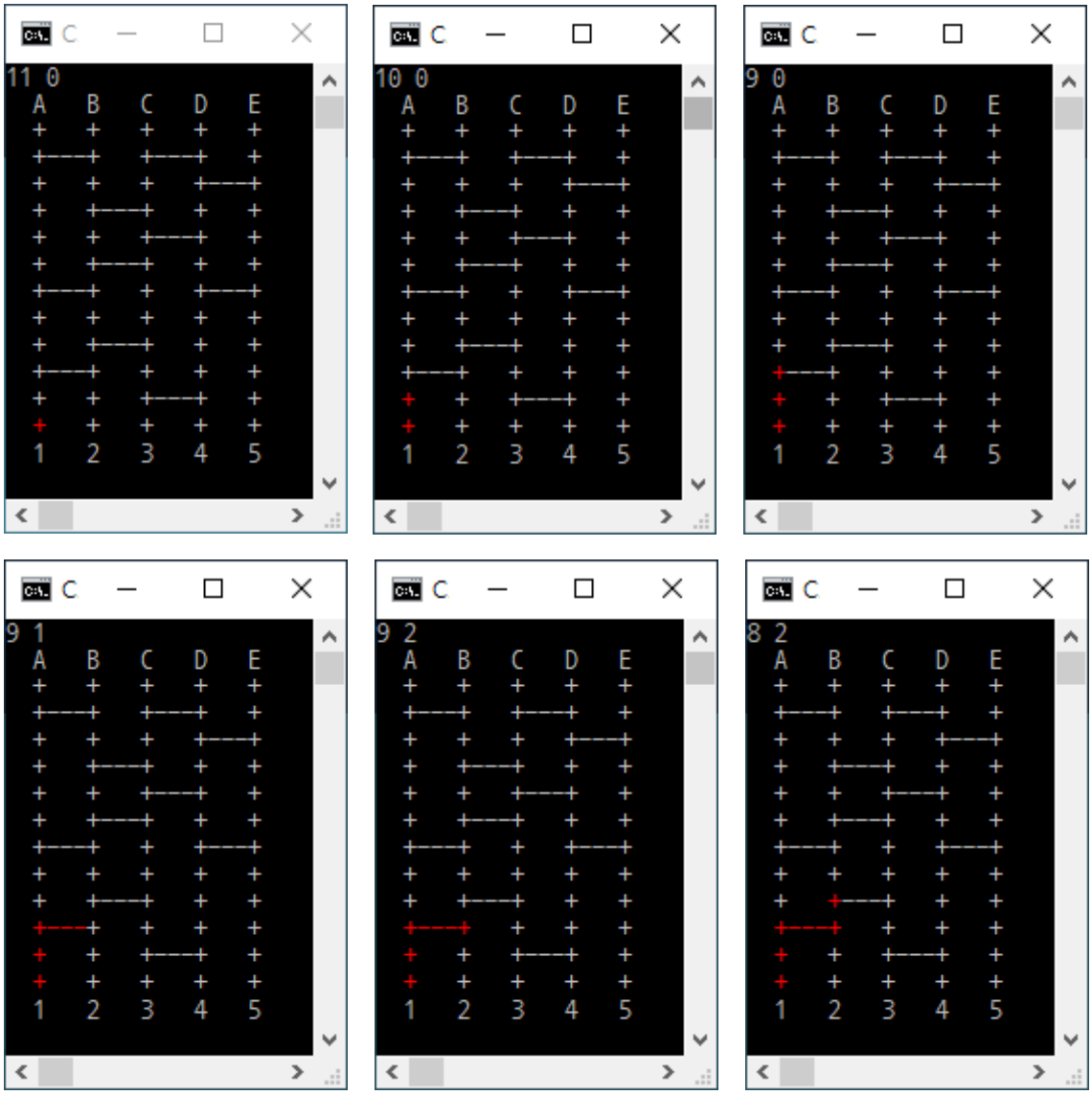
그림 7. 사다리 타기 게임을 위한 판 구성

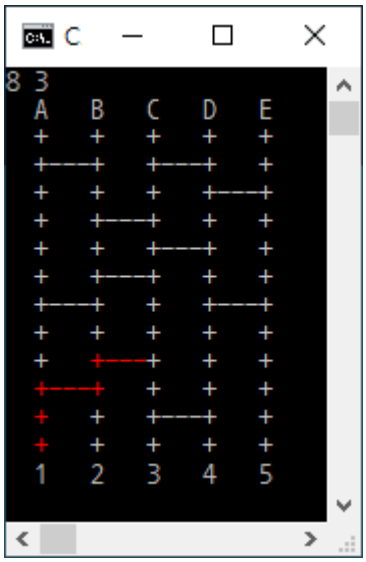
(힌트) 사다리의 선이 그어진 부분의 값은 1(true), 아닌 부분은 0(false)로 생각하고 구현한다. 그림 7처럼 생각할 수 있다.

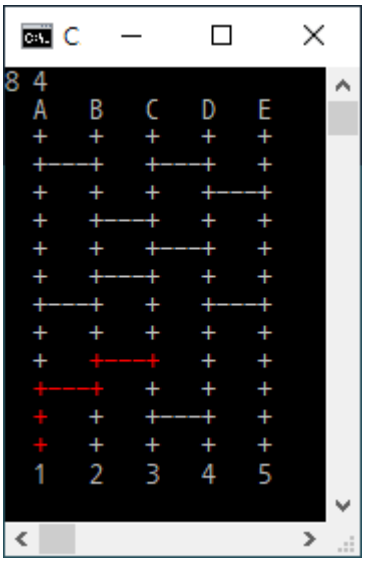
게임을 위한 판 구성 시, 2차원 배열을 동적 할당 해주고 그 배열에 정보를 저장하여 게임을 진행하도록 한다.

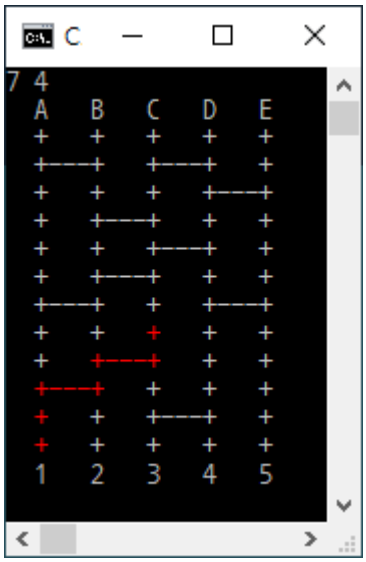
1) 출발지 선택
   - 그림 5과 같이 사다리를 출력한 후, 출발지를 입력 받기 위해 사용자 입력을 기다린다.
