CSED101. Programming & Problem solving

Fall, 2018

Programming Assignment #3

(70 points)

서민교(mkseo@postech.ac.kr)

- ***Due***: **2018.11.15 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** (.docx or. hwp) 예) assn3.docx 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 LMS 를 이용하여 제출한다.

- ***주의사항***
  - 각 문제에 해당하는 요구사항을 반드시 지킬 것.
  - 모든 문제의 출력 형식은 예시들과 동일해야 하며, 같지 않을 시 감점이 된다.
  - 각 문제에 제시되어 있는 파일이름으로 제출 할 것. 그 외의 다른 이름으로 제출하면 감점 또는 0 점 처리된다.
  - 컴파일 & 실행이 안되면 무조건 0 점 처리된다.
  - **하루 late 시 20%가 감점되며, 3 일 이상 지나면 받지 않는다. (0 점 처리)**
  - 부정행위에 관한 규정은 POSTECH 전자컴퓨터공학부 학부위원회의 'POSTECH 전자컴퓨터공학부 부정행위 정의'를 따른다. (LMS 의 과목 공지사항의 제목 [document about cheating]의 첨부파일인 disciplinary.pdf 를 참조할 것.)
  - 이번 과제에서는 추가 기능 구현에 대한 추가 점수가 있습니다.

■ Problem: simple pang

(목적)

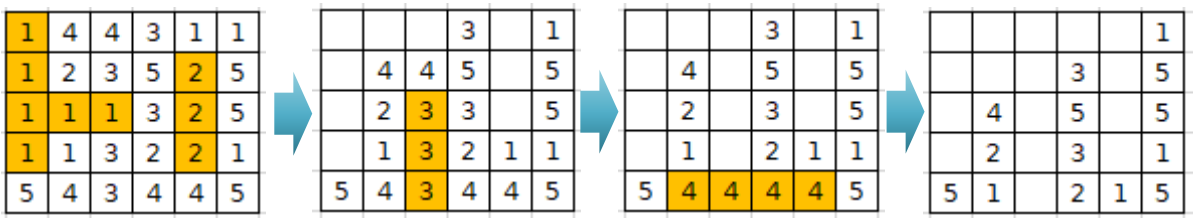
- 2 차원 배열의 선언과 사용을 익힌다.
- 함수에서 2 차원 배열을 매개변수로 사용하는 방법을 익힌다.
- 텍스트 파일 입출력을 익힌다.

(주의사항)

- 보고서는 **"assn3.docx"** 또는 "assn3.hwp"로 저장 할 것
  (기본 기능에 대한 보고서를 먼저 작성하고, 추가 기능의 목적, 구현 방법 등은 기본 기능에 대한 보고서 마지막에 1 페이지 이내로 간략히 설명할 것)
- 기본 기능만 구현한 파일 이름은 **"assn3.c"** 로 저장 할 것 **(필수 제출)**
- **추가 기능이 함께 구현된** 파일 이름은 **"assn3_add.c"** 로 저장 할 것
  (추가 기능을 구현했다면 assn3.c 와 assn3_add.c 2 개 파일 제출)
- 전역변수, goto 문, 구조체 등은 사용하지 않는다.
- 프로그램 구현 시, main() 함수를 호출하여 사용하지 않는다.
- 적절히 사용자 정의 함수를 작성하여야 한다. (main() 함수 내에 모든 기능을 구현 시 감점)
- 과제에서 요구하는 사용자 정의 함수(5 쪽 참고)는 반드시 구현하여야 하며, 이외 필요한 함수는 적절히 정의해서 사용한다.
- 문제의 출력 형식은 채점을 위해 실행예시와 최대한 비슷하게 작성해 주세요.

(문제)

이번 과제에서는 우리가 흔히 접할 수 있는 퍼즐게임(애니팡)을 간소화하여 구현하려고 합니다.



이 과제에서 구현하려고 하는 게임의 경우, 같은 무늬(숫자)가 3 개 이상 연속되는 경우 팡팡 터지게 되어 사라지고, 그 자리에는 위쪽의 숫자들이 내려와서 채우게 됩니다. 위의 그림처럼 더 이상 연속되는 무늬가 없는 경우, 게임이 끝나게 됩니다.

(퍼즐 규칙)

- 게임판은 최대 9 x 9 의 크기를 가진다. (단, 행의 수, 열의 수 >= 5)
- 무늬는 1~7 의 숫자로 7 개의 종류를 가진다.
- 가로나 세로로 3 개 이상의 같은 무늬가 연속될 경우 그 무늬들은 제거되고, 그 자리에는 위쪽의 무늬들이 순서대로 내려와서 채운다.
- 제거될 무늬가 없을 때까지 제거와 채우는 일을 반복한다.

