

Block Stack

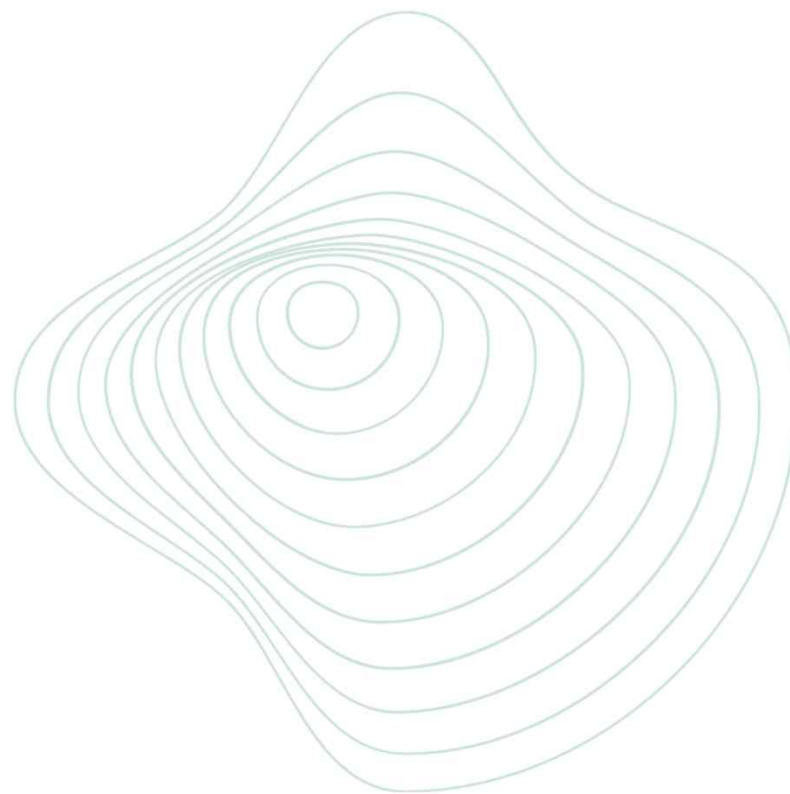
예제 8_6_1

학번 : 2021864039

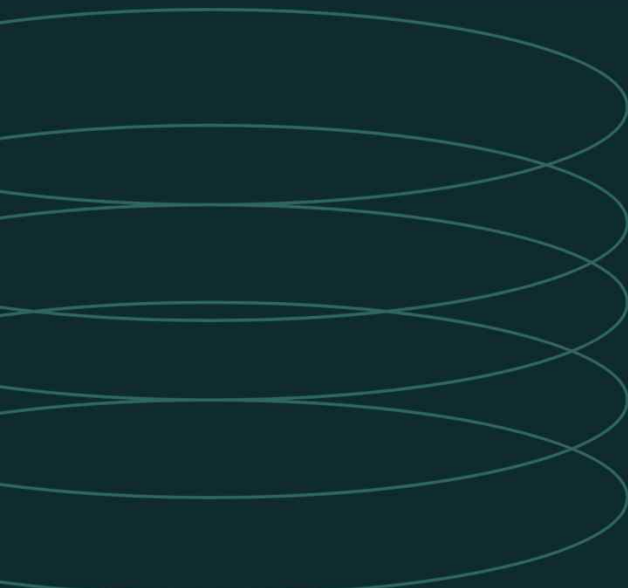
이름 : 김태헌

목차

- 프로그램 목표
- 코드 설명
- 실행 화면
- 개선점



프로그램 목표

- 
1. 블록이 좌우로 움직임
 2. 스페이스키를 누르면 블록이 아래로 떨어짐
 3. 떨어진 블록은 쌓임
 4. 우측에 블록의 개수와 시도한 횟수, 쌓인 블록 수가 카운팅 됨
 5. 블록의 개수만큼 실행이 되면 종료 후 메시지 출력

main

intro_game

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <windows.h>

#define box_length 15 //게임의 영역(좌우 길이)
#define box_height 15 //바닥의 높이(상하 길이)

void intro_game(void);
void game_control(void);
void gotoxy(int x, int y);
int left_right_move(void);
void move_down(int x);
void draw_rectangle(int c, int r);
int max_block(void);

int block_stack[box_length*2+1]={0}; //해당위치의 값을 0으로
초기화

int main(void)
{
    intro_game();
    game_control();
    gotoxy(1, box_height+3);
    printf("game이 종료되었습니다.      \n");
    return 0;
}
```

