

## Level19 Direct 사용하기 [난이도 : 3]

문제 1번 [숙제 목록보기]

3	5	4
1	1	2
1	3	9

위 3x3 배열을 하드코딩 해주세요.

그리고 좌표 하나를 입력 받아주세요.(y, x)

입력받은 좌표에서 바로 윗칸, 아랫칸, 왼쪽칸, 오른쪽칸의 합을

Direct 방식을 이용해서 출력 해주세요.

ex1) 1 1 입력시 (y, x)

	5	
1		2
	3	

이므로  $5+1+2+3 = 11$  출력

ex2) 0 1 입력시 (y, x)

3		4
	1	

유효한 칸은 세칸이므로  $3+1+4 = 8$  출력

## 입력 예제

1 1

## 출력 결과

11

```

#include <iostream>
using namespace std;
// direct 방식
int main()
{
    int arr2D[3][3] =
    {
        3,5,4,
        1,1,2,
        1,3,9
    };

    int x = 0, y = 0;
    cin >> y >> x;
    // 입력받은 좌표에 offset 값들을 더해주면 네방향의 좌표를 구할 수 있다.
    int offset[4][2] = {
        0,1,    // right
        0,-1,   // left
        1,0,    // bottom
        -1,0    // top
    };

    int sum = 0;
    for (int i = 0; i < 4; ++i)
    {
        int val = 0;
        int chY = 0;
        int chX = 0;
        chY = y + offset[i][0];
        chX = x + offset[i][1];
        if (chY >= 0 && chX >= 0)
        {
            val = arr2D[chY][chX];
            sum += val;
        }
    }

    cout << sum;

    return 0;
}

```

# Level19 구조체 변수 떠올리기

[난이도 : 1]

문제 2번 [[숙제](#) [목록보기](#)]

**Data**

int x

int y

int z

Data 구조체를 만들고 구조체 변수 a, b를 만들어 주세요.

숫자 6개를 **a.x** , **a.y** , **a.z** , **b.x** , **b.y** , **b.z** 에 각각 입력 받고,

**a.x + b.x**

**a.y + b.y**

**a.z + b.z**      값을 출력 해주세요.

ex)

입력:

1 2 3

4 5 6

출력:

5

7

9

## 입력 예제

1 2 3

4 5 6

## 출력 결과

5

7

9

```
#include <iostream>
using namespace std;

struct Data
{
    int x;
    int y;
    int z;
};

int main()
{
    Data a;
    Data b;
    cin >> a.x >> a.y >> a.z;
    cin >> b.x >> b.y >> b.z;

    cout << a.x + b.x << endl;
    cout << a.y + b.y << endl;
    cout << a.z + b.z << endl;

    return 0;
}
```

# Level19 맥도날드 주문받기 [난이도 : 2]

문제 3번 [숙제 목록보기]

MC

burger1	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
burger2	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

왼쪽과 같은 형태의 구조체를 정의해주세요.

그리고 구조체 변수 bob, tom을 만들고, 문장 4개를 구조체 변수에 입력 받아주세요.



sanghi, bulgogi, chicken, dove

만약 네 문장을 아래와 같이 입력 받았다면, 아래와 같이 채워주세요.

bob

burger1	s	a	n	g	h	i		
burger2	b	u	l	g	o	g	i	

tom

burger1	c	h	i	c	k	e	n	
burger2	d	o	v	e				

이제 각각 몇글자인지 출력하면 됩니다.

## 입력 예제

sanghi

bulgogi

chicken

dove

## 출력 결과

bob.burger1=6

bob.burger2=7

```
tom.burger1=7
```

```
tom.burger2=4
```

```
#include <iostream>
using namespace std;

struct MC
{
    char burger1[8];
    char burger2[8];
};

int Counting(char* str)
{
    bool isEnd = false;
    int len = 0;
    while (!isEnd)
    {
        if (str[len] == '\0')
            isEnd = true;
        else
            len++;
    }
    return len;
}

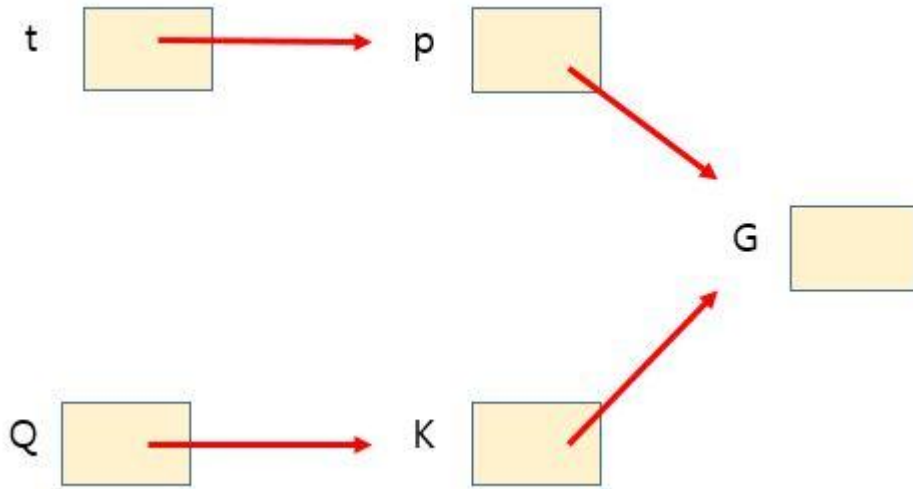
int main()
{
    MC bob;
    MC tom;
    cin >> bob.burger1 >> bob.burger2;
    cin >> tom.burger1 >> tom.burger2;

    cout << "bob.burger1=" << Counting(bob.burger1) << endl;
    cout << "bob.burger2=" << Counting(bob.burger2) << endl;
    cout << "tom.burger1=" << Counting(tom.burger1) << endl;
    cout << "tom.burger2=" << Counting(tom.burger2) << endl;

    return 0;
}
```

