다차원 배열

- 다차원 배열의 선언과 활용 -

성공회대학교 IT융합자율학부
소프트웨어공학전공
홍 성 준

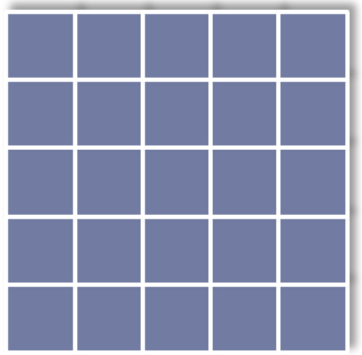
다차원 배열

◎ 다차원 배열?

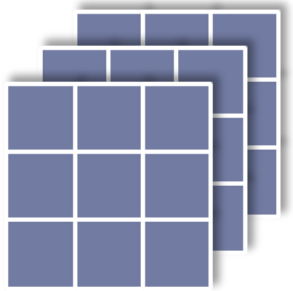
- 2차원 이상으로 선언된 배열

```
int arrOneDim[10];          길이가 10인 1차원 int형 배열
int arrTwoDim[5][5];        가로, 세로의 길이가 각각 5인 2차원 int형 배열
int arrThreeDim[3][3][3];   가로, 세로, 높이의 길이가 각각 3인 3차원 int형 배열
```

1차원 배열 arrOneDim



2차원 배열 arrTwoDim



3차원 배열 arrThreeDim

2차원 배열

◎ 2차원 배열의 선언

- TYPE name[세로길이][가로길이];

int arr1[3][4];

int arr2[2][6];

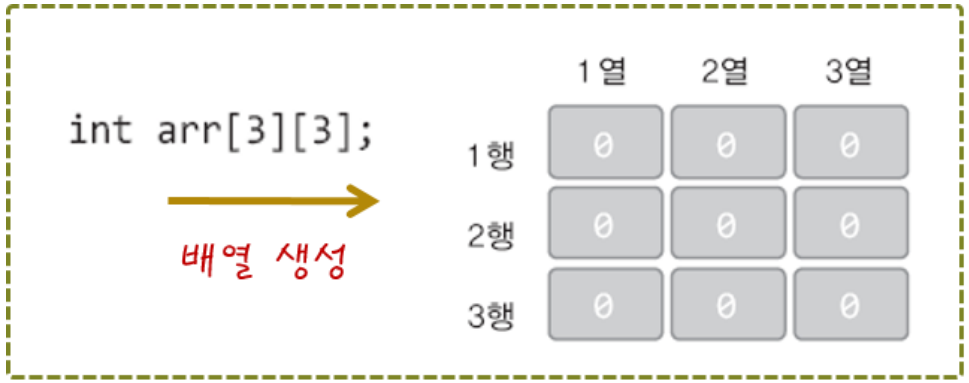
- TwoDimArraySize.c

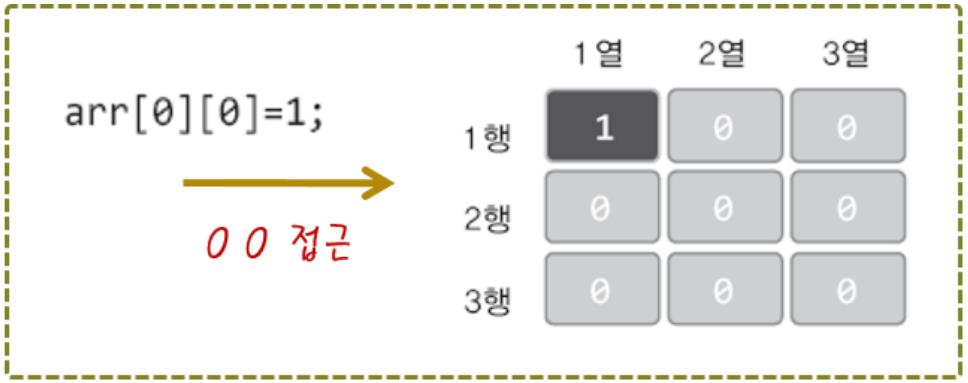
```
int main(void)
{
    int arr1[3][4];
    int arr2[7][9];
    printf("세로3, 가로4: %d \n", sizeof(arr1));
    printf("세로7, 가로9: %d \n", sizeof(arr2));
    return 0;
}
```

