CSED101. Programming & Problem solving

Spring, 2018

Programming Assignment #3

이승관(seungkwan@postech.ac.kr)

- ***Due***: **2018.05.04 23:59**
- ***Development Environment***: **GNU C Compiler (GCC) and Vi Editor (Editor is optional)**

- ***제출물***
  - **C Code files (*.c)**
    - 프로그램의 소스 코드를 이해하기 쉽도록 반드시 **주석**을 붙일 것.
  - **보고서 파일** (.doc(x) or .hwp) 예) assn3.doc(x) 또는 assn3.hwp
    - 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.
  - 문제에 제시되어 있는 파일이름으로 제출 할 것. 그 외의 다른 이름으로 제출하면 감점 또는 0점 처리된다.
  - 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.)
  - 과제 작성시 전역변수는 사용할 수 없으며, 사용자 정의 함수를 적절히 작성하도록 한다.
  - 이번 과제에서는 추가 기능 구현에 대한 추가 점수는 없습니다.

Problem: 사다리 게임

(목적)

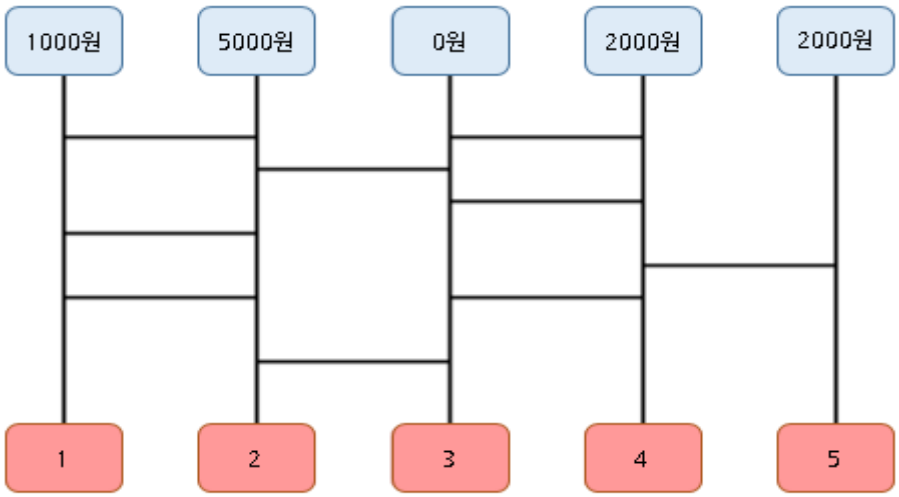
- 2차원 배열의 선언과 사용을 익힌다.
- 함수에서 2차원 배열을 매개변수로 사용하는 방법을 익힌다.
- 텍스트 파일 입출력을 익힌다.

(주의사항)

1. 보고서는 "assn3.doc" 또는 "assn3.hwp"로 저장 할 것 (보고서는 통합하여 작성)
2. Problem 1은 assn3_1.c , Problem 2는 assn3_2.c 로 저장할 것.
3. 실행 시, 정수를 요구했을 때에는 정수를 입력한다고 가정한다.
4. 전역변수, goto문, 포인터 그리고 구조체 등은 사용하지 않는다.
5. 프로그램 구현 시, main() 함수를 호출하여 사용하지 않는다.
6. 적절히 사용자 정의 함수를 작성하여야 한다. (main() 함수 내에 모든 기능을 구현 시 감점)
7. 문제의 출력 형식은 채점을 위해 아래의 실행예시와 최대한 비슷하게 작성해 주세요.

(설명)

이번 과제에서 구현할 사다리 타기는 제비 뽑기의 일종으로 일상 생활에서 간식 내기 등을 정할 때 한번씩은 해 보셨죠? 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 그립니다. 그리고 정해진 규칙에 따라 한 사람씩 자신이 선택한 출발지점에서 반대쪽 목적지로 움직여 목적지에 있는 금액만큼 지불하게 됩니다.



