

6주차 과제					
학년	2	학번	2016707079	이름	하상천

## 1. 과제 설명

### -과제1

2차원 배열을 통해 5x5 Map을 만들고, (0,0)에서 시작하여 X가 있는 위치까지의 최단 거리를 계산하여 출력하는 프로그램을 구현한다. 단, main 문에서는 map 만을 선언하고, 최단거리를 계산하는 부분은 다른 함수를 만들어서 구현하여 최종 결과값을 다시 main문에서 확인할 수 있도록 한다. Map은 자유롭게 선언 가능하나, 결과화면에 본인의 Map을 꼭 출력할 수 있도록 한다.

### -과제2

과제 1과 방식은 동일하나, 과제 2에서는 대각선 루트를 포함하는 최단거리를 계산한다. 과제 1과 마찬가지로 main문에서는 Map 선언과 결과 화면을 출력하도록 구현하고, 나머지 구현은 함수를 사용하도록 한다. Map은 자유롭게 선언 가능하고, 결과화면에 본인의 Map을 꼭 출력할 수 있도록 한다.

## 2. 과제 이론

### If문

조건을 주어 그 조건을 만족할 때에 실행하도록 한다.

Ex) if(i==1) 이면 i의 값이 1일 때만 실행하도록 한다.

그 외에는 다른 실행을 하고 싶다면 똑같은 형식으로 else if 또는 else로 조건을 주면 된다.

### For문

일정한 조건을 주어서 원하는 만큼 반복할 수 있게 해주는 반복문

Ex) for(i=0;i<10;i++); ---> i=0으로 시작하고 한번 반복 할 때마다 i값이 1씩 증가한다.

그리고 i<10이라는 조건이 만족 할 때까지만 반복한다. 즉 i=10이되면 반복하지 않는다.

## Sqrt 함수

매개변수 x로 들어온 숫자에 루트를 씌워서 계산한 값을 반환해주는 함수  
즉, 루트x를 구해주는 함수이다. Float형과 double형만 가능하다.

이 함수를 사용하기 위해서는 <math.h> 라이브러리가 필요하다.

Ex) float x = 25;

```
printf("sqrt(x) = %.2f", sqrt(x));
```

→sqrt(x) = 5.00

## Abs 함수

정수(int형)의 절댓값을 구해주는 함수

이 함수를 사용하기 위해서는 <math.h> 라이브러리가 필요하다.

Ex) abs(-4) = 4

## 3. 주요 소스 설명

-과제1

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

```
#include <math.h> // 절댓값 변환함수 abs를 사용하기 위한 헤더파일
```

```
int calculator(char map[5][5]) //반환형이 int인 최단거리를 계산하는 함수
```

```
{
```

```
    int count = 0, all_count = 0;
```

```
    int before, after = 0;
```

```
    int j = 0;
```

```
    for (int i = 0; i < 5; i++)
```

```
    {
```

```
        for ( j = 0; j < 5; j++)
```

```
        {
```

```
            count = 0; //매번 count를 0으로 초기화 시킨다.
```

```

        if (map[i][j] == 'X')
        {
            before = j; // for문을 반복하다가 'X'인 j를 만나면 그
값 j를 before에 대입해 놓는다.

            break; //가장 가까운 반복문을 중단한다.
        }
    } if (i == 0) // 첫째줄과 나머지줄들은 형태가 달라서 따로 계산하였다.
    {
        count += before; //count = count + before 라는 뜻으로 before
값을 count에 더한다.

        after = before; //before 값을 after에 저장해놓는다.

        count++; //세로방향(밑)으로 내려가는 1을 표현한 것이다.
    }

    if (i >= 1)
    {
        count += abs(after - before); //위에 줄에서 X를 만난 after 값
과 그 다음 아랫줄에서 X를 만난 before값 사이의 길이를 절댓값 변환함수 abs를 이용해
서 구한후 count에 더한다.

        after = before; //before 값을 after에 저장해놓는다.

        if (i != 4)
        {
            count++; //마지막줄이 아닌 경우는 세로방향(밑)으로
내려가는 1을 표현한 것이다.
        }
    }

    all_count += count; //모든 count 값을 all_count에 더

