

CSED101 Programming & Problem Solving

Fall, 2018

Programming Assignment #3

무은재 새내기학부

20180038

박형규

POVIS ID : hyeongkyu

Honor Code : 나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

1. 프로그램 개요

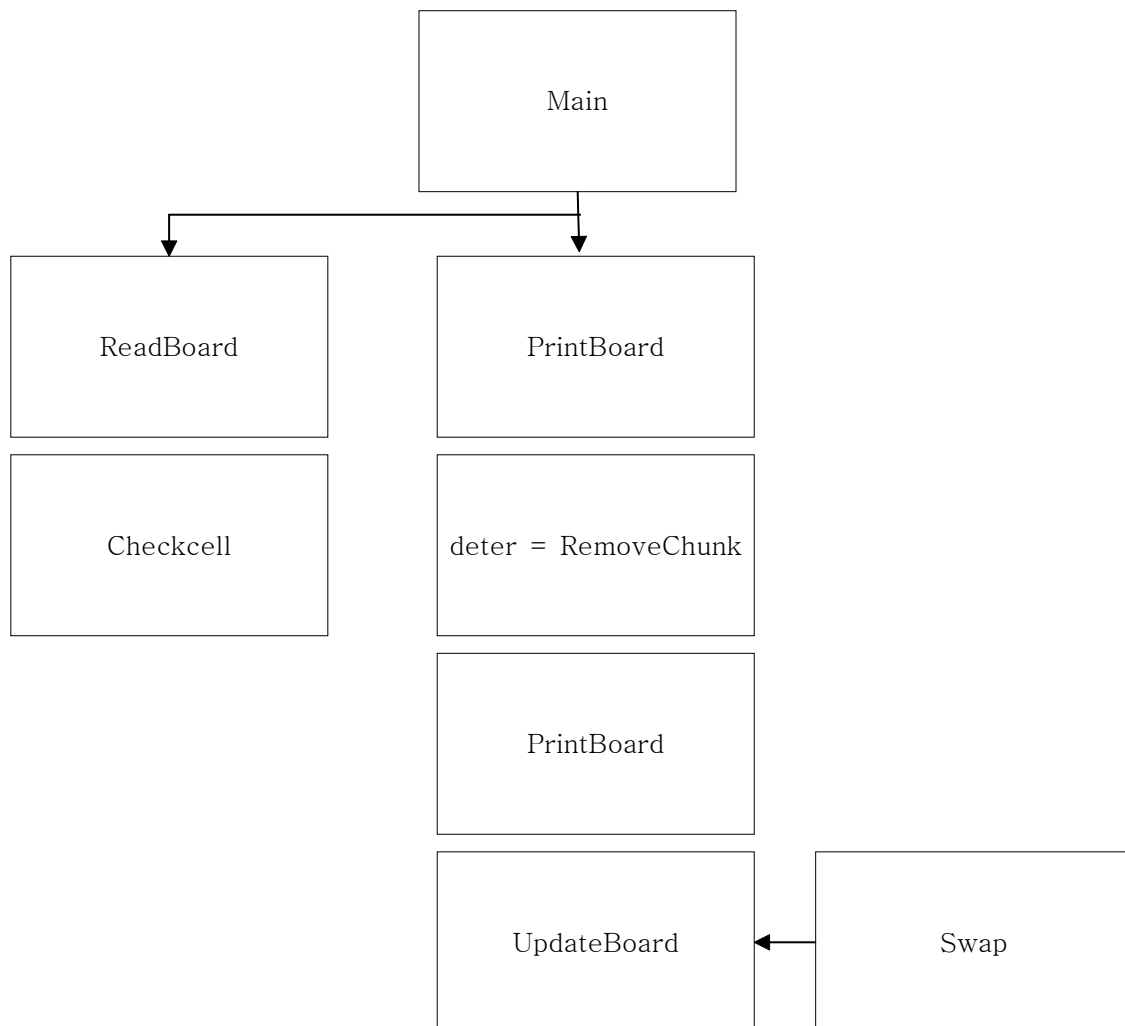
1.1 목적

이번 프로그램을 통하여 2차원 배열의 선언과 사용을 익히고, 함수에서 2차원 배열을 매개변수로 사용하는 방법을 익히며 텍스트 파일 입출력을 사용하는 법을 익힌다.