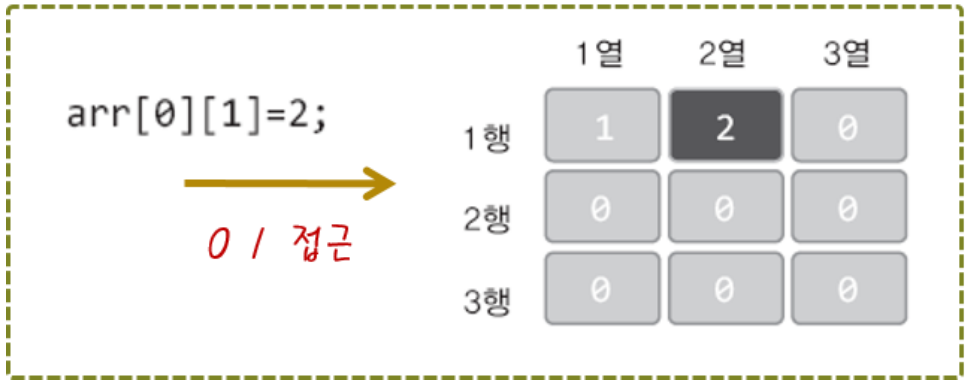
```
세로3, 가로4: 48
세로7, 가로9: 252
```

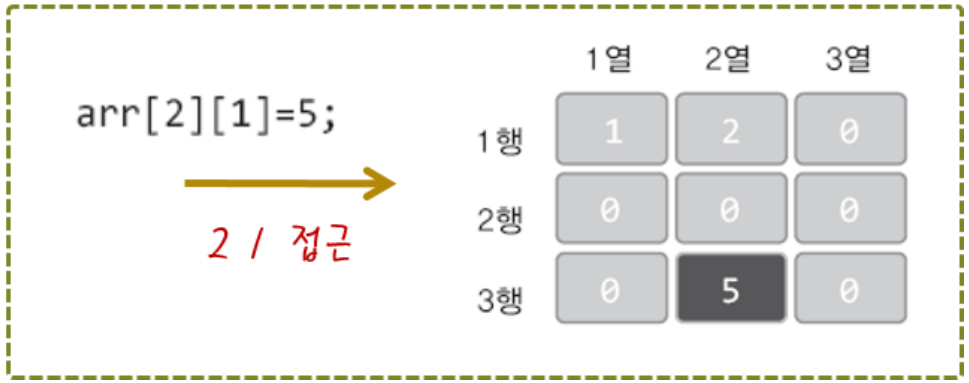
2차원 배열

◎ 2차원 배열에서 요소의 접근









2차원 배열

◎ 2차원 배열에서 요소의 접근

- PopuResearch.c

```
int main(void)
{
    int villa[4][2];
    int popu, i, j;

    /* 가구별 거주인원 입력 받기 */
    for(i=0; i<4; i++)
    {
        for(j=0; j<2; j++)
        {
            printf("%d층 %d호 인구수: ", i+1, j+1);
            scanf("%d", &villa[i][j]);
        }
    }

    /* 빌라의 층별 인구수 출력하기 */
    for(i=0; i<4; i++)
    {
        popu=0;
        popu += villa[i][0];
        popu += villa[i][1];
        printf("%d층 인구수: %d \n", i+1, popu);
    }
    return 0;
}
```

```
1층 1호 인구수: 2
1층 2호 인구수: 4
2층 1호 인구수: 3
2층 2호 인구수: 5
3층 1호 인구수: 2
3층 2호 인구수: 6
4층 1호 인구수: 4
4층 2호 인구수: 3
1층 인구수: 6
2층 인구수: 8
3층 인구수: 8
4층 인구수: 7
```

2차원 배열

◎ 2차원 배열의 메모리 할당

- 2차원 배열이더라도 가로(열)에 우선하여 1차원으로 주소 값이 할당
- TwoDimArrayAddr.c

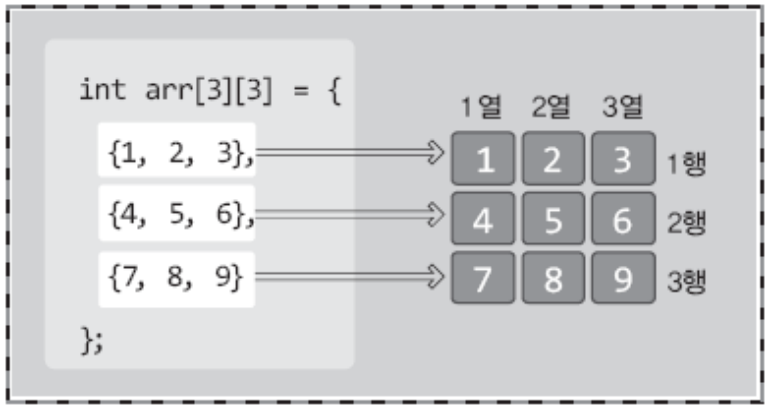
```
int main(void)
{
    int arr[3][2];
    int i, j;
    for(i=0; i<3; i++)
        for(j=0; j<2; j++)
            printf("%p \n", &arr[i][j]);
    return 0;
}
```

```
002AFD54
002AFD58
002AFD5C
002AFD60
002AFD64
002AFD68
```

