과제1

배열의 크기

sizeof() 연산자를 활용하여 배열의 크기를 구할 수 있다.

```c
#include <stdio.h>

void function(int* list, size_t size);

int main(void)
{
    int list[5] = { 10, 20, 30, 40, 50 };

    size_t size;
    size = sizeof(list) / sizeof(int);
    function(list, size);

    size = sizeof(list) / sizeof(list[0]);
    function(list, size);

    return 0;
}

void function(int* list, size_t size)
{
    for (size_t i = 0; i < size; i++)
    {
        printf("%d ", *(list+i));
    }
    printf("\n");
}
```

```
10 20 30 40 50
10 20 30 40 50
```

2차원 배열과 함수

다양한 방법으로 2차원 배열을 함수에서 참조할 수 있다.

```c
#include<stdio.h>

void myPrint1(int* a, int row, int col);
void myPrint2(int a[4][3], int row, int col);
void myPrint3(int a[][3], int row, int col);
void myPrint4(int (*a)[3], int row, int col);

int main() {
    int arr[4][3] = { {1, 2, 3}, {4, 5, 6}, {7, 8, 9}, {10, 11, 12} };
    int row = sizeof(arr) / sizeof(arr[0]);
    int col = sizeof(arr[0]) / sizeof(arr[0][0]);
    myPrint1(arr, row, col);
    myPrint2(arr, row, col);
    myPrint3(arr, row, col);
    myPrint4(arr, row, col);
}
```

```c
void myPrint1(int* a, int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) printf("%d ", *(a + i * col + j));
        printf("\n");
    }
}

void myPrint2(int a[4][3], int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) printf("%d ", a[i][j]);
        printf("\n");
    }
}

void myPrint3(int a[][3], int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) printf("%d ", a[i][j]);
        printf("\n");
    }
}

void myPrint4(int (*a)[3], int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) printf("%d ", *(*(a+i)+j));
        printf("\n");
    }
}
```

포인터와 문자열

- 이중 포인터의 사용

```c
#include <stdio.h>

void SetStr(char** str);

int main(void)
{
    char* str;

    str = "I'am sad";

    SetStr(&str);

    printf("%s", str);

    return 0;
}

void SetStr(char** str)
{
    *str = "I'am happy";
}
```

함수 외부에서 정의된 포인터값, 즉 주소값을 함수의 인수로 받아서 변경하고자 할때 사용. 즉, 포인터 변수가 가르키는 곳을 바꾸고자 함.

2차원 배열과 함수

다양한 방법으로 2차원 배열을 함수에서 참조할 수 있다.

```c
#include<stdio.h>

void myPrint1(int* a, int row, int col);
void myPrint2(int a[4][3], int row, int col);
void myPrint3(int a[][3], int row, int col);
void myPrint4(int (*a)[3], int row, int col);

int main() {
    int arr[4][3] = { {1, 2, 3}, {4, 5, 6}, {7, 8, 9}, {10, 11, 12} };
    int row = sizeof(arr) / sizeof(arr[0]);
    int col = sizeof(arr[0]) / sizeof(arr[0][0]);
    myPrint1(arr, row, col);
    myPrint2(arr, row, col);
    myPrint3(arr, row, col);
    myPrint4(arr, row, col);
}
```

```c
void myPrint1(int* a, int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) printf("%d ", *(a + i * col + j));
        printf("\n");
    }
}

void myPrint2(int a[4][3], int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) printf("%d ", a[i][j]);
        printf("\n");
    }
}

void myPrint3(int a[][3], int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) printf("%d ", a[i][j]);
        printf("\n");
    }
}

void myPrint4(int (*a)[3], int row, int col) {
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) printf("%d ", *(*(a+i)+j));
        printf("\n");
    }
}
```