## Level19 더블포인터 [난이도 : 2]

문제 4번 [\[숙제 목록보기\]](#)



t와 Q는 더블포인터 입니다.

위 상태를 구현하고

변수 G에다가 숫자 1개를 입력 받으세요.

그리고 **\*\*t**, **\*K**의 값을 출력 해주세요.

### 입력 예제

5

### 출력 결과

5 5

```
#include <iostream>
using namespace std;

int main()
{
    int G = 0;
    cin >> G;
    int* p = &G;
    int* K = &G;
    int** t = &p;
    int** Q = &K;

    cout << **t << " " << * K << endl;

    return 0;
}
```



## Level19 용의자의 GPS [난이도 : 3]

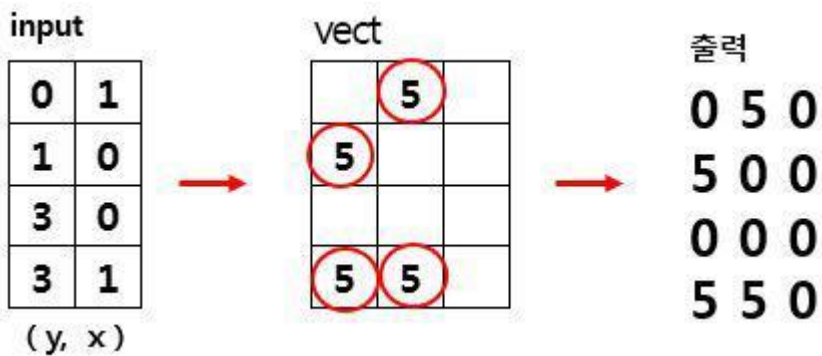
문제 5번 [\[숙제 목록보기\]](#)

네쌍의 좌표를 input 배열에 입력 받아주세요.

그리고 vect배열 4x3에 배열을 만들고 0으로 초기화 해 주세요.

vect배열에서 input 배열에 든 좌표에 해당하는 곳을 찾아 그 좌표에 숫자 5를 넣어주세요.

그리고 그 결과를 출력 해주세요.



[HINT]

```
vect[0][1] = 5;
```

```
vect[1][0] = 5;
```

```
vect[3][0] = 5;
```

```
vect[3][1] = 5;
```

## 입력 예제

```
0 1
1 0
3 0
3 1
```

## 출력 결과

```
0 5 0
5 0 0
```

0 0 0

5 5 0

```
#include <iostream>
using namespace std;


int main()
{
    int input[4][2] = {};
    for (int i = 0; i < 4; ++i)
    {
        cin >> input[i][0];
        cin >> input[i][1];
    }
    int vect[4][3] = {};
    for (int k = 0; k < 4; ++k)
    {
        int y = input[k][0];
        int x = input[k][1];
        vect[y][x] = 5;
    }
    for (int y = 0; y < 4; ++y)
    {
        for (int x = 0; x < 3; ++x)
        {
            cout << vect[y][x] << " ";
        }
        cout << endl;
    }

    return 0;
}
```

# Level19 기차에서 첫사랑 찾기

[난이도 : 4]

문제 6번 [숙제 목록보기]

	0	1	2	3	4	5	6
	win 15 name s u m m e r	win 33 name c l o e	win 24 name s u m m e r	win 28 name n i k i	win 32 name j e n n y	win 20 name s u m m e r	win 40 name c o c o

위와 같이 구조체 기차에 사람들이 타고 있습니다.

win에는 나이가

name에는 이름이 적어져 있습니다.

tom은 첫사랑을 찾기 위해 이 기차를 탔습니다. (구조체 배열 기차 입니다)

tom의 첫사랑 이름과 나이를 입력받고, 몇번 index에 있는지 찾아서 출력해주세요.

