C++ Programming	Student number	21300691
Homework 1	Name	Cheung, Won Sik

1. Problem Definition

I want to make a brick-breaking game. However, I do not have enough time during the semester, so I want to write a ball-bouncing program. If you just bounce the ball is not fun, add another function.

2. Design of your program

There are three options. First, it is an option to draw bricks on maps in the game. Secondly, there is a brick in the place where the ball passes after playing the game. Finally, erase all the bricks on the map.

3. Primary codes with comments

Main function draw map and give 3 options to player. If player enter ESC than program end.

```
int main(){
    char ch = 0;
    char screen[24][80];

    // initialize screen array
    clrscr();
    initMap(screen);
    markBoundary(screen);

    // show 3 options and activate
    do{
        switch(ch){

        // option1: draw map
        case '1':
            clrscr();
            mapCanvas(screen);
            break;
        // option2: bouncing ball
        case '2':
            clrscr();
            bouncingBall(10, 10, 2, 1, screen);
        // option3: clear screen array
        case '3':
            clrscr();
            initMap(screen);
            markBoundary(screen);
        }

        //show options
        clrscr();
        printOption();
        ch = getchar();

    }while(ch!=27);

    clrscr();
    gotoxy(1,1);
    system("pause");
    return 0;
}
```

gotoxy and clrscr

these functions allow move position where I want to go and erase entire screen.

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

    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
}

// clear screen
void clrscr(void){
    COORD Cur= {0, 0};
    unsigned long dwLen;
    FillConsoleOutputCharacter(GetStdHandle(STD_OUTPUT_HANDLE) , ' ', 80*25, Cur, &dwLen);
    Cur = {10, 12};
    FillConsoleOutputCharacter(GetStdHandle(STD_OUTPUT_HANDLE) , ' ', 80*25, Cur, &dwLen);
}
```

mapCanvas

this function allow draw block in screen

Initialize part

```
// draw map by block before start game
void mapCanvas(char map[24][80]){
    int key = 0;
    int x = 0, y = 0;
    int oldx = 0, oldy = 0;
    int mode = MODE_MOVE;
    int oldMode = -1;
    char *message = "1: Move, 2: Draw, 3: Erase, 4: Reverse  ||  i: up, j: left, l: right, k: down ";

    // initialize part
    displayMap(map);

    gotoxy(2, 2);
    printf("Map Canvas ");
    gotoxyAdjust(1, BOTTOM_BOUND+1, ADJUST_X, ADJUST_Y);
    printf(message);

    gotoxyAdjust(1, BOTTOM_BOUND+2, ADJUST_X, ADJUST_Y);
    printf("Press ESC to go next");

    oldx = x = (LEFT_BOUND + RIGHT_BOUND) / 2;
    oldy = y = (TOP_BOUND + BOTTOM_BOUND) / 2;

    // draw part
    do {
        gotoxyAdjust(x, y, ADJUST_X, ADJUST_Y);
        putchar('*');

        if(oldx != x || oldy != y)
            displayXY(oldx, oldy, map);

         key = getch();

        oldx = x;
        oldy = y;

        oldMode = mode;
```

Draw part

```
// draw part
do {
    gotoxyAdjust(x, y, ADJUST_X, ADJUST_Y);
    putchar('*');

    if(oldx != x || oldy != y)
        displayXY(oldx, oldy, map);

     key = getch();

    oldx = x;
    oldy = y;

    oldMode = mode;

    switch(key){
    // move cursor
    case 'j':
        if(x - 1 > LEFT_BOUND)
            x--;
        break;
    case 'l':
        if(x + 1 < RIGHT_BOUND)
            x++;
        break;
     case 'i':
        if(y - 1 > TOP_BOUND)
            y--;
        break;
     case 'k':
        if(y + 1 < BOTTOM_BOUND)
            y++;
        break;
```

```
        // change drawing mode
         case '1':
            mode = MODE_MOVE;
            break;
         case '2':
            mode = MODE_DRAW;
            break;
         case '3':
            mode = MODE_ERASE;
            break;
         case '4':
            mode = MODE_REVERSE;
            break;
         }

         markXY(x, y, mode, map);
    // while loop break when player enter ESC
    } while (key != 27);

    clrscr();
    gotoxy(1, 1);
}
```

Bouncing Ball

This function bounces the ball and draws the block where the ball passes.