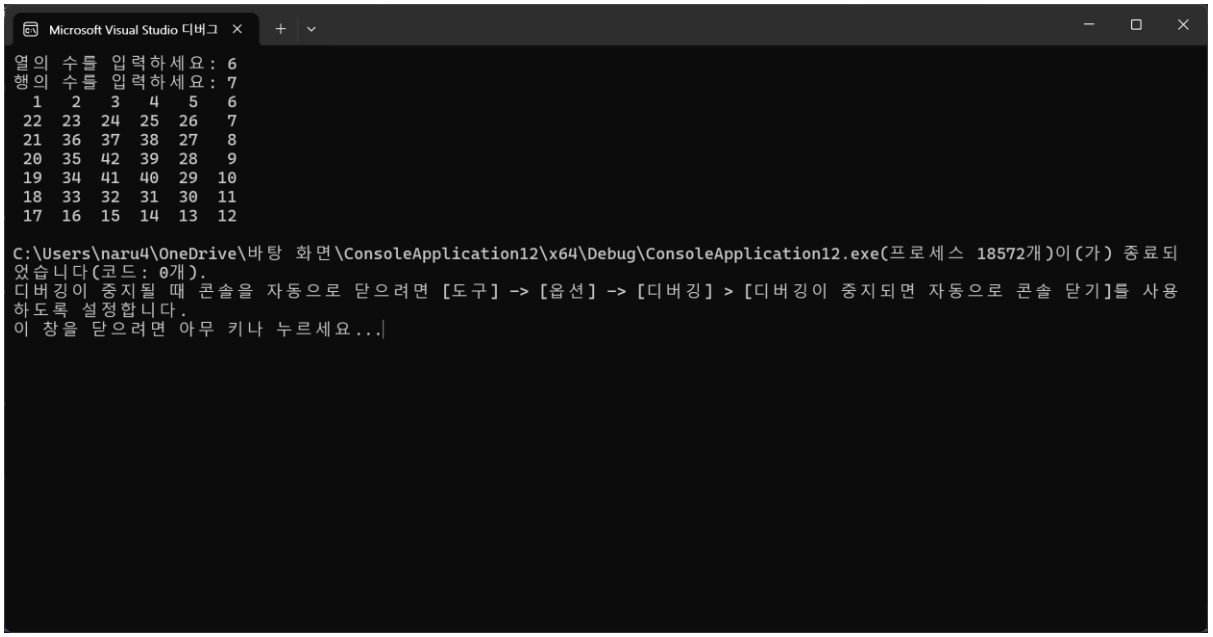
과제 3 소스 코드

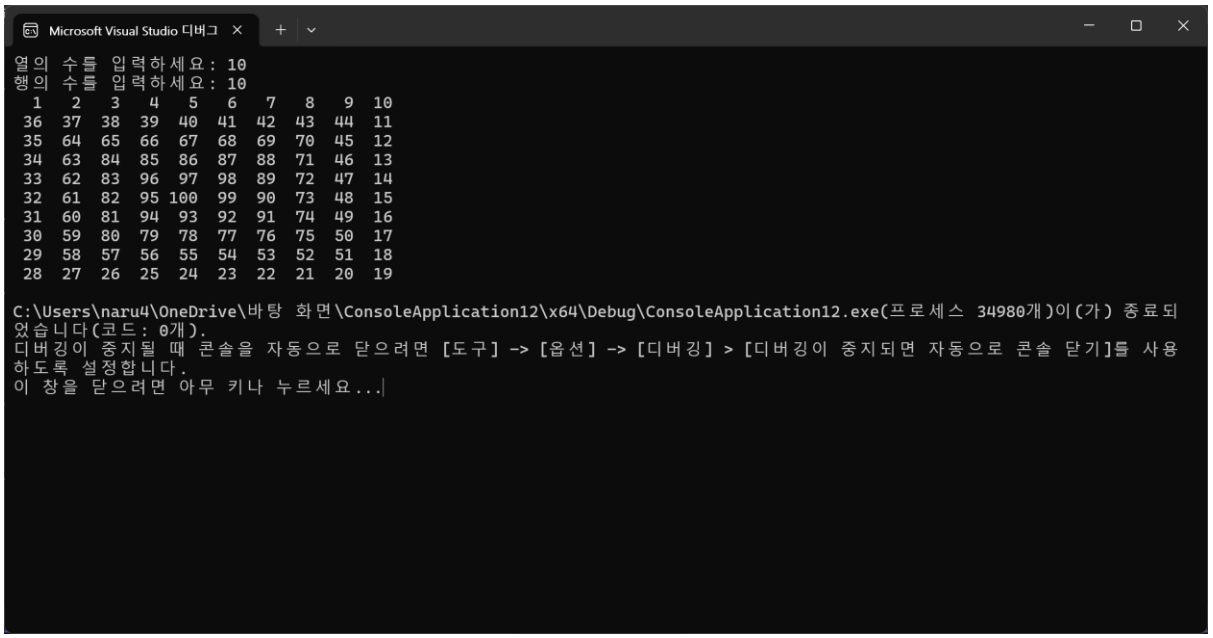
```c
#include <stdio.h>
#include <stdlib.h>

void print(int** arr, int sizeX, int sizeY);
void arr_ij(int** arr, int sizeX, int sizeY);

int main()
{
    int** arr;
    int row, col;

    printf("열의 수를 입력하세요: ");
    scanf_s("%d", &col);
    printf("행의 수를 입력하세요: ");
    scanf_s("%d", &row);

    arr = (int**)malloc(sizeof(int*) * row);//행 동적 할당
    for (int i = 0; i < row; i++)
    {
        arr[i] = (int*)malloc(sizeof(int) * col);//열 동적할당

    }
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            arr[i][j] = 0; // 배열 초기화
        }
    }

    arr_ij(arr, row, col);//2차원 배열에 숫자 입력
    print(arr, row, col);//도형 출력

    for (int i = 0; i < row; i++)
    {
        free(arr[i]);//열 동적 할당 해제
    }
    free(arr);//행 동적 할당 해제

    return 0;
}

void print(int** arr, int sizeX, int sizeY)
{
    for (int i = 0; i < sizeX; i++)
```

```
    {
        for (int j = 0; j < sizeY; j++)
        {
            printf("%3d ", arr[i][j]);//숫자가 출력될때 밀리지 않게 하기 위해 %3d,
만약 4가 출력 된다면 4가 아닌 004가 출력됨(00은 스페이스로 출력됨)
        }
        printf("₩n");
    }
}

void arr_ij(int** arr, int sizeX, int sizeY)
{
    int direction = 1;
    int x = 0, y = 0;
    for (int i = 1; i <= sizeX * sizeY; i++) //1부터 행x열(사각형의 크기)의
개수만큼 반복
    {
        arr[x][y] = i;//2차원 배열에 숫자 입력
        switch (direction)
        {
        case 1: // 우향
            if (y + 1 < sizeY && arr[x][y + 1] == 0)//우측 벽에 닿지
않을때=진행방향이 열의 크기보다 작거나 숫자가 차있지 않을때
            {
                y++;
            }
            else {
                direction = 2;//하향으로 방향전환
                x++;
            }
            break;
        case 2: // 하향
            if (x + 1 < sizeX && arr[x + 1][y] == 0)
            {
                x++;
            }
            else
            {
                direction = 3;//좌향으로 방향 전환
                y--;
            }
            break;
        case 3: // 좌향
            if (y - 1 >= 0 && arr[x][y - 1] == 0)
            {
                y--;
            }
```

```
                else {
                    direction = 4;//상향으로 방향전환
                    x--;
                }
                break;
            case 4: // 상향
                if (x - 1 >= 0 && arr[x - 1][y] == 0)
                {
                    x--;
                }
                else
                {
                    direction = 1;//우향으로 방향 전환
                    y++;
                }
                break;
            }
        }
    }
```

