2048 보고서

과목명 : 기초 공학 설계

담당 : 최명환 교수님

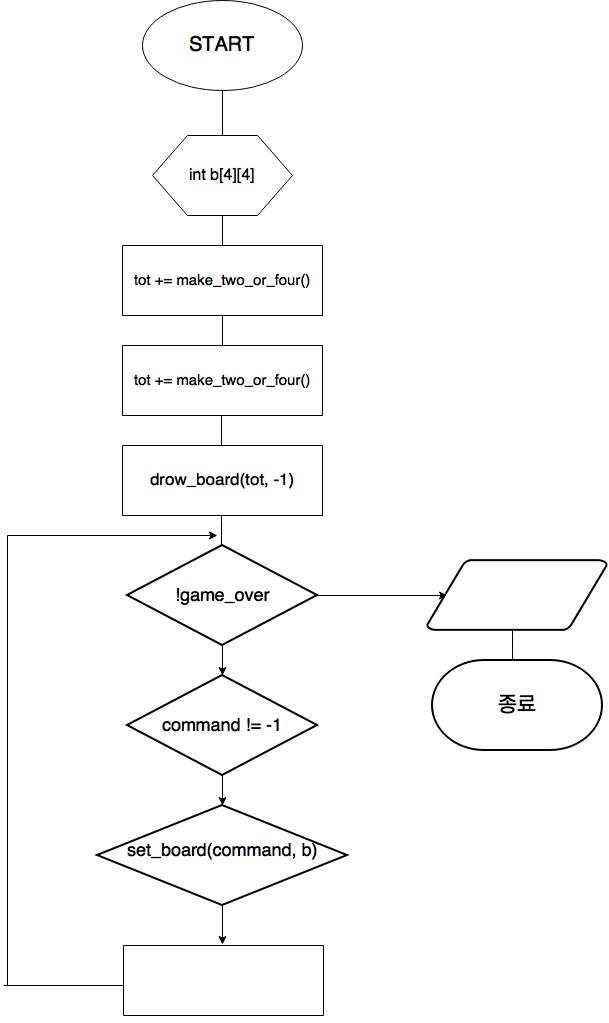
제출일 : 2017. 06. 13

서강대학교 컴퓨터 공학과

1학년 학번 : 20171653

성명 : 신나현

\*흐름도



1)int make\_two\_or\_four()

2가 나올 확률은 2/3이고 4가 나올 확률은 1/3이므로 make\_two\_or\_four함수에서 rand() 함수로부터 난수를 받아 3으로 나눈 나머지가 1일 때는 4가, 2나 0일 때는 2가 나오게 한다. 이 때 숫자가 들어갈 빈 칸이 있는지 for문을 이용해 확인한 후 빈칸이 있으면 빈칸에 들어갈 숫자인 2또는 4를 return해주고, 빈칸이 없을 때에는 0을 return해준다.

2)int GetCommand(void)

switch case를 이용해

‘d’나 ‘D’를 입력받았을 때는 0을 return 해준다.

‘w’나 ‘W’를 입력받았을 때는 1을 return 해준다.

‘a’나 ‘A’를 입력받았을 때는 2를 return 해준다.

‘s’나 ‘S’를 입력받았을 때는 3을 return 해준다.

그 외의 문자를 입력받았을 때는 –1을 return 해준다.

3)int rotate(int b[4][4])

시계방향으로 90도 회전하면 b[i][j]에 b[3-j][i]의 값을 저장해 주어야 한다.

그냥 저장을 해주면 중간에 값이 사라지므로, 정수 배열 a[4][4]를 선언해준다.

그런 뒤 이중 for문을 이용해 a[i][j] = b[3-j][i]의 식을 loop에 돌려 배열 a[4][4]를 완성해주고 이를 다시 for문을 이용해 배열 b[4][4]에 저장해준다.

4)int getch(void)

사용자로부터 입력받은 값을 반환해준다.

5)int set\_board(int dir, int b[4][4])

board를 이동시키기 전에 배열 int temp[4][4]를 선언한 뒤 이중for문을 이용해 배열b[4][4]의 내용을 그대로 저장해 놓는다.

그런 뒤, b[4][4]를 이동시킨다.

dir이 0일 때는 board를 오른쪽으로 이동시킨다.

dir이 1일 때는 board를 위쪽으로 이동시킨다.

dir이 2일 때는 board를 왼쪽으로 이동시킨다.

dir이 3일 때는 board를 아래로 이동시킨다.

오른쪽으로 이동시키는 코드만 짠 뒤, int rotate(b[4][4])함수를 이용한다.

오른쪽으로 이동시키는 함수에서는 for문을 이용해 배열에 0이 저장되어있을 때 왼쪽의 숫자를 한 칸씩 왼쪽으로 밀어주는 식을 3번 반복해 준다.

2 0 0 0 일 경우 한번 밀어주면 0 2 0 0 이 되고, 두 번 밀어주면 0 0 2 0 이 되고, 세 번 밀어주어야 0 0 0 2가 되므로 3번 밀어주어야 한다.