   - 사용자는 1부터 참여 인원수까지의 숫자를 선택하여 입력한다. 범위 외의 입력은 고려하지 않는다.

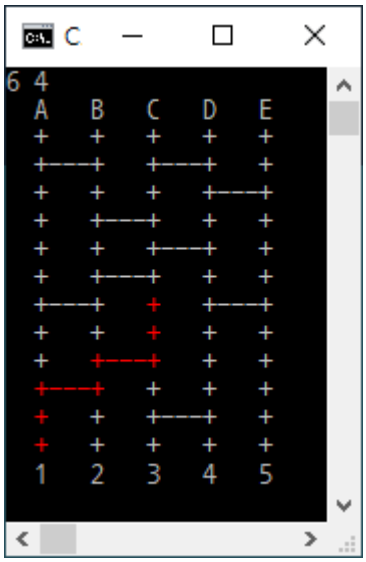
2) 선 긋기
   - 그림 5와 같은 초기상태에서 숫자를 선택하면 선 긋기가 시작된다. 아래는 1을 입력한 경우에 해당한다.
   - 실행 화면의 첫 줄은 현재 사다리의 이동 좌표를 나타내며, 현재까지의 이동경로가 빨간색으로 표시됨을 볼 수 있다.
   - 사다리 판에서 엔터를 입력할 때마다 위치가 이동되며, 화면을 아래와 같이 갱신한다. 갱신하기 전, 화면 지우기 기능을 이용하여 화면을 지우도록 한다.
   - 목적지까지 도달하면 도착 지점을 출력한다.