```
// Bouncing ball appear and continue bounce and draw block
void bouncingBall(int sx, int sy, int dx, int dy, char map[24][80])
{
    // sx, sy : first position of x, y
    // dx, dy : x, y 's speed

    // x,y : current position
    // oldx, oldy : where x, y passess position
    // fdx, fdy : real speed of x, y
    // max : speed control variable. The smaller the speed, the faster.
    // n : clock variable when n%max == 0 cursor move
    float x = sx, y = sy;
    float oldx = 0, oldy = 0;
    float fdx = 0., fdy = 0.;
    int max = dx;

    char key = 0;
    int n = 0;

    // set fdx, fdy, max
    if(dx != 0 && absolute(dx) >= absolute(dy)){
        fdx = (dx > 0 ? 1. : -1.);
        fdy = dy / (float)absolute(dx);
    } else if(dy != 0){
        fdx = dx / (float)absolute(dy);
        fdy = (dy > 0 ? 1. : -1.);
        max = dy;
    } else {
        fdx = dx;
        fdy = dy;
    }

    // for safety
    if(max <= 0)
        max = 1;

    // initialize part
    displayMap(map);

    gotoxy(2, 2);
    printf("Bouncing Ball ");
    gotoxyAdjust(1, BOTTOM_BOUND + 1, ADJUST_X, ADJUST_Y);
    printf("Press ESC to go next");

    x = oldx = sx;
    y = oldy = sy;
```

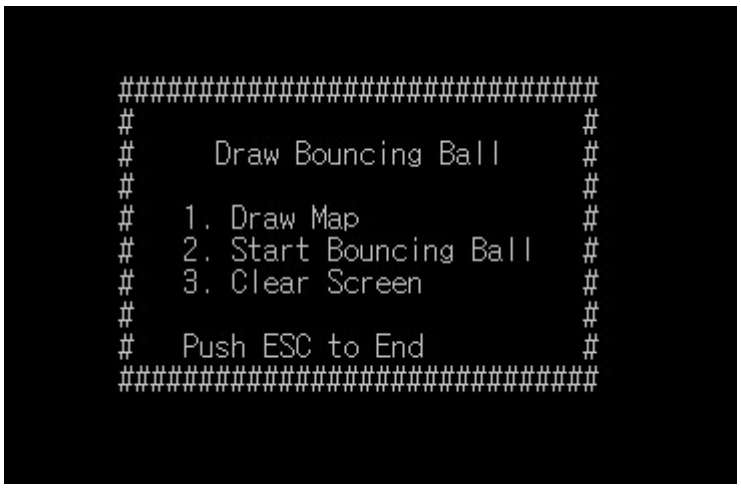
```
do {
    if(n % max == 0){
        // display ball
        gotoxyAdjust((int)x, (int)y, ADJUST_X, ADJUST_Y);
        putchar('*');

        // erase previous ball
        if(x != oldx || y != oldy){
            markXY(oldx, oldy, MODE_DRAW, map);
            gotoxyAdjust((int)oldx, (int)oldy, ADJUST_X, ADJUST_Y);
            putchar('#');
        }
    }

    Sleep(100 / max);

