변수 game_over

배열 b[4][4] 선언.

파일 스트림 선언.

변수 command =-1 선언

파일 열기.

game_over = 0

b의 16칸 모두 0으로 설정.

Srand(time(NULL))을 통해

난수를 시간값에 따라 생성하게 하여

무작위로 난수를 생성할 수 있도록 함.

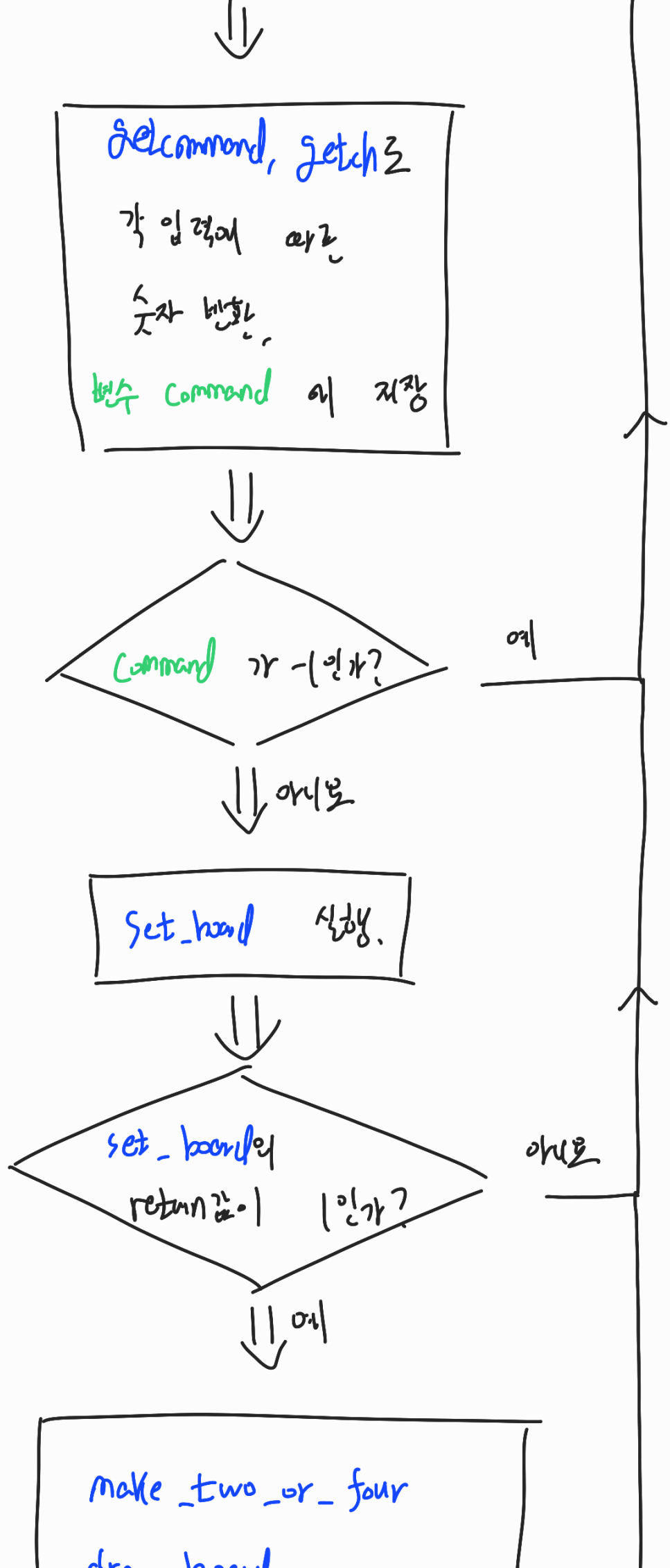
변수 tot=0 선언.