```

한다.

```
}
```

```
return all_count; //all_count값 즉, 최단거리를 main함수로 반환한다.
```

```
}
```

```
int main()
```

```
{
```

```
    char map[5][5];
```

```
    int shortest_distance;
```

```
    for (int i = 0; i < 5; i++) //5x5 map 만들기
```

```
    {
```

```
        map[0][i] = 'O';
```

```
        if (i == 4)
```

```
        {
```

```
            map[0][i] = 'X';
```

```
        }
```

```
    }
```

```
    for (int i = 0; i < 5; i++) //5x5 map 만들기
```

```
    {
```

```
        map[1][i] = 'O';
```

```
        if (i == 2)
```

```
        {
```

```

        map[1][i] = 'X';

    }

}

for (int i = 0; i < 5; i++) //5x5 map 만들기
{

    map[2][i] = 'O';

    if (i == 2)

    {

        map[2][i] = 'X';

    }

}

for (int i = 0; i < 5; i++) //5x5 map 만들기
{

    map[3][i] = 'O';

    if (i == 1)

    {

        map[3][i] = 'X';

    }

}

for (int i = 0; i < 5; i++) //5x5 map 만들기
{

    map[4][i] = 'O';

    if (i == 3)

```

```

        {

            map[4][i] = 'X';

        }

    }

    shortest_distance = calculator(map); //최단거리 값을 shortest_distance변수에
대입한다.

    printf("The shortest distance : %d\n", shortest_distance);

    printf("*****Map*****\n");

    for (int i = 0; i < 5; i++)    //map 출력

    {

        for (int j = 0; j < 5; j++)

        {

            printf("%3c", map[i][j]); //%3c를 이용해서 3칸을 할당하고 오른
쪽정렬 하였습니다.

        }

        printf("\n");

    }

}

```

-과제2

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

```
#include <math.h> // 절댓값 변환함수 abs와 제곱근을 구하는 함수 sqrt를 사용하기 위
한 헤더파일
```

float calculator(char map[5][5]) //반환형이 float인 최단거리를 계산하는 함수

```
{

    float count = 0, all_count = 0;

    int before, after = 0;

    int j, base = 0;

    for (int i = 0; i < 5; i++)

    {

        for (j = 0; j < 5; j++)

        {

            count = 0; //매번 count를 0으로 초기화 시킨다.

            if (map[i][j] == 'X')

            {

                before = j; // for문을 반복하다가 'X'인 j를 만나면

                그 값 j를 before에 대입해 놓는다.

                break; //가장 가까운 반복문을 중단한다.

            }

        }

        if (i == 0) // 첫째줄과 나머지줄들은 형태가 달라서 따로 계산하였

        다.

        {

            count += before; //count = count + before 라는 뜻으로

            before값을 count에 더한다.

            after = before; //before 값을 after에 저장해놓는다.

        }

        if (i >= 1)
```

```

        {

            base = abs(after - before); //위에 줄에서 X를 만난 after 값과
그 다음 아랫줄에서 X를 만난 before값 사이의 길이를 절댓값 변환함수 abs를 이용해서
구한 값을 base에 대입한다.

            count = sqrt(base * base + 1); //피타고라스의 정리를 변형한 후
제곱근을 구하는 함수 sqrt를 사용하고 그 값을 count에 대입한다.

            after = before; //before 값을 after에 저장해놓는다.

        }        all_count += count;    //모든 count 값을 all_count에
더한다.

    }

    return all_count; //all_count값 즉, 최단거리를 main함수로 반환한다.
}

int main()

{

    char map[5][5];

    float shortest_distance;

    for (int i = 0; i < 5; i++) //5x5 map 만들기

    {

        map[0][i] = 'O';

        if (i == 4)

        {

            map[0][i] = 'X';

        }

    }

```



```
}
```

```
for (int i = 0; i < 5; i++) //5x5 map 만들기
```

```
{
```

```
    map[1][i] = 'O';
```

```
    if (i == 2)
```

```
    {
```

```
        map[1][i] = 'X';
```

```
    }
```

```
}
```

```
for (int i = 0; i < 5; i++) //5x5 map 만들기
```

```
{
```

```
    map[2][i] = 'O';
```

```
    if (i == 2)
```

```
    {
```

```
        map[2][i] = 'X';
```

```
    }
```

```
}
```

```
for (int i = 0; i < 5; i++) //5x5 map 만들기
```

```
{
```

```
    map[3][i] = 'O';
```

```
    if (i == 1)
```

```
    {
```

```

        map[3][i] = 'X';

    }

}

for (int i = 0; i < 5; i++) //5x5 map 만들기
{

    map[4][i] = 'O';

    if (i == 3)

    {

        map[4][i] = 'X';

    }

}

shortest_distance = calculator(map); //최단거리 값을 shortest_distance변수에
대입한다.

printf("The shortest distance : %f\n", shortest_distance);

printf("*****Map*****\n");

for (int i = 0; i < 5; i++) //map 출력
{

    for (int j = 0; j < 5; j++)

    {

        printf("%3c", map[i][j]); //%3c를 이용해서 3칸을 할당하고 오
른쪽정렬 하였습니다.

    }


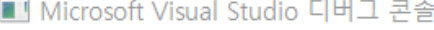
    printf("\n");

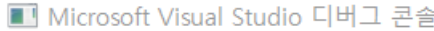

}