실행결과


Microsoft Visual Studio 디버그
열의 수를 입력하세요: 6
행의 수를 입력하세요: 7
1 2 3 4 5 6
22 23 24 25 26 7
21 36 37 38 27 8
20 35 42 39 28 9
19 34 41 40 29 10
18 33 32 31 30 11
17 16 15 14 13 12
C:\Users\naru4\OneDrive\바탕 화면\ConsoleApplication12\x64\Debug\ConsoleApplication12.exe(프로세스 18572개)이(가) 종료되었습니다(코드: 0개).
디버깅이 중지될 때 콘솔을 자동으로 닫으려면 [도구] -> [옵션] -> [디버깅] > [디버깅이 중지되면 자동으로 콘솔 닫기]를 사용하도록 설정합니다.
이 창을 닫으려면 아무 키나 누르세요...


Microsoft Visual Studio 디버그
열의 수를 입력하세요: 10
행의 수를 입력하세요: 10
1 2 3 4 5 6 7 8 9 10
36 37 38 39 40 41 42 43 44 11
35 64 65 66 67 68 69 70 45 12
34 63 84 85 86 87 88 71 46 13
33 62 83 96 97 98 89 72 47 14
32 61 82 95 100 99 90 73 48 15
31 60 81 94 93 92 91 74 49 16
30 59 80 79 78 77 76 75 50 17
29 58 57 56 55 54 53 52 51 18
28 27 26 25 24 23 22 21 20 19
C:\Users\naru4\OneDrive\바탕 화면\ConsoleApplication12\x64\Debug\ConsoleApplication12.exe(프로세스 34980개)이(가) 종료되었습니다(코드: 0개).
디버깅이 중지될 때 콘솔을 자동으로 닫으려면 [도구] -> [옵션] -> [디버깅] > [디버깅이 중지되면 자동으로 콘솔 닫기]를 사용하도록 설정합니다.
이 창을 닫으려면 아무 키나 누르세요...