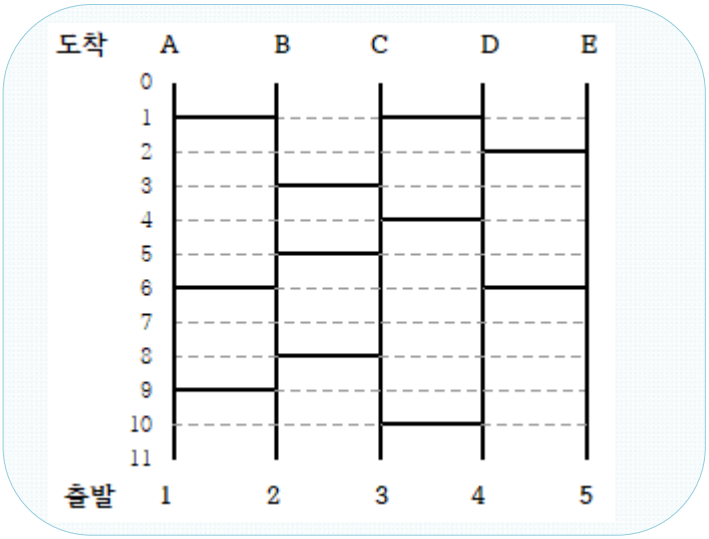
(규칙)

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

■ Problem 1: 사다리 게임을 위한 판 생성

(문제)

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.



(설명 및 요구사항)

프로그램을 실행하면, 아래와 같이 게임의 참여할 인원수와 가로줄 개수를 입력 받는다.

(실행예제의 빨간색 밑줄은 사용자 입력에 해당)

```
seungkwan@seungkwan-pc:~$ ./a.out
참여인원수: 5
가로줄 개수: 10
```

(가정)

- 참여인원수 입력범위: 2명~10명
- 사다리의 높이는 12로 고정(위의 그림으로 설명하면 0번부터 11번까지 12개)
- 가로줄의 개수의 입력 범위: 0 ~ (참여인원수 – 1) x 5개

만약 참여인원수 혹은 가로줄의 개수가 지정된 범위를 벗어났을 경우, 다음과 같이 프로그램을 종료한다. 단, 정수 외의 입력은 고려하지 않는다.

```
seungkwan@seungkwan-pc:~$ ./a.out
참여인원수: 11
가로줄 개수: 20
입력이 잘못됐습니다.
seungkwan@seungkwan-pc:~$ ./a.out
참여인원수: 5
가로줄 개수: 100
입력이 잘못됐습니다.
seungkwan@seungkwan-pc:~$
```

사용자로부터 지정 범위내의 참여인원수와 가로줄 개수를 입력 받았으면 사다리 게임 판을 생성한 후 파일에 저장하도록 한다.

2차원 배열을 이용하여 사용자 입력한 가로줄 개수만큼 랜덤하게 가로줄을 채운다.

- 가로줄은 0번과 11번의 높이에는 넣을 수 없다. 즉 (0, 1), (11, 1) 등은 있을 수 없음
- 인접한 가로줄은 넣을 수 없다. (가로줄 (3, 2)의 인접한 가로줄은 (3, 1)과 (3, 3)이며, 가로줄 (3, 2)가 먼저 생성된 (3, 1)과 (3, 3)의 위치에는 가로줄을 넣을 수 없다.)
- 프로그램을 실행할 때마다 랜덤하게 가로줄을 채우도록 구현할 것.

판을 성공적으로 생성했으면 map.txt 파일에 저장하도록 한다. 파일의 첫 줄에는 참여인원수와 가로줄 개수를 출력한다. 그 다음 줄부터는 가로줄의 위치를 파일로 출력한다. 가로줄의 위치는 (높이, 왼쪽의 세로줄 번호)의 형태로 출력하도록 한다. (가로줄이 하나도 없는 열이 존재할 수 있다.)

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

(코드작성)

반드시 작성해야 하는 함수는 다음과 같다. 이 함수들 말고도 자유롭게 추가로 함수를 정의해서 사용하도록 한다.