    // save current position
    if(n % max == 0){
        oldx = x;
        oldy = y;
    }

    x += fdx;
    y += fdy;
```

Check when ball meet wall

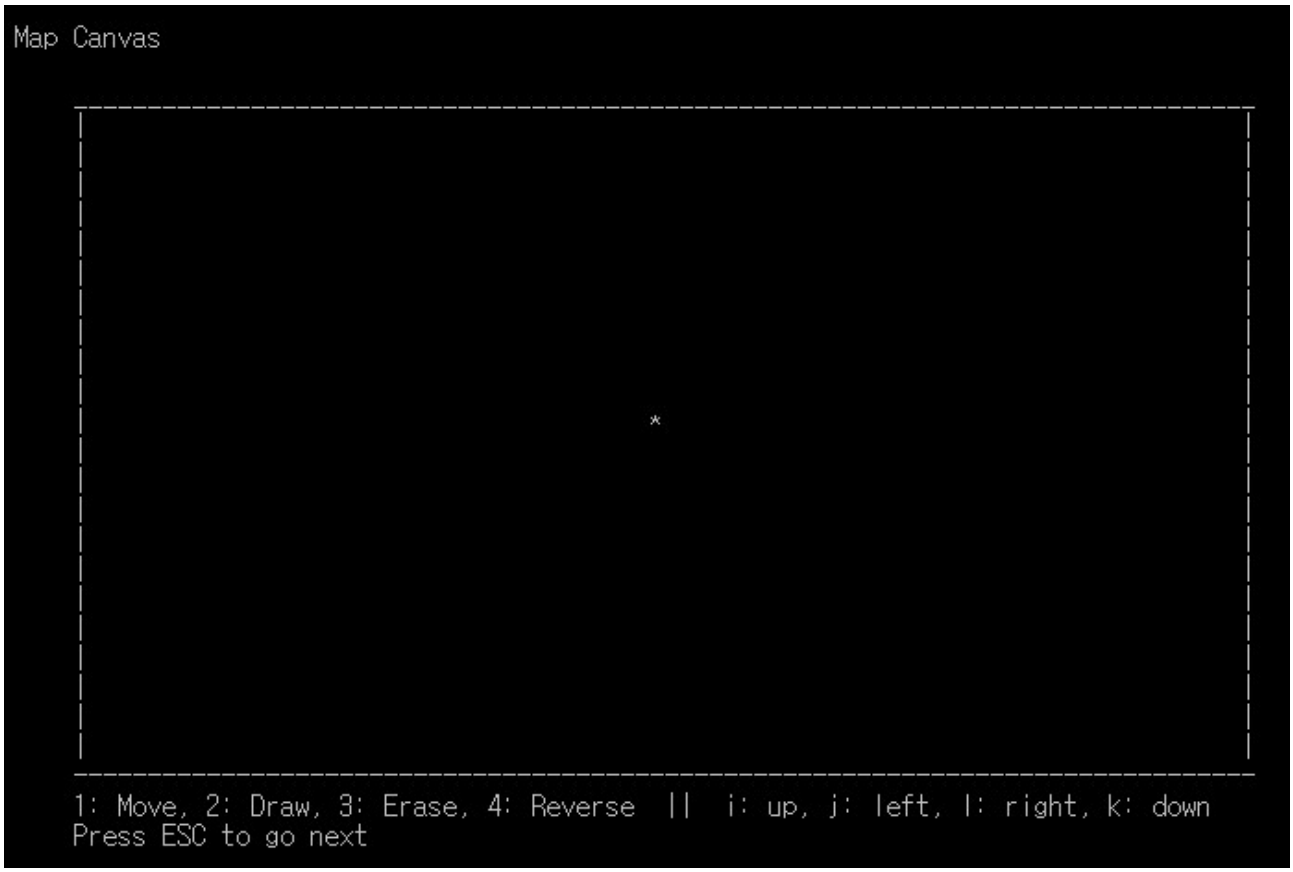
```
if(isWallBlock(map[(int)y][(int)x])){
    // meet vertical wall
    if(isWallBlock(map[(int)oldy][(int)x]) && isBlank(map[(int)y][(int)oldx])){
        fdx = -fdx;
        x = oldx + fdx;
        // bounced again
        if(isWallBlock(map[(int)y][(int)x])){
            // corner
            if(isBlank(map[(int)oldy][(int)oldx])){
                x = oldx;