```
void intro_game(void)
{
    system("cls");
    printf("블록 쌓기 \n\n");
    printf("블록이 좌우로 움직일때 스페이스키를 누르면 \n");
    printf("블록이 떨어져 바닥에 쌓입니다.\n\n");
    printf("아무키나 누르면 게임을 시작합니다. \n");
    getch();
}
```

game_control

gotoxy

```
void game_control(void)
{
    int x, count=0;
    system("cls");
    draw_rectangle(box_length, box_height);
    gotoxy(box_length*2+5,3);
    printf("블록의 개수: %2d", box_height);
    gotoxy(1, box_height+3);
    printf("스페이스키를 누르면 블록이 떨어지고
    \n");
    printf("바닥에 쌓입니다. \n");
    while(count<box_height)
    {
        gotoxy(box_length*2+5,4);
        printf("시도한 횟수: %2d", count+1);
        gotoxy(box_length*2+5,5);
        printf("쌓인 블록수: %2d", max_block());
        x=left_right_move();
        move_down(x);
        count++;
        getch();
    }
```

```
void gotoxy(int x, int y)
{
    COORD Pos = {x - 1, y - 1};
```

```
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),
    Pos);
}
```

COORD pos = x와 y를 가지고 있는 구조

SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),pos

= 콘솔의 위치를 알려주는 함수

GetStdHandle = 화면출력 핸들러

COORD 타입 변수 pos

left_right_move move_down

Getch와 kbhit
Kbhit 함수는 키보드의 입력 여부를 단순히 입력
버퍼만 확인하고 true랑 false값 리턴
단순히 버퍼만 확인하기 때문에 값이 입력되고 난 뒤
비우지 않으면 계속 true값을 리턴함

Getch 함수는 버퍼에 있는 값을 그대로 출력하고
버퍼를 비워줌

```
int left_right_move(void)
{
    int x=3, y=2, temp=2;
    do
    {
        x+=temp;
        if (x>(box_length*2)) //x방향 최대값 설정
            temp=-2;
        if (x<3)
        {
            x=3;
            temp=2;
        }
        gotoxy(x, y);
        printf("□");
        Sleep(50); //블록이 좌우로 움직이는 속도를
        조절
        gotoxy(x, y);
        printf(" ");

    }while(!kbhit());
    block_stack[x] +=1;
    return x;
}
```

```
void move_down(int x)
{
    int y;
    for(y=2;y<(box_height+2-
        block_stack[x];y+=1)
    {
        gotoxy(x, y);
        printf("□");
        Sleep(20);
        gotoxy(x, y);
        printf(" ");
        Sleep(10);
    }
    gotoxy(x,box_height+2-block_stack[x]);
    printf("□");
}
```

draw_rectangle

max_block

```
void draw_rectangle(int c, int r)
{
    int i, j;
    unsigned char a=0xa6;
    unsigned char b[7];
    for(i=1;i<7;i++)
        b[i]=0xa0+i;

    printf("%c%c",a, b[3]);
    for(i=0;i<c*2+1;i++)
        printf("%c%c", a, b[1]);
    printf("%c%c", a, b[4]);
    printf("\n");