- **int main() 함수**
  참여인원수와 가로줄 개수를 입력 받아서, 사다리 게임의 규칙에 맞게 사다리를 랜덤 생성한다. 사다리는 int ladder[(MAX_PLAYER – 1) * 5][2] 배열에 저장하며, 이 배열은 main함수 안에서 선언해야 한다. 이 배열의 한 행이 가로줄 하나를 의미한다.

- **void GenerateLadder(int num_player, int num_hline, int ladder[][2]) 함수**
  사다리를 랜덤하게 생성하는 함수이다. 사람의 수, 가로줄의 수, 그리고 생성된 사다리를 저장할 배열을 인자로 받아서, 해당 개수만큼 가로줄을 랜덤하게 생성하여 인자로 받은 ladder 배열에 저장한다. (필요한 매개변수가 있으면 추가하여 구현할 것)

- **int SaveLadder(int num_player, int num_hline, int ladder[][2]) 함수**
  사람의 수, 가로줄의 수, 그리고 사다리가 저장된 배열을 인자로 받아서, map.txt 파일을 만들고 사다리를 저장하는 함수이다. 저장 성공시 1을 반환하고, 실패시 0을 반환한다. (필요한 매개변수가 있으면 추가하여 구현할 것)
  힌트) fopen 함수는 성공, 실패시 반환값이 다르다.

■ Problem2: 사다리 타기 시뮬레이션

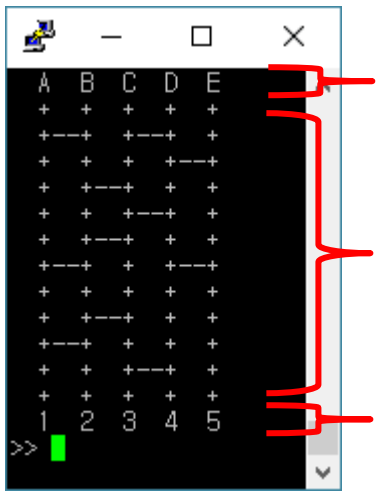
(문제)

문제1의 결과로 만들어진 사다리 데이터 파일을 읽어서 시뮬레이션을 진행하는 프로그램을 작성한다. 이 때 예시 외에도 규칙에 맞는 모든 데이터 파일에 대해, 프로그램이 정상적으로 동작해야 한다.

(설명 및 요구사항)

프로그램을 실행하면 "map.txt"에서 정보를 읽어 아래의 그림 1과 같이 출력한다.

map.txt 예시
5 11
1 1
1 3
2 4
3 2
4 3
5 2
6 1
6 4
8 2
9 1
10 3



목적지는 알파벳 대문자 A 부터 순서대로 출력할 것

세로선은 +(더하기)기호 이용, 가로선은 -(빼기) 기호 이용하여 2개를 출력 할 것

출발지는 정수 1부터 순서대로 출력할 것

그림 1. 초기 상태

map.txt를 실제 사다리로 나타내면 그림 2와 같이 구성된다.