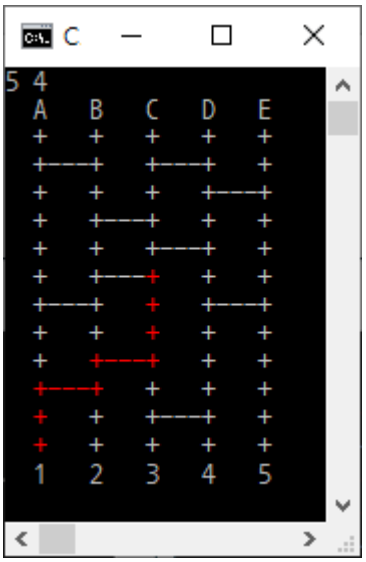


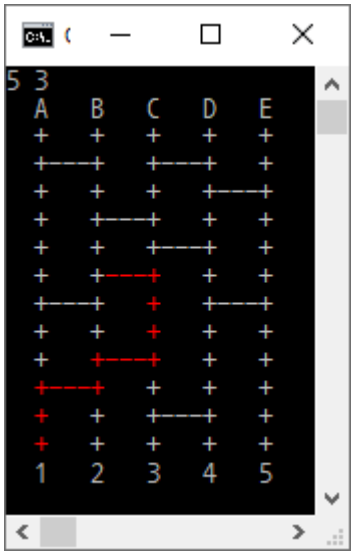




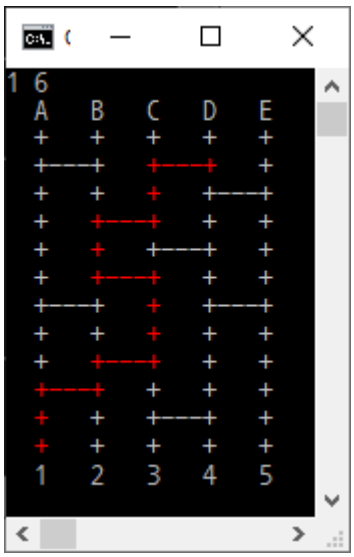


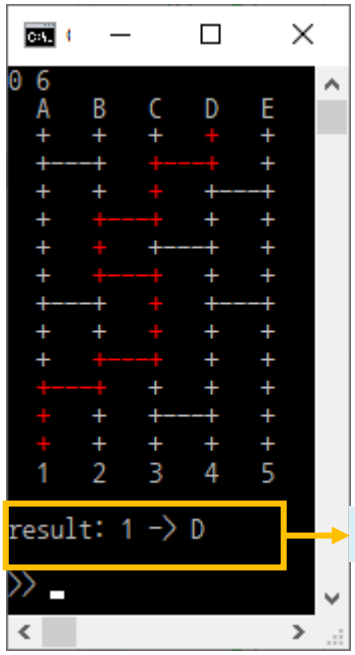






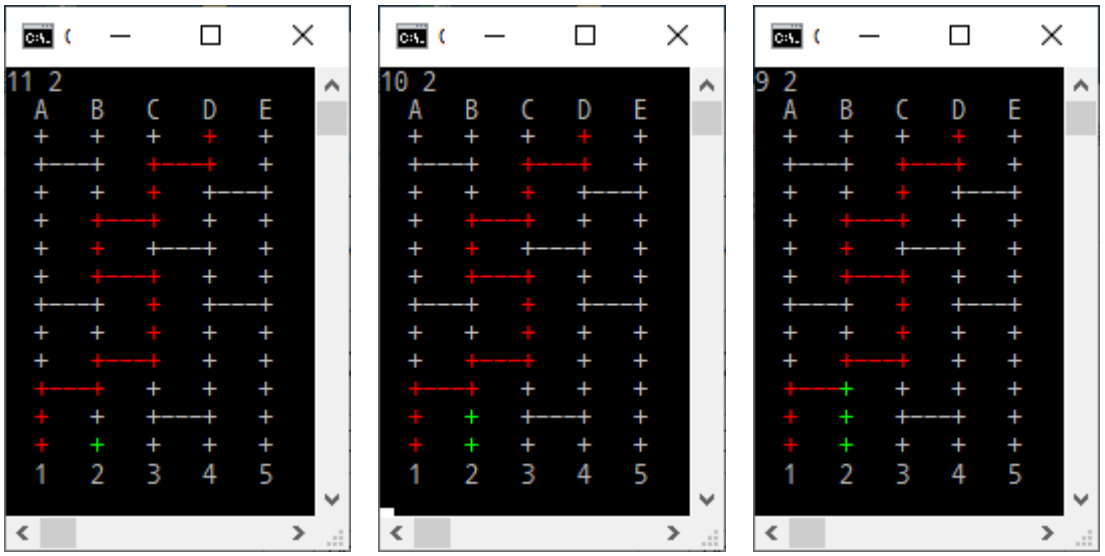
중간 과정 생략

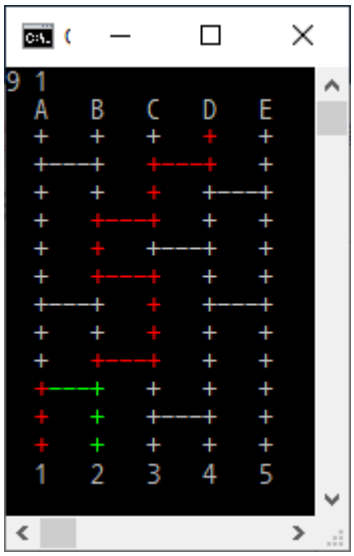




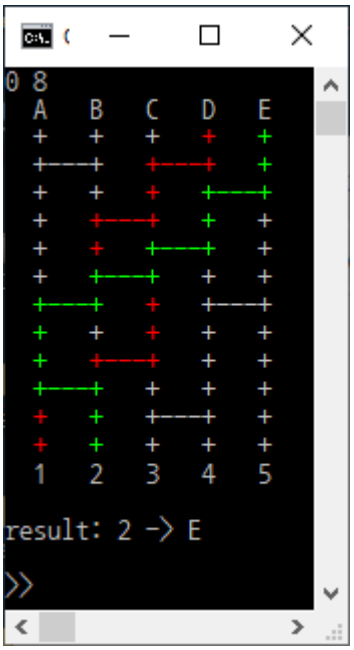
목적지 출력

목적지에 도달 한 후, 다음 입력을 기다린다. 아래는 다음 입력으로 2를 선택한 경우이다.
이번에는 연두색으로 선 긋기가 시작되며, 최신 색으로 지나가는 자리마다 색을 업데이트 하도록 한다.





중간 과정 생략



3) 글자 색
글자 색은 (출발지 번호 % 5) 연산의 결과가 1이면 빨간색, 2이면 연두색, 3이면 노란색, 4이면 파란색, 0이면 자주색으로 사용하도록 한다. 색을 바꾼 경우 보고서에 기록한다.
글자 색은 **Tips-글자 색 출력**(마지막 쪽)을 참고하여 구현한다.

4) 사용자 입력
출발지를 선택하여 입력하는 부분에,

- 0을 입력하는 경우, 사다리 타기 게임을 종료하고 그림 1의 메뉴 화면으로 돌아간다. 이 때, 사다리 타기 게임을 위해 동적 할당한 메모리는 할당 해제 한다.

- -1을 입력하는 경우, 1번부터 순서대로 사다리 타기를 진행하여 그 결과를 아래의 그림 8과 같이 출력한 후 사용자 입력을 기다린다. (순서대로 사다리 타기 진행 시, 엔터 입력 없이 진행되도록 한다.)