    for(i=0;i<r;i++)
    {
        printf("%c%c", a, b[2]);
        for(j=0;j<c*2+1;j++)
            printf(" ");
        printf("%c%c",a, b[2]);
        printf("\n");
    }
    printf("%c%c", a, b[6]);
    for(i=0;i<c*2+1;i++)
        printf("%c%c", a, b[1]);
    printf("%c%c", a, b[5]);
    printf("\n");
}
```

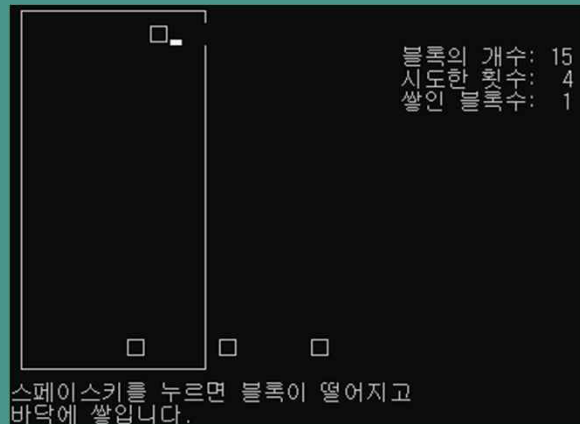
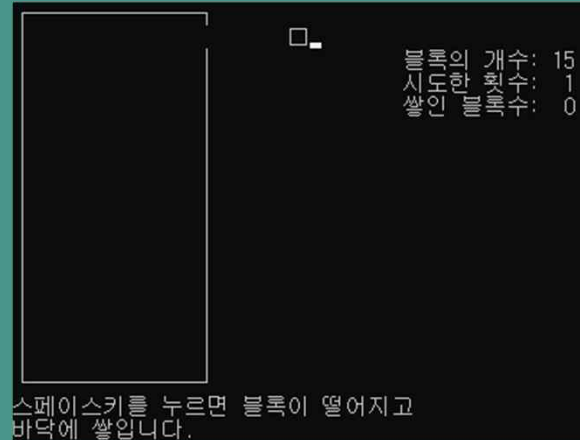
```
int max_block(void)
{
    int i, max=0;

    for(i=1;i<box_height*2+1;i++)
    {
        if (max<=block_stack[i])
            max=block_stack[i];
    }
    return max;
}
```

변경 전

```
void draw_rectangle(int c, int r)
{
    int i, j;
    unsigned char a=0xa6;
    unsigned char b[7];
    for(i=1;i<7;i++)
        b[i]=0xa0+i;

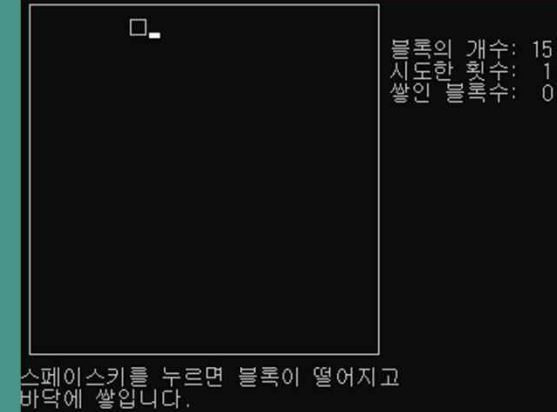
    printf("%c%c",a, b[3]);
    for(i=0;i<c;i++)
        printf("%c%c", a, b[1]);
    printf("%c%c", a, b[4]);
    printf("\n");
    for(i=0;i<r;i++)
    {
        printf("%c%c", a, b[2]);
        for(j=0;j<c;j++)
            printf(" ");
        printf("%c%c",a, b[2]);
        printf("\n");
    }
    printf("%c%c", a, b[6]);
    for(i=0;i<c;i++)
        printf("%c%c", a, b[1]);
    printf("%c%c", a, b[5]);
    printf("\n");
}
```



변경 후