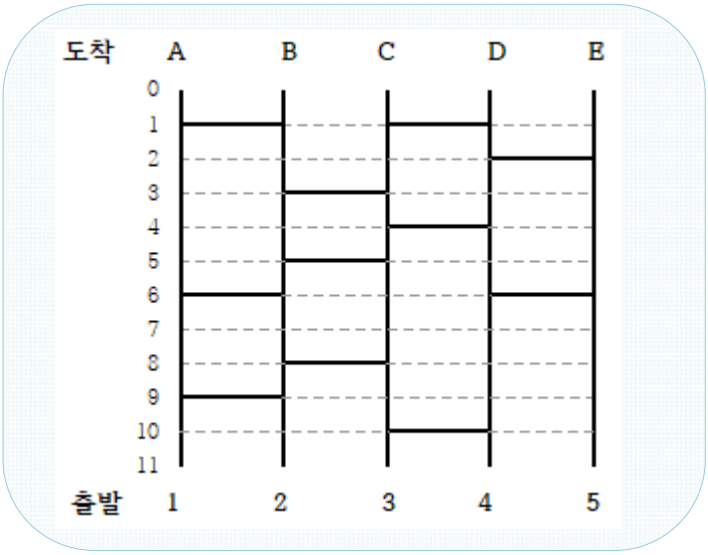


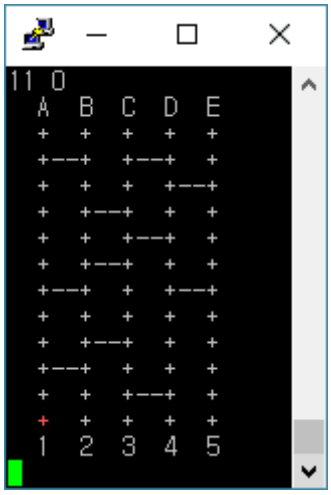
그림 2. 사다리 예시

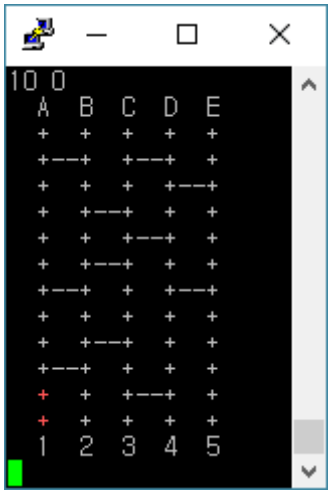
	0	1	2	3	4	5	6	7	8
0	1		1		1		1		1
1	1	1	1		1		1		1
2	1		1		1		1	1	1
3	1		1	1	1		1		1
4	1		1		1	1	1		1
5	1		1	1	1		1		1
6	1	1	1		1		1	1	1
7	1		1		1	1	1		1
8	1		1	1	1		1		1
9	1	1	1		1		1		1
10	1		1		1	1	1		1
11	1		1		1		1		1

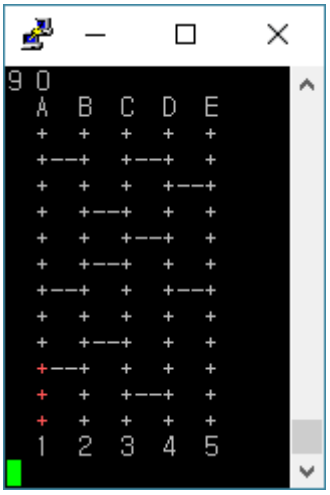
그림 3. 사다리 타기 게임을 위한 판 생성

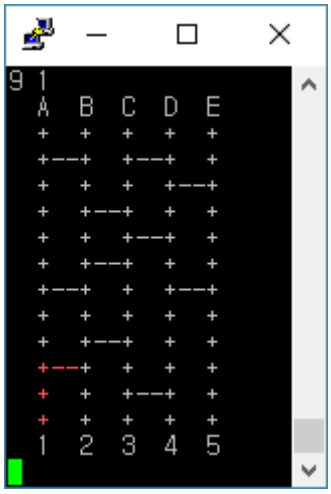
(힌트) 사다리의 선이 그어진 부분의 점은 1(true), 아닌 부분은 0(false)로 생각하고 구현한다. 그림 3처럼 생각할 수 있다.

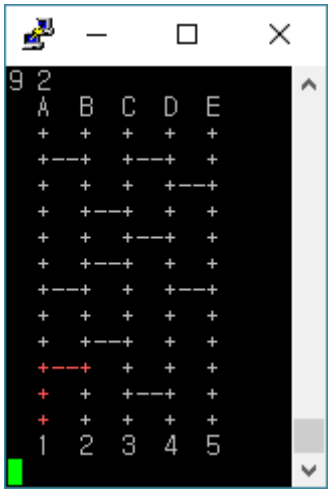
그림 1과 같이 사다리를 출력한 후, 사용자 입력을 기다린다. 사용자는 1부터 참여인원수까지의 숫자를 선택하여 입력한다. 범위 외의 입력은 고려하지 않는다.
그림 1과 같은 초기상태에서 숫자를 선택하면 선 긋기가 시작된다. 아래는 1을 입력한 경우에 해당한다. 실행화면의 첫 줄은 현재 사다리의 이동 좌표를 나타내며, 현재까지의 이동경로가 주황색으로 표시됨을 볼 수 있다. 사다리 판에서 엔터를 입력할 때마다 위치가 이동되며, 화면을 아래와 같이 갱신한다. 화면을 갱신하기 전에 system("clear")를 이용하여 화면을 지운 후 목적지까지 도달하면 도착지점을 출력한다.