(설명 및 요구사항)

1. 프로그램을 실행하면 "board.txt"에서 정보를 읽어 아래 그림 1 과 같이 초기 게임판을 출력한다. 파일의 첫 줄에는 예시처럼 게임판의 행의 개수와 열의 개수가 순서대로 주어지며, 그 다음 줄부터는 게임판의 무늬들이 순서대로 주어진다.
   그림 1 은 아래의 board.txt 예시를 읽어 출력한 화면이다. (글자색은 6 쪽의 "*리눅스에서 글자색 출력하기*"부분을 참고한다.)

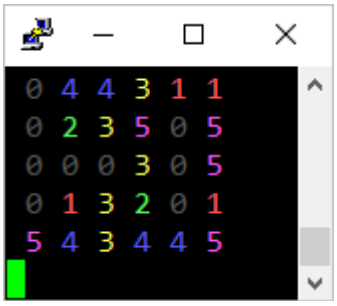
board.txt 예시
5 6
1 4 4 3 1 1
1 2 3 5 2 5
1 1 1 3 2 5
1 1 3 2 2 1
5 4 3 4 4 5

그림 1. 초기 게임판 출력화면

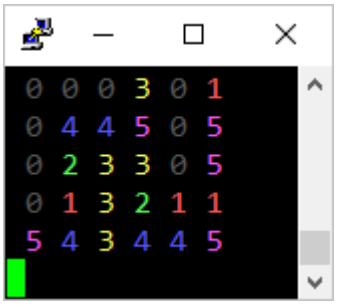
파일 예외 처리)
board.txt 파일이 없는 경우, 1 부터 7 이외의 무늬가 있는 경우, 게임판의 셀의 개수가 다른 경우 적절한 에러메시지를 출력 후, 프로그램을 종료한다.

※ 게임 판의 셀의 개수가 다른 경우: 위의 예시는 5 x 6 개의 셀을 가지는 데, 30 개 이외의 값을 가지는 경우를 말함

2. 게임판을 출력 후 사용자 입력을 기다린다. 사용자가 엔터를 입력하면 게임판에서 가로 또는 세로가 3 개 이상 연속된 부분을 찾아 0 으로 채워 제거한 후, 아래의 그림처럼 출력한다.
   모든 게임판은 출력되기 전에 system("clear")를 이용하여 화면을 먼저 지우도록 한다.
   (화면 지우기는 7 쪽의 "*화면 지우기*"부분을 참고한다.)



3. 위의 게임판 출력 후 사용자 입력을 기다린다. 사용자가 엔터를 입력하면 제거된 자리(0 으로 채워진 자리)에는 위쪽의 무늬들이 순서대로 내려와서 채운 후, 아래의 그림처럼 출력한다.
   (힌트: 아래쪽을 먼저 처리)



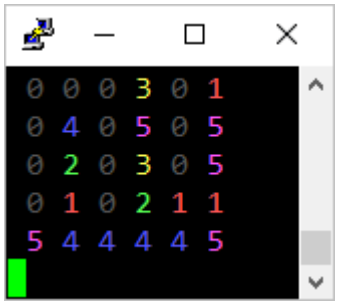
4. 더 이상 제거할 연속적인 무늬가 없을 때까지 2~3 과정을 반복한다.
   아래 그림 2 는 게임시작(a)부터 사용자가 엔터를 입력할 때마다 갱신된 화면을 순서대로 볼 수 있다.

(a)

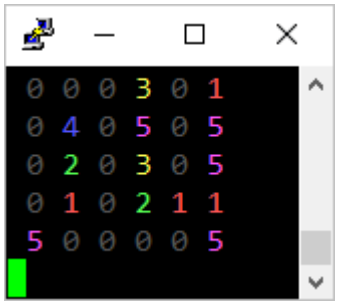
(b)

(c)

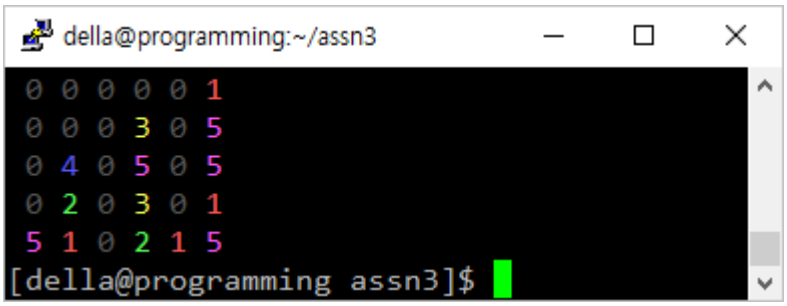
(d)