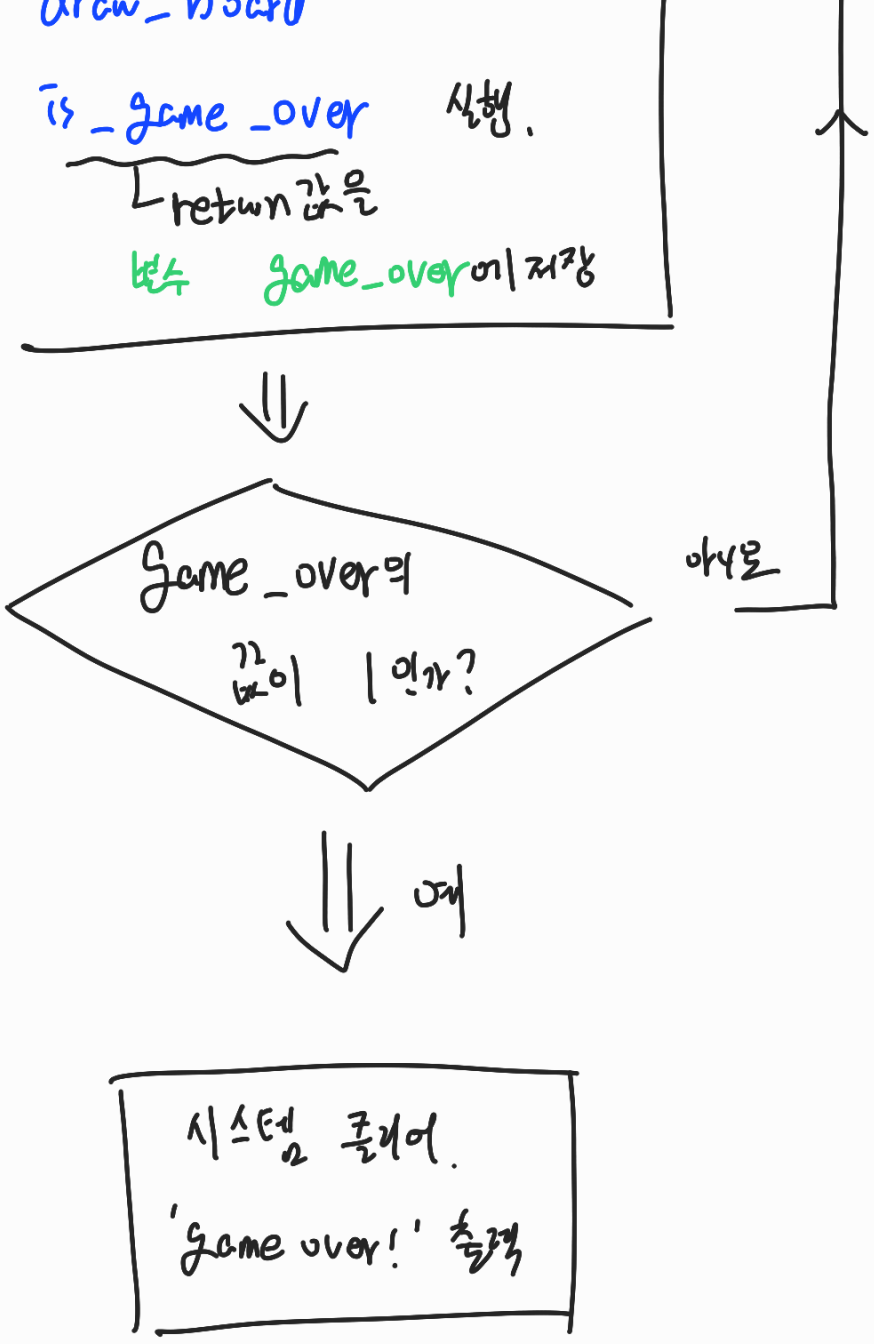
tot += make_two_or_four

두 번 실행해서 처음에 2개 만들기.

draw_board 실행

wasd 입력


GetCommand, getch로
각 입력에 따른
숫자 반환,
변수 command 에 저장
command 가 -1인가?
예
아니오
set_board 실행.
set_board의
return값이 1인가?
아니오
예
make_two_or_four



함수에 대한 설명

rotate : 배열을 시계방향으로 90도 돌려줌.
set_board에서 사용.

set_board : wasd 누름에 따라 배열을 이동시켜줌.

배열 a를 만들고 배열 b의 값들을 똑같이 배열 a에 넣음.
(이는 후에 배열의 이동이 정상적인 이동인가 판단하기 위함)

wasd 중 무엇을 눌렀냐에 따라 적절한 만큼 rorate를 사용해서 배열 b를 회전시킨 후,
왼쪽으로 한 칸씩 쭉 밀어주고,
왼쪽 칸과 오른쪽 칸의 값이 같은지 두 칸씩 확인하면서 만약 같으면 합쳐주고,
또 다시 왼쪽으로 쭉 밀어주고,
다시 rorate 사용해 원래대로 되돌려줘서
배열 이동시키기.

만약 이동시킨 후의 배열 b와 이동하기 전의 배열 b를 저장해둔 배열 a가 같다면,
즉 이동 전과 이동 후에 변화가 없다면 return 0
한 칸이라도 변화가 있다면 return 1

is_game_over : gameover인지 판단.

배열 b를 복사하여 배열 a, c, d, e에 저장.
배열 a, c, d, e를 각각 w, a, s, d가 입력된 경우를 가정하여
set_board 해보고 return값을 확인.
네 경우 다 return값이 0이면 어떻게 움직이든
이동 전과 이동 후에 배열에 차이가 없다는 뜻.

움직일 수 없다는 뜻. return 1을 해줌

네 경우 중 한 경우라도 set_board의 return값이 1이 나오면
움직일 수 있는 경우가 한 경우라도 있다는 뜻이므로
아직 이동시킬 수 있다는 뜻. return 0을 해줌.

make_two_or_four : 빈칸이 있나 확인하고 2 혹은 4 무작위로 생성.

b의 각 칸 확인. 만약 값이 0인 칸이 하나도 없다면
새로운 숫자 생성하지 않고 return 0

만약 빈 칸이 하나라도 있다면,
난수 하나 생성한 뒤, 그 난수를 3으로 나눠서
나머지가 0이면 4, 나머지가 1이거나 2이면 2 생성.

반복문을 사용해 난수 두 개를 계속 생성하고,
그 난수 두 개를 4로 나눔. 그 나머지를 배열의 행, 열 번호로 지정.
해당 행, 열의 숫자(예를 들면 1행 2열의 숫자)가 0인지 확인하고,
0이면 위에서 생성한 2 또는 4 삽입.
0이 아니면 0인 칸이 나올 때까지 반복.

생성한 2 또는 4를 return하고, tot에 더해줌.

draw_board : 보드 합 올바른지 확인하고 보드 그리기.

score = 0 선언,
배열 b의 16칸 숫자 하나하나 score에 더해줘서 합치기.
만약 score 과 tot의 값이 다르다면 board 새로 그리지 않음.

score과 tot의 값이 같다면
새로운 4X4 화면 그려주고, output.txt에 점수, 키, 판 정보 출력.