1.2 전체 구조

이번 과제는 우리가 흔히 접하는 퍼즐게임(애니팡)을 간소화하여 구현한 것으로 처음에 주어진 txt파일에서 행과 열의 값, 그리고 이를 채우는 값들을 받아서 같은 숫자가 3개 이상 연속되는 경우 그 숫자들이 0으로 바뀌고, 그 자리에는 위쪽의 숫자들이 내려와서 채우게 된다. 그리고 마지막에 더 이상 연속되는 무늬가 없는 경우에 게임이 종료되게 된다.

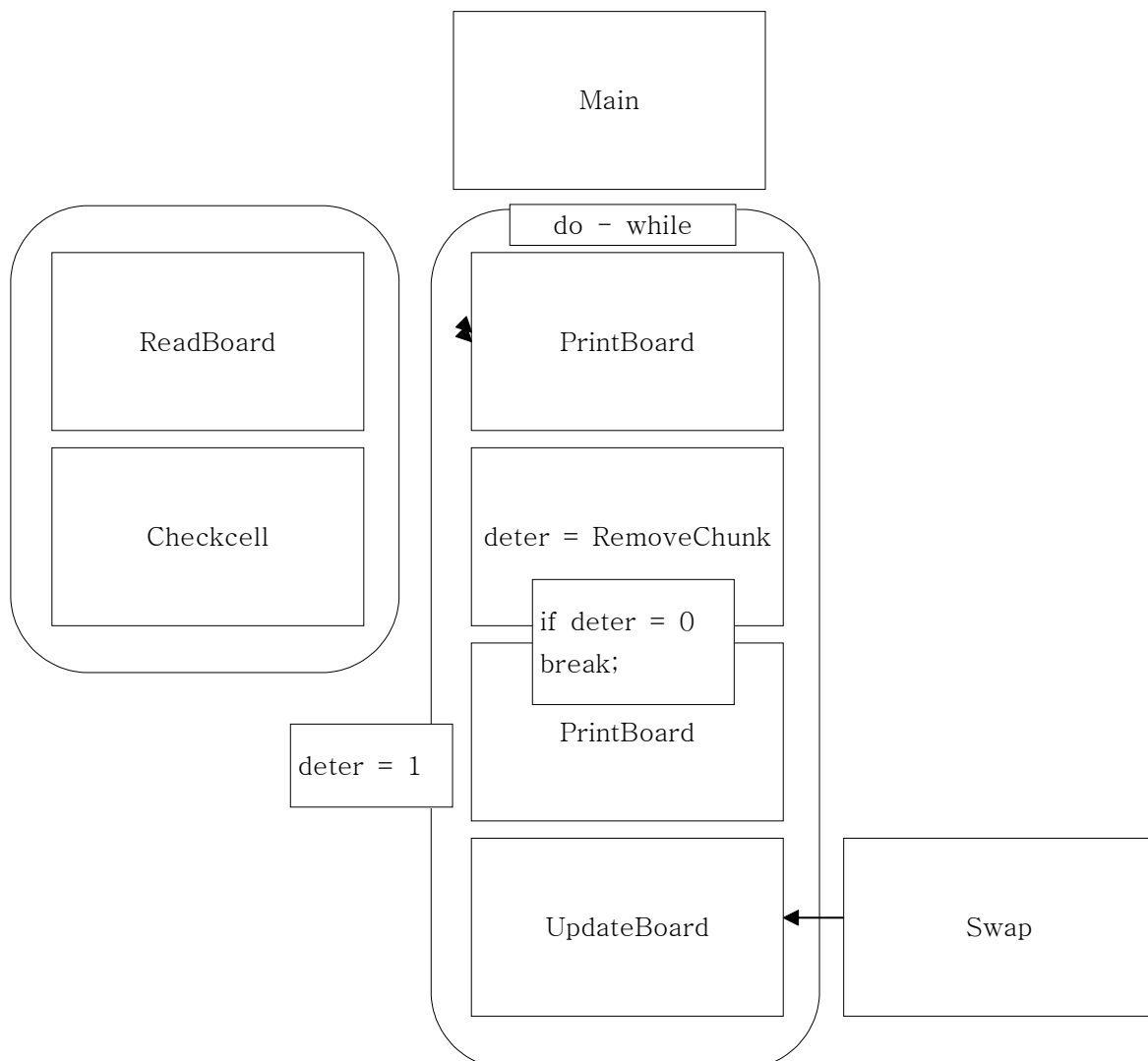
2. 전체 구조도



3. pseudocode

```
Algorithm simple pang
ReadBoard();
do{
  PrintBoard();
  deter = RemoveChunk();
  if(deter == 0)
    break;
  PrintBoard();
  UpdateBoard();
} while(deter);
outfile = fopen(result.txt);
fprintf(outfile, "%2d", board[i][j]);
```

4. flowchart



5. 프로그램 코드

5.1 assn3.c

Header file 포함

```
#include <stdio.h>
#include <stdlib.h>
#define MaxArr 9
#define GRAY "Wx1b[90m"
#define ORANGE "Wx1b[91m"
#define GREEN "Wx1b[92m"
#define YELLOW "Wx1b[93m"
#define BLUE "Wx1b[94m"
#define PURPLE "Wx1b[95m"
#define SKY "Wx1b[96m"
#define WHITE "Wx1b[97m"
#define RESET "Wx1b[0m"
```

리눅스에서 색을 나타내는 것을 간단하게 하기 위하여 #define을 사용하였다.

```
int ReadBoard(int board[MaxArr][MaxArr], int* row, int* col);
void PrintBoard(int board[MaxArr][MaxArr], int row, int col);
int RemoveChunk(int board[MaxArr][MaxArr], int sub[MaxArr][MaxArr], int row, int col);
void UpdateBoard(int board[MaxArr][MaxArr], int row, int col);
void swap(int* x, int* y);
void Checkcell(FILE *infile, int row, int col);