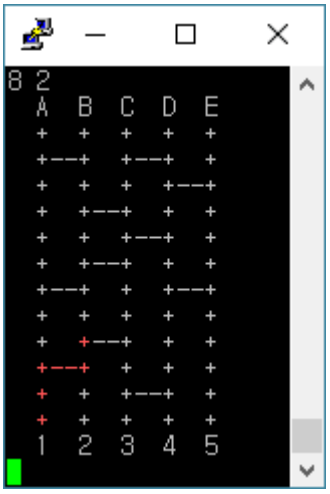


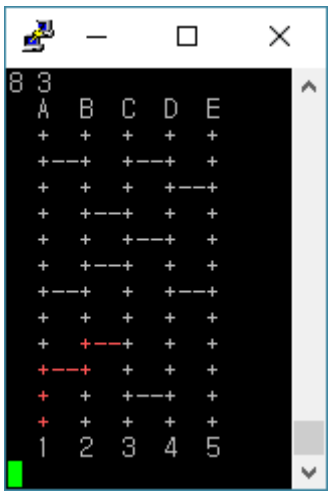


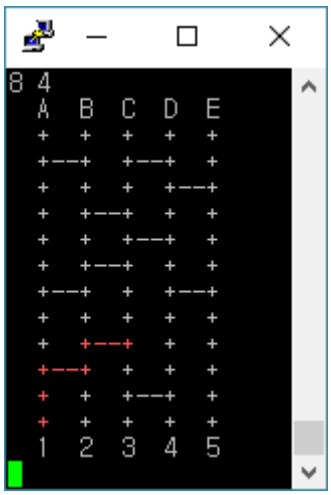


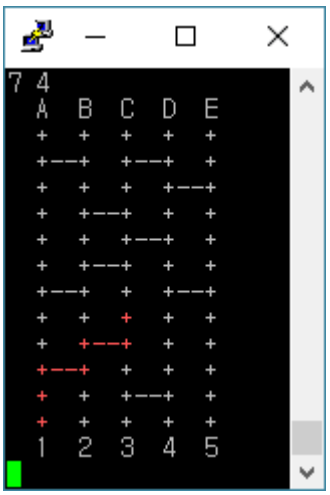


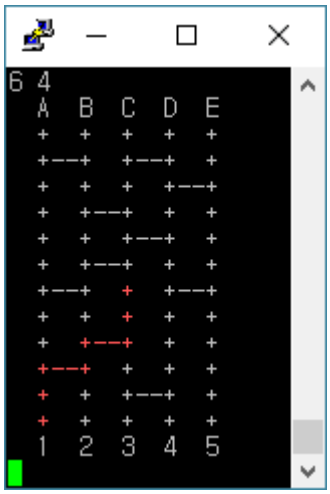
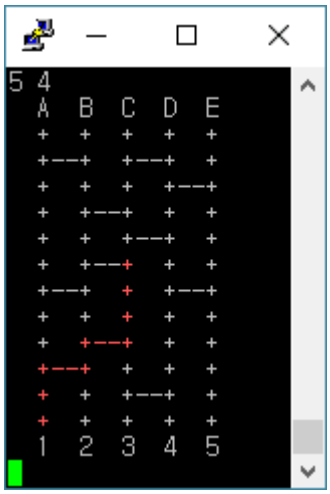
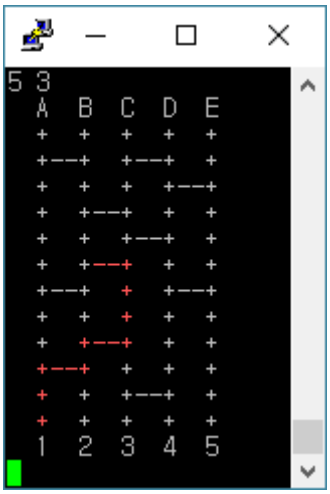