(e)



(f)



(g)

그림 2. 게임 진행 화면 예시

5. 제거할 연속적인 무늬가 없으면 프로그램을 종료한다.
   그림 2. (f)에서 연속적인 무늬가 제거되고, 사용자가 엔터를 입력하여 그림 2. (g)처럼 제거된 자리로 위쪽의 무늬가 채워진 후, 더 이상 연속적인 무늬가 없다고 판단되면 프로그램을 종료한다.

6. 게임 종료 시, 최종 화면(그림 2. (g))을 파일 result.txt 에 저장하도록 한다.

result.txt 의 예시
0 0 0 0 0 1 0 0 0 3 0 5 0 4 0 5 0 5 0 2 0 3 0 1 5 1 0 2 1 5

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.

(코드 작성)

반드시 작성해야 하는 함수는 다음과 같다. 이 함수들 말고도 자유롭게 추가로 함수를 정의해서 사용하도록 한다. (필요한 매개변수가 있으면 추가하여 구현할 것.)

- **int main()**
  프로그램이 시작되면 ReadBoard 함수를 호출해서 게임판 배열을 생성한다.
  int board[MaxArr][MaxArr] 배열로 게임판을 나타낸다. 이 배열은 그림 1 과 같은 형태로 출력할 수 있도록 게임판을 저장한다. 이 배열은 main 함수 안에서 선언해야 한다.
  게임판을 구성하기 위하여 MaxArr 를 적절한 상수로 define 한 후 사용하도록 한다**.**

- **int ReadBoard(int board[MaxArr][MaxArr], int* row, int* col)**
  board.txt 파일을 읽고 게임판의 행의 개수(row)와 열의 개수(col)를 저장한다.
  파일이 없는 경우 등에 적절한 정수를 반환하여 프로그램을 종료하는데 이용하도록 한다.

- **void PrintBoard(int board[MaxArr][MaxArr], int row, int col)**
  그림 1 과 같이 게임판을 출력하는 함수

- **int RemoveChunk(int board[MaxArr][MaxArr], int row, int col)**
  3 개 이상의 연속적인 무늬를 제거하는 함수로, 제거되는 무늬가 있는 경우 적절한 값을 반환하여 프로그램의 종료를 결정하는데 이용할 수 있음

- **void UpdateBoard(int board[MaxArr][MaxArr], int row, int col)**
  연속적인 무늬가 제거 된 경우, 제거 된 자리의 위쪽의 무늬들이 순서대로 내려와서 채우는 기능을 하는 함수

(부록)

● 리눅스에서 글자색 출력

ANSI ESC code 를 사용하여 리눅스 터미널에 출력되는 글자의 색을 바꿀 수 있다.
"<ESC>[<색상코드>m" 을 터미널에 출력하면, 이후에 출력되는 글자의 색이 색상코드에 따라 바뀐다. ESC 를 출력하기 위해서는 ASCII 코드 0x1b 를 사용하면 된다.
색상코드는 다음과 같은 것들이 있다. (참조: https://en.wikipedia.org/wiki/ANSI_escape_code)

회색	90
주황색	91
연두색	92
노란색	93
파란색	94
자주색	95
하늘색	96
흰색	97
RESET	0

이 과제에서는 편의상 0~7 에 90~97 의 코드를 순서대로 주어 출력하도록 한다.
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("[%dm", 91); 로도 쓸 수 있음
	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
#include <stdio.h>
#define RED "\x1b[31m"
#define RESET "\x1b[0m"
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.h 를 포함된 system 함수를 사용한다. 다만, 윈도우의 비주얼 스튜디오와 리눅스의 프로그래밍 환경이 다름에 따라 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. (int arr[i][j];).
- 충분한 크기로 배열을 선언 한 뒤 필요한 부분만 화면에 출력한다.
- 2차원 배열을 함수의 매개변수로 넘겨주기 위해서는 arr[][size]와 같은 표현을 사용한다.

```
#define MAX_ROW 3
#define MAX_COL 3

int arrayTestFunc(int arr[][MAX_COL], int …);
int main()
{
	int arr[MAX_ROW][MAX_COL] = {0, 1, 2, 3, 4, 5, 6, 7, 8};
	arrayTestFunc(arr, MAX_ROW);
	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.