void Checkseq(FILE *infile, int row, int col);
```

함수를 선언하는 부분이다.

```

int board[MaxArr][MaxArr] = { 0 };
int sub[MaxArr][MaxArr];
int row, col;
int deter;
char c;
ReadBoard(board, &row, &col);
FILE *outfile;
outfile = fopen("result.txt", "w");
do //
{
system("clear");
PrintBoard(board, row, col);
getchar();

deter = RemoveChunk(board, sub, row, col);
if (deter == 0)
{
break
}

system("clear");
PrintBoard(board, row, col);
getchar();

UpdateBoard(board, row, col);
} while (deter);

```

main함수에서 ReadBoard 함수를 호출하여 주어진 보드판과 row, col의 값을 받아 오고, do- while문을 통하여 deter값이 1일 경우에는 이 반복문을 반복한다. do-while문 안에서는 PrintBoard, RemoveChunk, UpdateBoard 함수를 호출한다.

```

int i, j;
for (i = 0; i < row; i++)
{
for (j = 0; j < col; j++)
{
fprintf(outfile, "%2d", board[i][j]);
}
fprintf(outfile, "\n");
}
fclose(outfile);
return 0;

```

main함수에서 result.txt파일에 결과값을 출력하는 부분이다.

```
int i, j;
int test = 0;
FILE *infile;
if ((infile = fopen("board.txt", "r")) == NULL)
{
printf("ERROR opening board.txt\n");
exit(100);
}
fscanf(infile, "%d %d\n", row, col);
for (i = 0; i < *row i++)
{
for (j = 0; j < *col j++)
{
fscanf(infile, "%d", &board[i][j]);
if (board[i][j] - 7 < -6 || board[i][j] - 7 > 0)
{
printf("ERROR pattern\n");
exit(101);
}
}
}
Checkcell(infile, *row, *col);
Checkseq(infile, *row, *col);
```