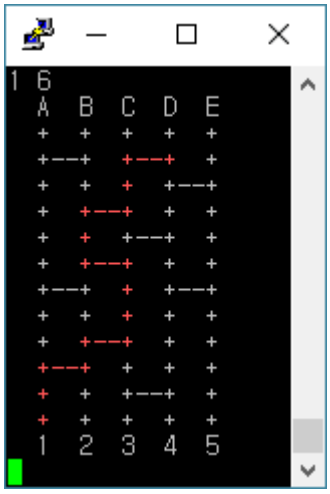
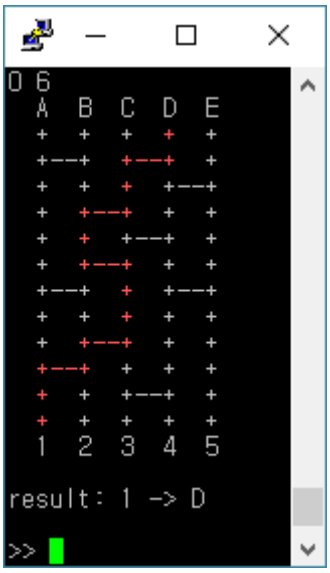






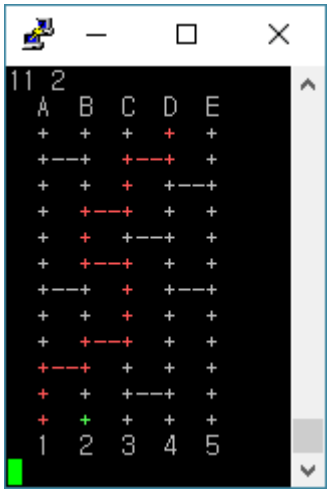
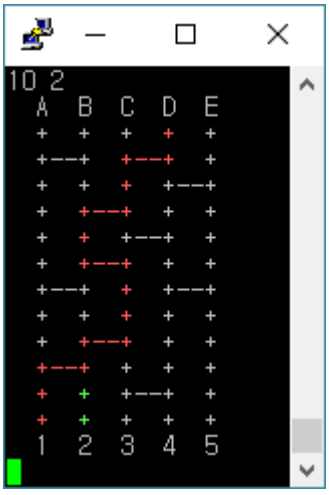
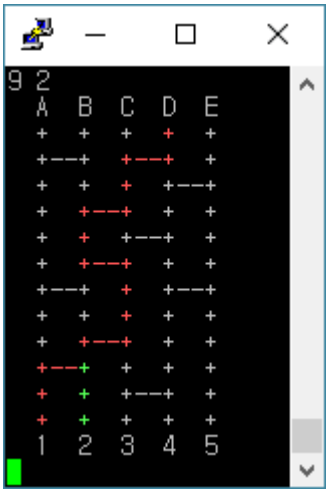




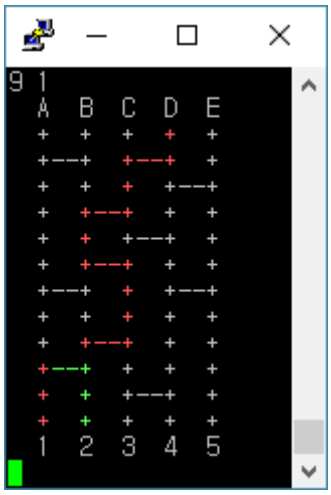
중간 과정 생략

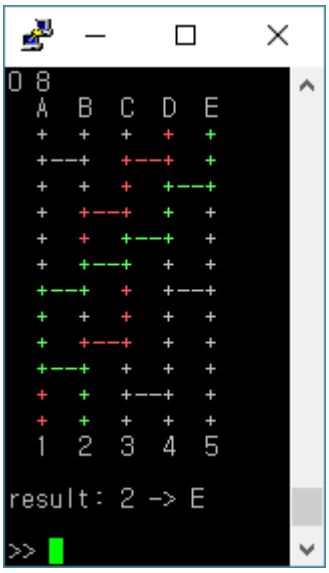
위의 실행 예시 중 마지막 실행 예시는 목적지에 도달하여 그 결과를 (1 -> D) 출력함을 볼 수 있다.