[힌트1] 구조체변수 하드코딩 하기

```
#include<iostream>
using namespace std;

struct Node
{
    int x, y;
};

int main()
{
    Node a = { 3, 4 };
    Node b = { 5, 6 };

    cout << a.x << " " << a.y << endl;
    cout << b.x << " " << b.y << endl;
}
```

[힌트2] 구조체배열 하드코딩하기

```

#include<iostream>
using namespace std;

struct Train
{
    int win;
    char name[8];
};

int main()
{
    Train t[3] = { {35, "ABC"}, {100, "BBQ"}, {15, "KFC"} };

    cout << t[0].name;
}

```

[힌트3] 문장은 비교가 되지 않습니다.

```
char a[10] = "BBQ";
```

```
char b[20] = "ABC";
```

```
if (a == b) //error!
```

for문을 돌려 한 글자씩 비교를 해주어야 합니다.

함수를 만들어 비교를 하면 됩니다.

## 입력 예제

```
summer
```

```
20
```

## 출력 결과

```
5
```

```

#include <iostream>
using namespace std;

struct Node
{
    int win;        // 나이
    char name[10];  // 이름
};

int Counting(char* name)
{
    bool isEnd = false;
    int len = 0;
    while (!isEnd)
    {
        if (name[len] == '\0')
            break;
        else
            len++;
    }
    return len;
}

int main()
{
    Node train[7] = {};
    train[0] = { 15, "summer" };           // 구조체 값 넣는 방법1
    train[1] = { 33, "cloe" };
    train[2] = { 24, "summer" };
    train[3] = { 28, "niki" };
    train[4] = { 32, "jenny" };
    train[5] = { 20, "summer" };
    train[6] = { 40, "coco" };

    int loverAge = 0;
    char loverName[10] = {};
    cin >> loverName >> loverAge;

    int len1 = Counting(loverName);
    int len2 = 0;
    int seat = 7;
    for (int i = 0; i < 7; ++i)
    {
        if (loverAge == train[i].win)      // 나이 비교
        {
            len2 = Counting(train[i].name);
            if (len1 == len2)              // 이름 글자수 비교

```

```

        {
            int cnt = 0;
            for (int k = 0; k < len1; ++k)    // 이름 철자 비교
            {
                if (loverName[k] == (train[i].name)[k])
                    cnt++;
            }
            if (cnt == len1)
            {
                seat = i;                // 첫사랑 좌석
            }
            break;
        }
    }
    if (seat != 7)
        cout << seat;
    else
        cout << "없음";

    return 0;
}

```

# Level19 가장 큰 곳 찾기 [난이도 : 5]

문제 7번 [숙제 [목록보기](#)]

map

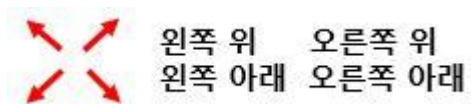
3	3	5	3	1
2	2	4	2	6
4	9	2	3	4
1	1	1	1	1
3	3	5	9	2

map 배열을 하드코딩 해주세요.

그리고 `sum(y,x)` 함수를 만들어 주세요.

이 함수는 특정좌표(y,x)에서

왼쪽위, 오른쪽 위, 왼쪽 아래, 오른쪽 아래의 합을 return 해주는 함수입니다.



이 `sum`함수를 이용해서 가장 큰 값이 나오는 좌표를 출력하세요.

(direct 기법을 사용해주세요, 입력값은 없습니다.)

## 출력 결과

3 2

```

#include <iostream>
using namespace std;

int map[5][5] =
{
    3,3,5,3,1,
    2,2,4,2,6,
    4,9,2,3,4,
    1,1,1,1,1,
    3,3,5,9,2
};

int sum(int y, int x)
{
    int sum = 0;
    int offset[4][2] =
    {
        -1,-1, // left-top
        -1,1,  // right-top
        1,-1,  // left-bottom
        1,1    // right-bottom
    };

    for (int i = 0; i < 4; ++i) // 순서 : top -> right -> bottom -> left (시계방향)
    {
        int chX = 0;
        int chY = 0;
        chY = y + offset[i][0];
        chX = x + offset[i][1];
        if (chY >= 0 && chX >= 0)
        {
            sum += map[chY][chX];
        }
    }

    return sum;
}

int main()
{
    int maxSum = sum(0, 0);
    int maxX = 0;
    int maxY = 0;
    for (int y = 0; y < 5; ++y)
    {
        for (int x = 0; x < 5; ++x)
        {

```



```
        int newSum = sum(y, x);
        if (maxSum < newSum)
        {
            maxSum = sum(y, x);
            maxY = y;
            maxX = x;
        }
    }

    cout << maxY << " " << maxX;

    return 0;
}
```

## Level19 폭탄 투하 [난이도 : 4]

문제 8번 [숙제 [목록보기](#)]

-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

4x5 char 배열을 준비해주세요.