ReadBoard 함수의 내용으로 board.txt 파일을 받아서 2차원 배열인 board에 값을 채워주는 역할을 한다. board.txt 파일이 없을 경우, 값에 1부터 7이외의 수가 있을 경우, 셀의 값이 맞지 않은 경우, row와 col이 거꾸로 생각되어 셀의 개수는 정상적이지만 2차원 배열의 모양이 거꾸로 되었을 경우에 ERROR를 띄운다.

```
int i, j;
for (i = 0; i < row i++)
{
for (j = 0; j < col j++)
{
if (board[i][j] == 0)
printf(GRAY);
else if (board[i][j] == 1)
printf(ORANGE);
else if (board[i][j] == 2)
printf(GREEN);
else if (board[i][j] == 3)
printf(YELLOW);
else if (board[i][j] == 4)
printf(BLUE);
else if (board[i][j] == 5)
printf(PURPLE);
else if (board[i][j] == 6)
printf(SKY);
else if (board[i][j] == 7)
printf(WHITE);
printf("%2d", board[i][j]);
printf(RESET);
}
printf("Wn");
}
```

PrintBoard함수의 내용으로 실제 화면에 board를 프린트 해주며 0부터 7까지의 수에 따라 리눅스 화면에서 다른 색을 나타낸다.

```
int result = 0;
int sum = 0;
int i, j;
for (i = 0; i < row i++)
for (j = 0; j < col j++)
{
sub[i][j] = board[i][j];
}

for (i = 0; i < row i++)
{
for (j = 0; j < col - 2; j++)
{
if (board[i][j] == board[i][j + 1] && board[i][j + 1] == board[i][j + 2])
{
sub[i][j] = 0;
sub[i][j + 1] = 0;
sub[i][j + 2] = 0;
result = 1;
}
}
}

for (j = 0; j < col j++)
for (i = 0; i < row - 2; i++)
{
if (board[i][j] == board[i + 1][j] && board[i + 1][j] == board[i + 2][j])
{
sub[i][j] = 0;
sub[i + 1][j] = 0;
sub[i + 2][j] = 0;
result = 1;
}
}

for (i = 0; i < row i++)
for (j = 0; j < col j++)
{
if (sub[i][j] == board[i][j])
sum++;
}

if (sum == row * col)
{
return 0;
}

for (i = 0; i < row i++)
for (j = 0; j < col j++)
{
board[i][j] = sub[i][j];
}

return result;
```


RemoveChunk 함수의 내용으로 sub라는 2차원배열을 하나 더 만들어서 board와 같은 위치에 같은 값이 있도록 만들어준 후 , 가로, 세로 각각 3개 이상 중복되는 값이 있을 경우에 sub에서 그 위치에 있는 값을 0으로 바꾸어 준다. 또한 바뀌는 값이 있을 경우 result값을 1로 해준다. 바뀌는 값이 없을 경우에는 result 값을 0으로 해주어서 return되는 값이 0으로 하여 main의 do-while문이 종료되도록 한다.

```
int i, j, num;
for (num = 0; num < row - 1; num++)
for (i = 1; i < row; i++)
for (j = 0; j < col; j++)
{
if (board[i][j] == 0)
swap(&board[i][j], &board[i - 1][j]);
}
```

UpdateBoard 함수의 내용으로 swap함수를 이용하여 0이 위쪽으로 올라가게 해준다.

```
int temp = *x
*x = *y
*y = temp;
```

swap함수의 내용으로 받아온 두 값을 바꾸어주는 역할을 한다.

```
int i, j;
int test = 0;
int arr[MaxArr][MaxArr] = { 0 };
infile = fopen("board.txt", "r");
fscanf(infile, "%d %d", &row, &col);
for (i = 0; i < MaxArr; i++)
{
for (j = 0; j < MaxArr; j++)
{
fscanf(infile, "%d", &arr[i][j]);
if (arr[i][j] != 0)
test++;
}
}
if (test != row * col)
{
printf("ERROR number of cells\n");
exit(102);
}
```