그리고, 다음 입력을 기다린다. 아래는 다음 입력으로 2를 선택한 경우이다. 이번에는 연두색으로 선 긋기가 시작되며, 최신 색으로 지나가는 자리마다 색이 업데이트 됨을 볼 수 있다.

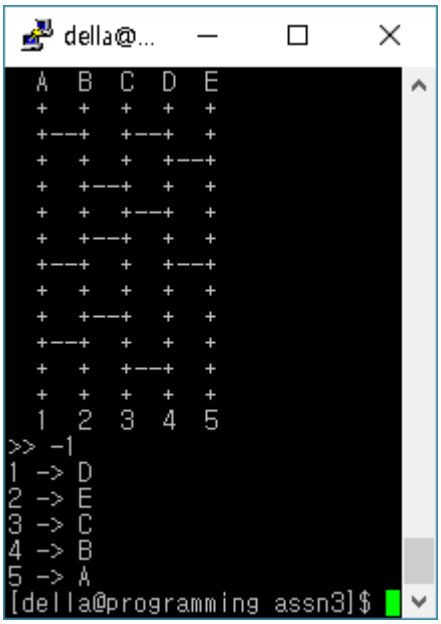


중간 과정 생략



1, 6은 주황색, 2, 7은 연두색, 3, 8은 연한 노란색, 4, 9는 연한 파란색, 5, 10은 연한 자주색으로 선긋기를 하면 된다. 글자색은 11쪽의 "*리눅스에서 글자색 출력*" 부분을 참고한다.

사용자 입력에서 0을 입력하는 경우 프로그램이 종료된다.
사용자 입력에서 -1을 입력하는 경우 1번부터 순서대로 사다리 타기를 진행하여 그 결과를 아래와 같이 출력한 후 종료한다. 또한 파일 result.txt 에는 1번부터 순서대로 이동 경로를 출력한다. 해당 출발 번호 출력 후, 이동 경로를 순서대로 출력하도록 한다. (9쪽의 result.txt 예시를 참고할 것.)



참고로, 생성된 result.txt 파일을 윈도우 메모장에서 열면 한 줄에 모든 내용이 표시 될 수 있습니다. 리눅스와 윈도우의 줄바꿈 문자의 차이 때문에 발생되는 현상으로 리눅스에서 열면 문제가 없습니다.


result.txt 예시
출발번호: 1
1
11 0
10 0
9 0
9 1
9 2
8 2
8 3
8 4
7 4
6 4
5 4
5 3
5 2
4 2
3 2
3 3
3 4
2 4
1 4
1 5
1 6
0 6
1번의 이동경로
출발번호: 2
2
11 2
10 2
9 2
9 1
9 0
8 0
7 0
6 0
6 1
6 2
5 2
5 3
5 4
4 4
4 5
4 6
3 6
2 6
2 7
2 8
1 8
0 8
이하 생략
2번의 이동경로

(코드작성)

반드시 작성해야 하는 함수는 다음과 같다. 이 함수들 말고도 자유롭게 추가로 함수를 정의해서 사용하도록 한다.

- **int main() 함수**

  프로그램이 시작되면 LoadLadder 함수를 호출해서 사다리 판 배열을 생성한다.. 그리고 사용자의 입력에 따른 시뮬레이션을 진행한다.

  int ladder_board[MAX_ROW][MAX_COL] 배열로 사다리 판을 나타낸다. 이 배열은 그림3과 같은 형태로 사다리를 저장한다. 이 배열은 main함수 안에서 선언해야 한다. 사다리 판을 구성하기 위하여 MAX_ROW와 MAX_COL을 적절한 상수로 define한 후 사용하도록 한다.

- **int LoadLadder(int ladder_board[][MAX_COL]) 함수**

  map.txt 파일로부터 사다리 데이터를 읽고, 인자로 받은 ladder_board 배열에 그림 3과 같은 형태로 사다리 배열을 채우는 함수이다.

  map.txt 파일이 없는 경우 0을 반환하여 프로그램을 종료하는데 이용하도록 한다.