                y = oldy;
                fdy = -fdy;
            } else {
                fdx = -fdx;
                x = oldx;
            }
        }
    // meet horizontal wall
    } else if(isBlank(map[(int)oldy][(int)x]) && isWallBlock(map[(int)y][(int)oldx])){
        fdy = -fdy;
        y = oldy + fdy;
        // bounced again
        if(isWallBlock(map[(int)y][(int)x])){
            // corner
            if(isBlank(map[(int)y][(int)oldx])){
                x = oldx;
                y = oldy;
                fdx = -fdx;
            } else {
                fdy = -fdy;
                y = oldy;
            }
        }
    // corner
    } else {
        fdx = -fdx;
        fdy = -fdy;
        x = oldx + fdx;
        y = oldy + fdy;
        // bounced again
        if(isWallBlock(map[(int)y][(int)x])){
            x = oldx;
            y = oldy;
        }
    }
}
```

4. Screen show of the result

Options

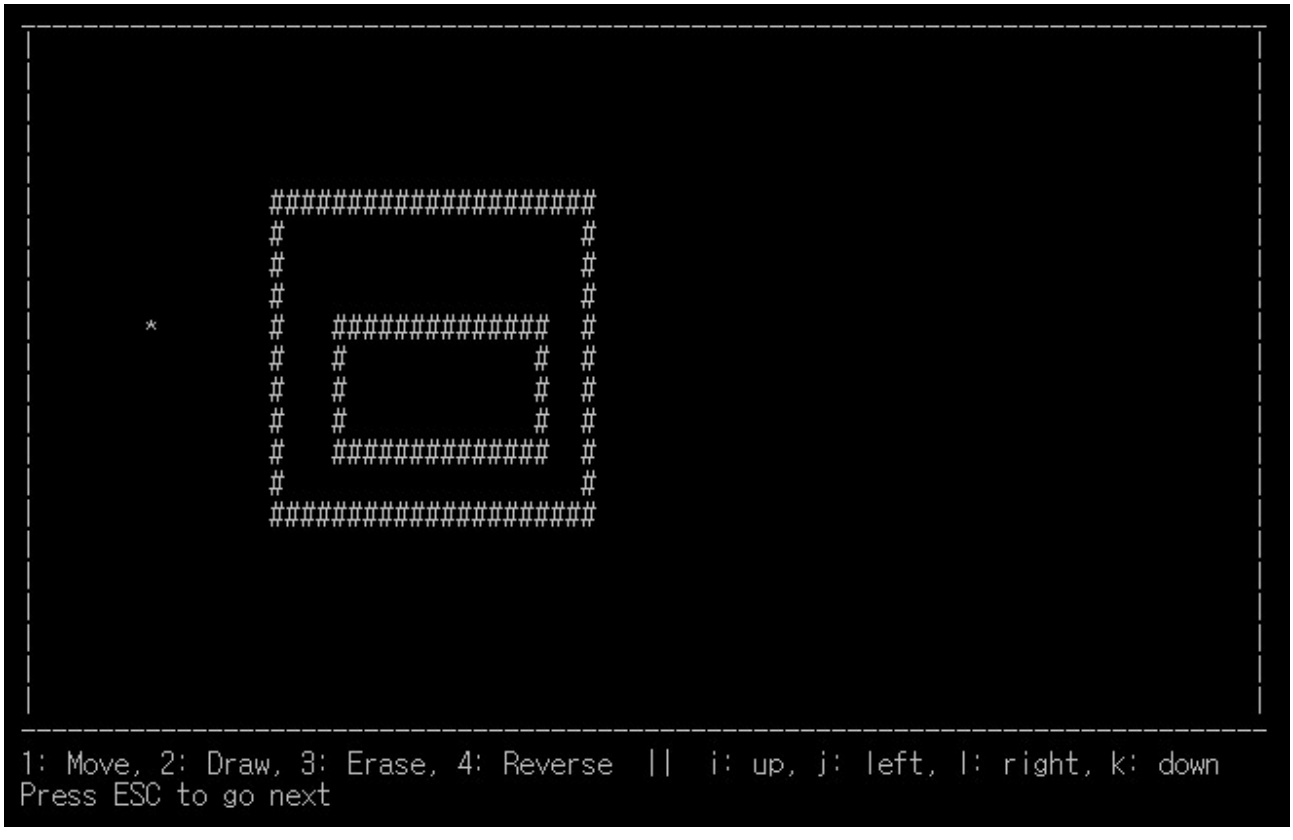


Option No.1 Draw Map



Move Cursor

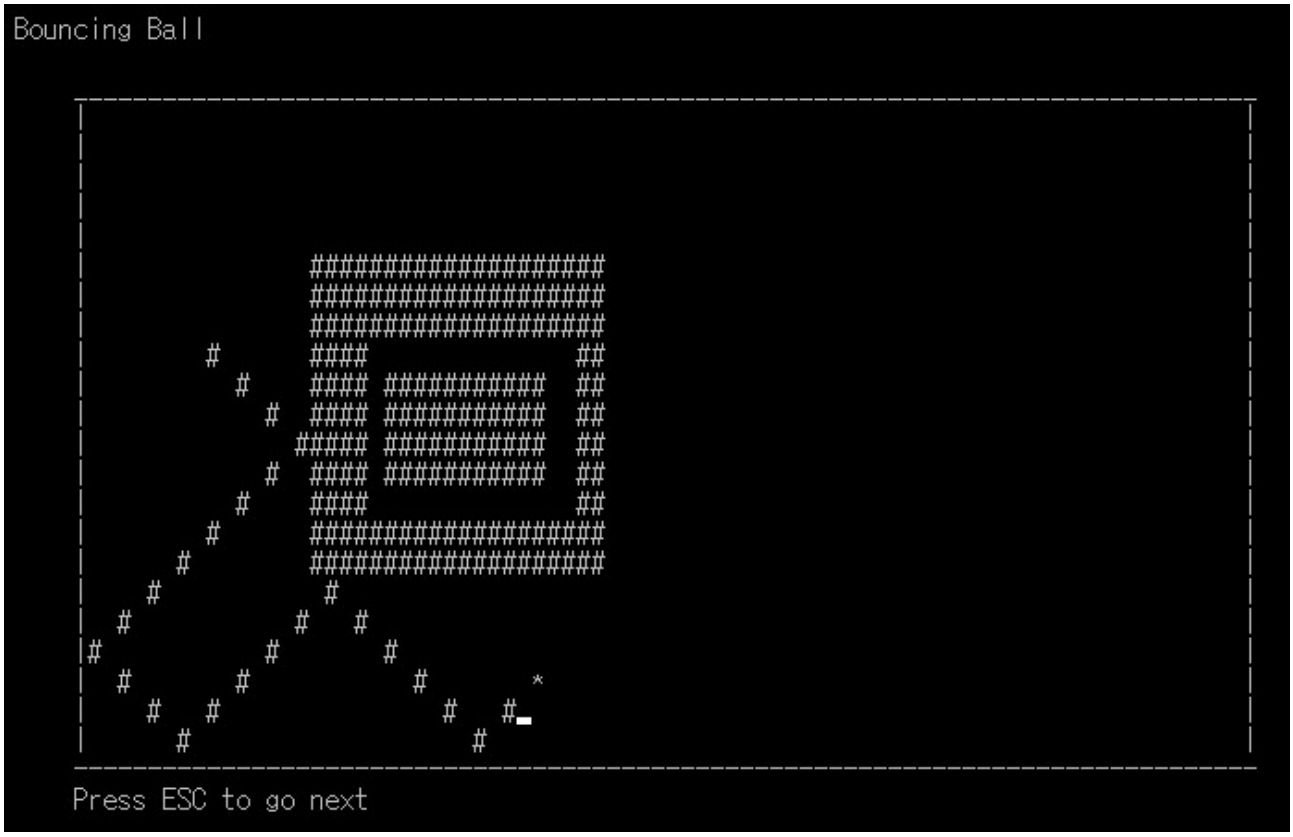
Draw Cursor



Erase Cursor

Reverse Cursor

Option No.2 Bouncing Ball



Option No.3 Clear Screen