Checkcell 함수의 내용으로 셀의 개수가 맞지 않을 경우에 ERROR를 출력한다.

```

infile = fopen("board.txt", "r");
fscanf(infile, "%d %d", &row, &col);
int arr[MaxArr][MaxArr];
char ent;
int i, j;
for (i = 0; i < row i++)
{
for (j = 0; j < col j++)
{
fscanf(infile, "%d", &arr[i][j]);
}
ent = fgetc(infile);
if (ent != 'Wn')
{
printf("ERROR row and col are changedWn");
exit(103);
}
}
}


```

Checkseq함수의 내용으로 row와 col을 거꾸로 생각하여 셀의 개수는 같지만 행과 열이 뒤집어 졌을 경우에 ERROR를 출력해주는 함수이다.

6. 프로그램 실행 방법

board.txt파일에 9이하의 row값과 col 값을 입력한 후, 그에 맞는 2차원 배열을 채울 수를 써놓는다. 그 후 프로그램을 실행하면 Board가 프린트 되며 Enter키를 누르면 가로, 세로로 3개 이상의 같은 값이 있으면 그 값들이 모두 0으로 변하고 다시 Enter키를 누르면 0이 위로 올라간다. 이는 더 이상 0으로 바뀔 값이 없을 때 까지 반복되며 바뀔 값이 없으면 프로그램이 종료된다.

7. 프로그램 실행 예제



```

login as: hyeongkyu
hyeongkyu@programming.postech.ac.kr's password:
Last login: Thu Nov  8 21:13:43 2018 from 141.223.204.78
*****
Welcome to programming server
Inquiry: 279-2520, hemose@postech.ac.kr
*****
[hyeongkyu@programming ~]$

```

<리눅스 로그인>

```
1 // 3. 100 이하의 자연수 중에서 3과 5의 배수를 곱할 수 있는 최소의 자연수를 구하는 문제
2 #include <stdio.h>
3 #include <math.h>
4
5 int main()
6 {
7     int a, b, c;
8     a = 1;
9     b = 1;
10    c = 1;
11
12    while (1)
13    {
14        if (a % 3 == 0 && b % 5 == 0)
15        {
16            printf("%d\n", a * b);
17            break;
18        }
19        a++;
20        b++;
21        c++;
22    }
23
24    return 0;
25 }
```

<vim을 이용하여 프로그램 짜는 모습>

```
hyeongkyu@programing assn3$ vim assn3.c
hyeongkyu@programing assn3$ gcc assn3.c -o assn3.out
hyeongkyu@programing assn3$
```

<gcc assn3.c -o assn3.out>

```
1 4 4 3 1 1
2 2 3 2 2 2
3 1 1 3 2 2
4 3 3 2 2 1
5 4 4 4 5
```

(a)

```
0 4 4 3 1 1
1 2 3 5 1 5
2 0 0 3 0 5
3 1 3 2 0 1
4 3 4 4 5
```

(b)

```
0 0 0 3 0 1
1 4 4 5 0 5
2 2 3 3 0 5
3 1 1 3 2 1
4 3 3 2 2 1
5 4 4 4 5
```

(c)

```
0 0 0 3 0 1
1 4 4 5 0 5
2 2 0 3 0 5
3 1 0 2 1 1
4 0 4 4 5
```

(d)

```

0 0 0 3 0 1
0 0 0 3 0 5
0 4 0 5 0 5
0 2 0 3 0 5
0 1 0 2 1 1
5 4 4 4 4 5

```

(e)

```

0 0 0 3 0 1
0 0 0 3 0 5
0 4 0 5 0 5
0 2 0 3 0 5
0 1 0 2 1 1
5 0 0 0 0 5