값을 '-' 문자로 꼭 채워주세요.

폭탄을 투하할 좌표 두곳을 입력 받아주세요.(y,x)

만약 폭탄이 (1,1)에 투하되면 8방향으로 폭탄이 터집니다.

따라서

#	#	#	-	-
#	-	#	-	-
#	#	#	-	-
-	-	-	-	-

그리고 다시 (3,3)에 투하 되면 8방향으로 폭탄이 터집니다.

따라서

#	#	#	-	-
#	-	#	-	-
#	#	#	#	#
-	-	#	-	#

폭탄 2개가 투하 되었을때 그 현장을 출력 해주세요.

(Direct 기법을 사용해서 코딩해주세요)

## 입력 예제

1 1  
3 3

## 출력 결과

# # # \_ \_  
# \_ # \_ \_  
# # # # #  
\_ \_ # \_ #

```

#include <iostream>
using namespace std;

int main()
{
    char map[4][5] = {};
    for (int y = 0; y < 4; ++y)
    {
        for (int x = 0; x < 5; ++x)
        {
            map[y][x] = '_';
        }
    }

    int bombX[2] = {};
    int bombY[2] = {};
    cin >> bombX[0] >> bombY[0];
    cin >> bombX[1] >> bombY[1];

    int offset[8][2] =
    {
        -1,0,    // top
        -1,1,    // top-right
        0,1,     // right
        1,1,     // bottom-right
        1,0,     // bottom
        1,-1,    // bottom-left
        0,-1,    // left
        -1,-1    // top-left
    };

    for (int i = 0; i < 2; ++i)
    {
        int chX = 0;
        int chY = 0;
        for (int k = 0; k < 8; ++k)
        {
            chY = bombY[i] + offset[k][0];
            chX = bombX[i] + offset[k][1];
            if (chY >= 0 && chY <= 3 && chX >= 0 && chX <= 4)
                // 범위초과 조건도 고려해줘야한다.
            {
                map[chY][chX] = '#';
            }
        }
    }
}

```

```
    for (int y = 0; y < 4; ++y)
    {
        for (int x = 0; x < 5; ++x)
        {
            cout << map[y][x];

        }
        cout << endl;
    }

    return 0;
}
```

# Level19 sigma 이미지 프로세싱

[난이도 : 4]

문제 9번 [[숙제](#) [목록보기](#)]

입력 →

image			
1	5	5	4
4	2	1	1
3	9	3	2
4	5	9	1

4x4 image배열이 있습니다.

먼저 4x4 image배열에 숫자를 입력 받아주세요.

image 배열의 특정 좌표를 지목하면,

2x3 사이즈의 합을 return 해주는 rectSum 함수를 만들어 주세요

예를들어 rectSum(0,0)을 호출하면  $1+5+5+4+2+1 = 18$  이 return 됩니다.

위 예제에서는 (2,0)이 최대 합 입니다.

이 함수를 활용하여 2x3 사이즈의 합의 최대값이 나오는 좌표를 찾아주세요.

(direct를 쓰는 문제가 아닙니다)

## 입력 예제

```
1 5 5 4
4 2 1 1
3 9 3 2
4 5 9 1
```

## 출력 결과

```
(2,0)
```

```

#include <iostream>
using namespace std;

int image[4][4] =
{
    1,5,5,4,
    4,2,1,1,
    3,9,3,2,
    4,5,9,1
};

int rectSum(int y, int x)
{
    int sum = 0;
    int offset[6][2] =
    {
        0,0,
        0,1,
        0,2,
        1,0,
        1,1,
        1,2
    };
    for (int i = 0; i < 6; ++i)
    {
        int chY = y + offset[i][0];
        int chX = x + offset[i][1];
        if (chY >= 0 && chY <= 3 && chX >= 0 && chX <= 3)
        {
            sum += image[chY][chX];
        }
    }

    return sum;
}

int main()
{
    /*for (int y = 0; y < 4; ++y)
    {
        for (int x = 0; x < 4; ++x)
        {
            cin >> image[y][x];
        }
    }*/
    int maxSum = rectSum(0, 0);

```

```
int maxX = 0;
int maxY = 0;
for (int y = 0; y < 4; ++y)
{
    for (int x = 0; x < 4; ++x)
    {
        int newSum = rectSum(y, x);
        if (maxSum < newSum)
        {
            maxSum = newSum;
            maxY = y;
            maxX = x;
        }
    }
}

cout << "(" << maxY << "," << maxX << ")";

return 0;
}
```