```
void draw_rectangle(int c, int r)
{
    int i, j;
    unsigned char a=0xa6;
    unsigned char b[7];
    for(i=1;i<7;i++)
        b[i]=0xa0+i;

    printf("%c%c",a, b[3]);
    for(i=0;i<c*2+1;i++)
        printf("%c%c", a, b[1]);
    printf("%c%c", a, b[4]);
    printf("\n");
    for(i=0;i<r;i++)
    {
        printf("%c%c", a, b[2]);
        for(j=0;j<c*2+1;j++)
            printf(" ");
        printf("%c%c",a, b[2]);
        printf("\n");
    }
    printf("%c%c", a, b[6]);
    for(i=0;i<c*2+1;i++)
        printf("%c%c", a, b[1]);
    printf("%c%c", a, b[5]);
    printf("\n");
}
```



실행 화면


INTRO

블록 쌓기

블록이 좌우로 움직일때 스페이스키를 누르면
블록이 떨어져 바닥에 쌓입니다.

아무키나 누르면 게임을 시작합니다.

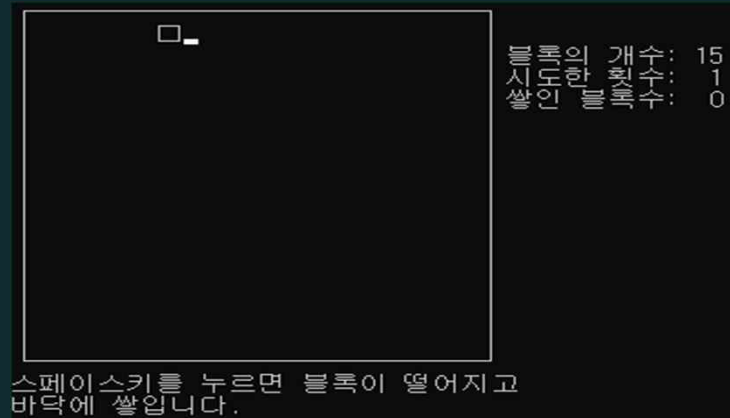
플레이 화면 2



블록의 개수: 15
시도한 횟수: 10
쌓인 블록수: 3

스페이스키를 누르면 블록이 떨어지고
바닥에 쌓입니다.

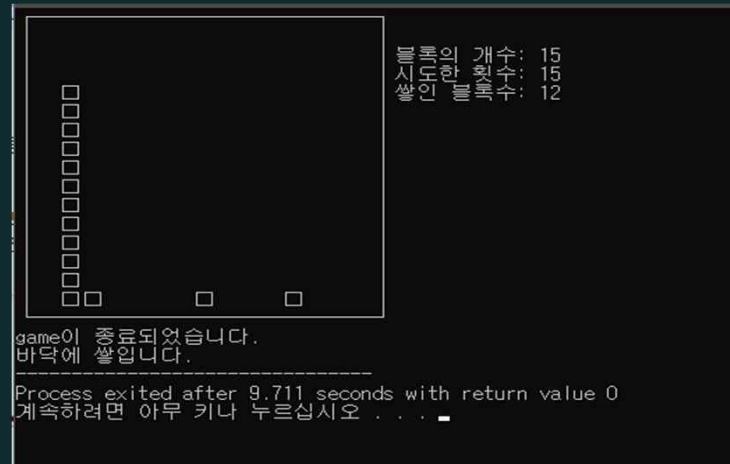
플레이 화면 1



블록의 개수: 15
시도한 횟수: 1
쌓인 블록수: 0

스페이스키를 누르면 블록이 떨어지고
바닥에 쌓입니다.

END



블록의 개수: 15
시도한 횟수: 15
쌓인 블록수: 12

game이 종료되었습니다.
바닥에 쌓입니다.

Process exited after 9.711 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .

개선점

- 블록의 개수 선택가능

현재 15개로 고정되어있는 블록의 갯수를 입력받은 값만큼 횟수를 지정함

- 난이도 조절

sleep의 값으로 블록의 움직임 속도를 조절

sleep의 값을 직접 입력 받음

sleep의 값을 switch를 이용하여 난이도 조절을 함