```

}

#### 4. 실행화면

 The shortest distance : 13 *****Map***** 0 0 0 0 X 0 0 X 0 0 0 0 X 0 0 0 X 0 0 0 0 0 0 X 0  C:\Users\seomk\source\repos\homework1 이 창을 닫으려면 아무 키나 누르세요.	 The shortest distance : 14 *****Map***** 0 0 X 0 0 0 0 0 0 X 0 X 0 0 0 0 0 X 0 0 X 0 0 0 0  C:\Users\seomk\source\repos\homework1 이 창을 닫으려면 아무 키나 누르세요.
과제1 실행화면 예시1	과제1 실행화면 예시2

 The shortest distance : 10.886349 *****Map***** 0 0 0 0 X 0 0 X 0 0 0 0 X 0 0 0 X 0 0 0 0 0 0 X 0  C:\Users\seomk\source\repos\homework1 이 창을 닫으려면 아무 키나 누르세요.	 The shortest distance : 8.990705 *****Map***** 0 0 X 0 0 0 X 0 0 0 0 X 0 0 0 0 0 0 0 X 0 0 0 X 0  C:\Users\seomk\source\repos\homework2 이 창을 닫으려면 아무 키나 누르세요.
과제2 실행화면 예시1	과제2 실행화면 예시2

#### 5. 고찰

과제1번과 과제2번은 서로 비슷해서 과제1번을 푸니까 과제2번은 금방 나왔다. 하지만 과제1번을 생각해내는 시간은 정말 길었다. 과제1번에서 첫번째줄과 나머지줄들은 방식이 달라서 if문을 이용해서 따로따로 계산했다. 매번 count변수를 0으로 초기화 시키고 for문을 반복시켜 X를 만났을 때의 j 값을 before라는 변수에 저장해놓았다. 그리고 그 윗줄에서 X를 만난 after값과 before 사이의 거리를 절댓값 변환함수 abs를 이용해서 구했다. abs함수를 이용하려면 <math.h>라이브러리가 필요하다. 그리고 abs는 정수형만 절댓값으로 변환해준다. 실수형을 절댓값으로 변환하려면 어떻게 해야 하는지 궁금해서 책과 인터넷을 찾아보았다. 실수형을 절댓값으로 변환하고 싶으면 fabs를 이용하면 된다. 한줄씩 내려갈 때마다 변수 count에 1을 더했다. 그 다음 all\_count

`+=count` 를 이용해서 `count` 값을 `all_count`에 더했다. `all_count += count` 는 `all_count = all_count + count` 라는 뜻이다. 중간고사에 출제되었을 때 앞뒤 순서를 제대로 몰라서 틀렸기 때문에 한번 더 복습하고 싶었는데 과제에 나와서 다시한번 생각해보는 시간이 되었다. 과제2번은 마찬가지로 `after`와 `before`사이의 거리를 `abs`함수를 통하여 구했다. 그 길이와 아랫줄로 내려가는 길이 1을 이용하여 피타고라스의 정리를 사용했고, 즉 대각선 최단거리를 구했다. 이때 제곱근을 구할 때 사용하는 함수 `sqrt`를 사용했다. `sqrt`함수를 사용하려면 마찬가지로 `<math.h>`라이브러리가 필요하다. 그리고 `sqrt`함수는 `float`형과, `double`형만 가능하다. 이렇게 최단거리를 구하는 함수를 만들고 최종 최단거리를 `main`함수로 반환했다. 처음에 반환형을 생각하지 않고 `int`형으로만 계속하다가 오류가 나서 결과값이 나오지 않았다. 이번 과제를 통해서 함수 만드는 것에 대한 부담감을 조금 덜게 되었고 반환형을 잘 확인해야겠다는 생각을 하였다. 그래도 한문제, 한문제 풀때마다 성취감도 생기고 점점 C언어가 재밌어지는 것 같다. 그리고 `<math.h>`라이브러리에는 또 어떤 다른 것이 있는지 찾아보아야겠다.