```

(f)

```

0 0 0 3 0 1
0 0 0 3 0 5
0 4 0 5 0 5
0 2 0 3 0 5
0 1 0 2 1 1
5 4 4 4 4 5

```

(g)

```

0 0 0 3 0 1
0 0 0 3 0 5
0 4 0 5 0 5
0 2 0 3 0 5
0 1 0 2 1 1
5 1 0 2 1 5
[hyeonkyu@programming assn3]$

```

<마지막 Enter치면 게임 종료>

```

[hyeonkyu@programming assn3]$ ls
assn3.c assn3.out board.txt result.txt
[hyeonkyu@programming assn3]$ ./assn3.out
ERROR pattern
[hyeonkyu@programming assn3]$

```

<1 ~ 7이외의 수가 있을 경우>

```

[hyeonkyu@programming assn3]$ ./assn3.out
ERROR number of cells
[hyeonkyu@programming assn3]$

```

<셀의 개수가 다를 경우>

```

[hyeonkyu@programming assn3]$ ./assn3.out
ERROR row and col are changed
[hyeonkyu@programming assn3]$

```

<row와 col을 거꾸로 입력했을 경우>

8. 추가기능 구현 설명

먼저 main함수에서 do - while문을 가장 바깥쪽에 한 번 더 사용하여 사용자가 한 게임이 종료된 후 추가적인 게임을 원하면(y나 Y를 입력하면) 처음 사용자가 입력한 row와 col에 알맞은 배열에 1부터 7의 수를 랜덤으로 배열하여 추가적인 게임을 진행할 수 있도록 하였다. 또한 추가적인 게임을 원하지 않으면(n이나 N)을 입력하면 프로그램이 종료될 수 있게 하였다.

RemoveChunk 함수에서 추가기능이 구현된 부분으로 양 방향의 대각선으로도 3개 이상의 같은 숫자가 있으면 0으로 바뀌도록 구현하였다.

추가기능이 구현된 프로그램을 실행하는 방법은 먼저 board.txt파일에 9이하의 row 값과 col 값을 입력한 후, 그에 맞는 2차원 배열을 채울 수를 써놓는다. 그 후 프로그램을 실행하면 Board가 프린트 되며 Enter키를 누르면 가로, 세로, 대각선으로 3개 이상의 같은 값이 있으면 그 값들이 모두 0으로 변하고 다시 Enter키를 누르면 0이 위로 올라간다. 이는 더 이상 0으로 바뀔 값이 없을 때 까지 반복되며 바뀔 값이 없으면 게임이 종료된 후 게임 추가 진행 여부를 물어보는 문구가 나온다. 여기서 Y나 y를 입력하면 다음 게임으로 처음 입력한 row와 col 값에 알맞은 배열에 1부터 7사이의 수가 랜덤으로 입력되어 나오며 다시 게임을 진행하게 되고, N이나 n을 입력하면 프로그램이 종료되게 된다.

9. 토론 및 결론

이번 assignment를 통하여 2차원 배열이라는 어려운 개념을 제대로 이해하는데 많은 도움이 된 것 같다. 또한 앞에서 배운 파일 입출력 부분도 다시한번 다뤄보며 잊어버리지 않게 되었고, 반복문, 조건문, 사용자 함수 생성 등 앞에서 배운 여러 가지 개념들도 적용시켜보며 지금까지 배운 개념들을 다잡는 계기가 된 것 같다. 또한 이번 assignment에서는 처음으로 추가 기능 구현에 대한 점수가 부여된다고 하였다. 그래서 기본적으로 제시된 기능 이외에 내가 더 생각하여 추가적인 기능을 구현해 보았는데, 주어진 문제에 어떤 기능을 부여할 수 있는지를 생각하는 것 자체도 쉽지 않은 일이라는 것을 느꼈다. 또한 실제 그런 행위를 해보고 나니 이전에 프로그래밍에 대해 아무것도 모르던 나와 현재 많이 발전된 모습을 비교해보며 프로그래밍에 대한 자신감과 열정을 가지게 되는 계기도 된 것 같다.