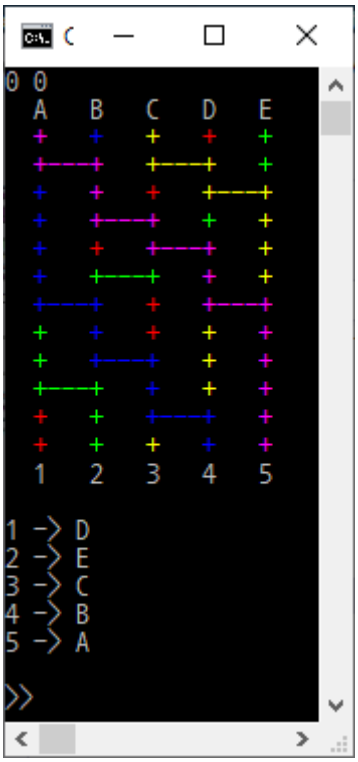


그림 8. 사용자 입력으로 -1을 입력한 경우의 출력 예시

(필수 구현 함수)
반드시 작성해야 하는 함수는 다음과 같다. 함수 기능 외에 함수 이름, 매개 변수, 반환 자료형은 자유롭게 변경이 가능하다. 매개 변수를 추가하는 것 외에 해당하는 변경 사항은 보고서에 기록한다.

이 함수들 말고도 자유롭게 추가로 함수를 정의해서 사용하도록 한다.

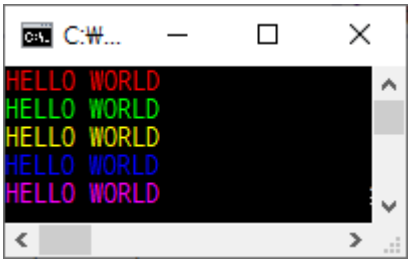
- `int ** load_ladder(char filename[], …)`
  매개변수로 전달받은 파일이름으로 파일을 열어, 사다리 데이터를 읽고 그림 7과 같은 사다리 판을 생성하는 함수로 동적 할당 받은 사다리 판을 반환

[Tips]

■ 글자 색 출력

콘솔에 색을 출력하기 위해서는 Escape 코드("\033")를 활용해야 한다. 이를 활용한 함수는 아래와 같으며, set_color(색상코드) 함수를 호출하면, 이후에 출력되는 글자의 색이 지정된 색상코드 색으로 출력이 된다. reset_color()를 호출하면 글자에 지정된 색이 원래대로 돌아온다.
이 과제에서 사용할 색상코드는 다음과 같다.

빨간색	91
연두색	92
노란색	93
파란색	94
자주색	95
RESET	0



윈도우에서 안 되는 경우, 노란색 형광으로 표시한 부분의 주석을 모두 풀어서 사용합니다.

```c
#include <stdio.h>
//#include <windows.h> // GetConsoleMode () 등의 사용을 위해 포함

/*void set_vt_mode() {
    DWORD l_mode;
    GetConsoleMode(GetStdHandle(STD_OUTPUT_HANDLE), &l_mode);
    SetConsoleMode(GetStdHandle(STD_OUTPUT_HANDLE), l_mode | 0x0004 | 0x0008);
}*/

void set_color(int code) {
    printf("\033[%dm", code);
}

void reset_color() {
    printf("\033[0m");
}

int main() {
    //set_vt_mode(); // 메인에서 set_color() 함수 호출되기 전에 1회만 호출하면 됨
    for (int i = 91; i <= 95; i++)
    {
        set_color(i); // 색상코드 설정
        printf("HELLO WORLD\n");
        reset_color();
    }
    return 0;
}
```

■ 화면 지우기

화면을 지우고 싶을 경우, stdlib 헤더 파일을 포함시킨 후 system() 함수를 사용할 수 있습니다. system() 함수의 매개변수로 "cls"를 넘겨주면 됩니다. 아래의 소스 코드를 컴파일하여 실행하면 콘솔 화면에 "Erase me!" 문자열 이 출력됩니다. Enter 키를 입력하면 화면이 지워진 후에 "Erased" 문자열이 출력되는 것을 볼 수 있습니다.

```c
#include <stdio.h>
#include <stdlib.h> // system() 사용을 위해 포함
int main() {
    char c;
    printf("Erase me!\n");
    scanf("%c", &c);
    system("cls"); // 리눅스 또는 맥의 경우 system("clear");
```

```
    printf("Erased\n");
    return 0;
}
```