그런 뒤

for (i = 0; i < 4; i++)

{

for (j = 3; j > 0; j--)

{

if (b[i][j] == b[i][j - 1])

{

b[i][j] = b[i][j] + b[i][j - 1];

for (k = j - 1; k > 0; k--)

{

b[i][k] = b[i][k - 1];

}

b[i][0] = 0;

}

}

}

이 코드를 이용해 조건문에서 왼쪽의 수와 같은지 확인하고 같다면 더해주어 오른쪽으로 당겨준다.

오른쪽으로 board를 이동시키는 코드만 짠 뒤에,

왼쪽, 위, 아래로 이동하는 코드는 rotate(b)를 이용하는 횟수를 다르게 해가면서 실행한다.

위로 이동시킬 때에는 rotate(b)를 1번 사용한 뒤 오른쪽으로 이동하는 코드를 쓰고 다시 rotate(b)를 3번 사용한다.

왼쪽으로 이동시킬 때에는 rotate(b)를 2번 사용한 뒤 오른쪽으로 이동하는 코드를 쓰고 다시 rotate(b)를 2번 사용한다.

아래로 이동시킬 때에는 rotate(b)를 3번 사용한 뒤 오른쪽으로 이동하는 코드를 쓰고 다시 rotate(b)를 1번 사용한다.

그런 뒤, 이중 for문을 이용해 temp[4][4]와 b[4][4]를 비교하여 서로 다른 부분이 하나라도 있으면 1을 return해주고, 없을 때에는 0을 return해준다.

6)void draw\_board(int tot, int command)

함수가 호출 될 때 4x4 화면을 새로 그려준다.

이중for문을 이용해 board위에 가장 큰 숫자를 찾아 score값에 대입한다.

board위의 모든 숫자의 합을 구해 tot값과 비교한다.

이 때 두 값이 같지 않으면 error를 출력한다.

또한 4x4판이 변화 될 때마다 output.txt에 점수, 키, 판 정보를 출력한다.

7)int is\_game\_over()

2 또는 4를 생성하는지는 고려하지 않으므로 board를 이동시킬 수 있는가 없는 가로 game over를 판정해야 한다.

배열 int temp[4][4]를 선언해 준 뒤 이중 for문을 이용해 배열 b[4][4]의 내용을 그대로 저장해준다.

정수 a=0 과 k를 선언해 준 뒤 while문을 이용해 k값이 0일 때부터 3일 때까지 loop를 돌린다.

a += set\_board(k, temp);

k가 0 일 때는 오른쪽으로 이동

k가 1일 때는 위로 이동

k가 2일 때는 왼쪽으로 이동

k가 3일 때는 아래로 이동

(이 식으로 loop를 돌릴 때 배열b[4][4]는 변화가 없어야하기 때문에

이 배열 temp를 선언해 while문에 사용한 것이다,)

이 때 set\_board함수에서 정상적인 이동이 일어나지 않았을 경우 0을 return해주기 때문에 board가 더 이상 움직일 수 없는 경우, k가 0일 때부터 3일 때까지 loop의 결과로 나온 a값이 0을 유지한다.

따라서 a의 값이 0인 경우에는 game over이므로 1을 return해주고

a의 값이 그 외의 숫자인 경우 0을 return해준다.

8) int main(void)

output.txt를 쓰기 형식으로 실행한다.

Init()함수를 호출해 board위의 숫자를 모두 0으로 만들어 준다.

srand(time(NULL)); 이 함수를 이용해서 rand함수의 초기값을 변화시켜준다.

정수 tot을 선언하고 초기값을 0으로 설정해준다.

tot 값은 make\_two\_or\_four함수에서 생성되는 2또는 4의 값의 합이다.

게임이 시작 될 때 두개의 숫자가 board에 생성되므로, for문을 이용해

tot += make\_two\_or\_four(); 의 코드를 이용해 함수를 두 번 호출해 준다.

drow\_board(tot, -1)을 호출해 tot값과, board위의 모든 숫자의 합이 같은지 확인하고

board위의 가장 큰 수인 score를 구한다.

이 함수에서 output.txt에 점수, 키, 판 정보를 출력한다.

게임이 실행되는 동안 계속해서 2나 4의 숫자를 생성해주고 입력키를 받아야하기 때문에 !game\_over인 동안 loop를 돌린다.

loop안에서

command = GetCommand(); 코드로 함수를 호출하여 입력 받은 입력키를 -1~3의 정수로 바꿔준다.

command가 -1이 아닌 경우에 set\_board(command, b)의 return값이 1일 때만

tot += make\_two\_or\_four(); 로 함수를 호출해주고,

draw\_ board(tot, command)함수를 실행해 준다.

이때 is\_game\_over()함수를 호출해 더 이상 board가 이동할 수 없을 때 1을 return 받아

game\_over는 1의 값을 가지게 되어 !game\_over는 0이 되므로 loop를 빠져 나오게 된다.

System(“clear”) 함수를 호출해 board를 모두 지워준뒤

“game over!”를 출력해준다.

마지막으로 아까 열었던 output.txt파일을 닫아준다.