  파일을 읽어 사다리 배열을 채우게 되면 1을 반환한다.

  단, 사다리 규칙에 맞지 않은 map.txt 파일에 대해서는 고려하지 않아도 된다.

  (필요한 매개변수가 있으면 추가하여 구현할 것)

(힌트)

■ 리눅스에서 글자색 출력

ANSI ESC code를 사용하여 리눅스 터미널에 출력되는 글자의 색을 바꿀 수 있다.

"<ESC>[<색상코드>m" 을 터미널에 출력하면, 이후에 출력되는 글자의 색이 색상코드에 따라 바뀌게 된다. ESC를 출력하기 위해서는 ASCII 코드 0x1b를 사용하면 된다. 색상코드는 다음과 같은 것들이 있다.

주황색	91
연두색	92
노란색	93
밝은 파란색	94
밝은 자주색	95
RESET	0

한번 색상코드를 지정하면, 다른 색으로 다시 지정하거나, 초기화를 하기 전까지 해당 색이 출력된다. 아래의 코드는 빨간색으로 "HELLO WORLD"를 출력하는 예시이다.

```c
#include <stdio.h>
int main()
{
	printf("\x1b"); // ESC 출력
	printf("[91m"); // 주황색 색상코드 출력
	printf("HELLO WORLD\n"); // 원하는 문자 출력
	printf("\x1b\n"); // ESC 출력
	printf("[0m\n"); // 초기화 코드 출력
	return 0;
}
```

다음과 같이 매크로를 사용하면, 좀 더 편하게 색을 지정할 수 있다.

```c
#define RED "\x1b[31m"
#define RESET "\x1b[0m"
#include <stdio.h>
int main()
{
	printf(RED); // 빨강 색상코드 지정
	printf("HELLO WORLD\n"); // 원하는 문자 출력
	printf(RESET); // 색상 초기화

	printf(RED "HELLO WORLD\n" RESET); // 이렇게 한번에 붙여서 출력해도 됨

	return 0;
}
```

■ 화면 지우기

화면을 지우기 위해서는 stdlib.h를 포함된 system 함수를 사용한다. 다만, 윈도우의 비주얼 스튜디오와 리눅스의 프로그래밍 환경이 다름에 따라서 system 함수로 넘겨줄 매개변수로 "cls" 대신 "clear"를 사용한다.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
        printf("HELLO WORLD\n");
        system("clear"); // 윈도우의 경우 system("cls");
        return 0;
}
```

■ 배열 사용

- 배열의 크기를 선언할 때 변수를 사용하면 안 된다. (char led[i][j];).
- 충분한 크기로 배열을 선언 한 뒤 필요한 부분만 화면에 출력한다.
- 2차원 배열을 함수의 매개변수로 넘겨주기 위해서는 led[][size]와 같은 표현을 사용한다.

```
int arrayTestFunc(int arr[][colSize], int …);
int main()
{
        int arr[3][3] = {0, 1, 2, 3, 4, 5, 6, 7, 8};
        int arrSize = 9;
        arrayTestFunc(arr, arrSize);
        return 0;
}
```

■ 어싸인 팁

1. UNIX/LINUX 는 서버용 운영체제이기 때문에 putty 창을 여러 개 띄워놓고 동시에 작업해도 된다. 창 하나에는 VIM 을 띄워놓고 다른 창에서는 gcc 를 이용하면 매번 VIM 을 껐다 켰다 하지 않아도 된다.

2. 프로그램 작성 중 무한루프에 의하여 프로그램이 끝나지 않을 경우 Ctrl+C 를 누르면 된다.

3. 혹시 윈도우에서 개발하고 리눅스로 파일만 옮기는 학생들의 경우 반드시 linux 에서 제대로 컴파일 되고 동작되는지, 또 소스 파일은 제대로 열리는지 확인하도록 한다. linux 환경에서 제대로 실행되지 않으면 감점된다.