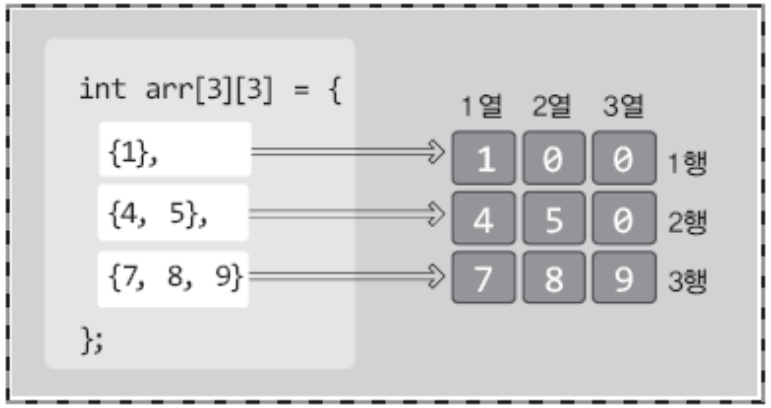
2차원 배열

◎ 2차원 배열의 선언과 초기화

- 2차원 배열의 초기화 리스트 안에 중괄호로 행 단위를 구분하여 배열의 요소를 초기화



- 초기화 리스트에서 일부를 생략하는 경우, 비는 공간은 0으로 초기화 (1차원 배열과 동일)

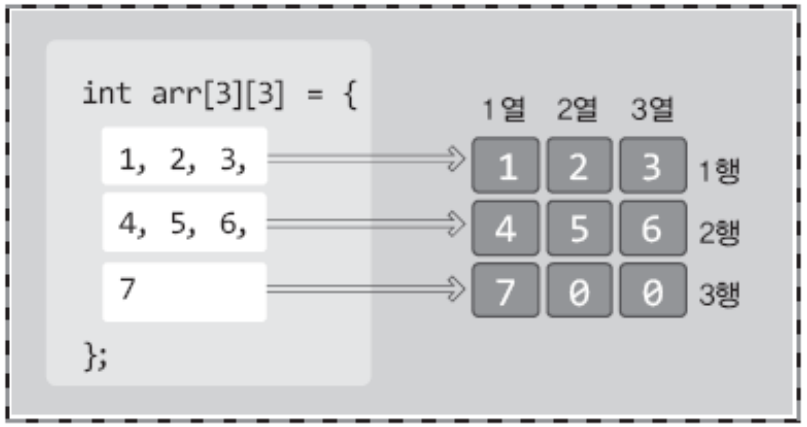



```
int arr[3][3] = {
    {1, 0, 0},
    {4, 5, 0},
    {7, 8, 9}
};
```

2차원 배열

◎ 2차원 배열의 선언과 초기화 (cont.)

- 행 단위 중괄호를 사용하지 않으면 열에 우선하여 순서대로 초기화




```
int arr[3][3]={1, 2, 3, 4, 5, 6, 7};
int arr[3][3]={1, 2, 3, 4, 5, 6, 7, 0, 0};
```

2차원 배열

◎ TwoDimArrayInit.c

```
int main(void)
{
    int i, j;

    /* 2차원 배열 초기화의 예 1 */
    int arr1[3][3]={
        {1, 2, 3},
        {4, 5, 6},
        {7, 8, 9}
    };

    /* 2차원 배열 초기화의 예 2 */
    int arr2[3][3]={
        {1},
        {4, 5},
        {7, 8, 9}
    };
    /* 2차원 배열 초기화의 예 3 */
    int arr3[3][3]={1, 2, 3, 4, 5, 6, 7};

    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
            printf("%d ", arr1[i][j]);
        printf("\n");
    }
    printf("\n");

    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
            printf("%d ", arr2[i][j]);
        printf("\n");
    }
    printf("\n");

    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
            printf("%d ", arr3[i][j]);
        printf("\n");
    }
    return 0;
}
```

```
1 2 3
4 5 6
7 8 9

1 0 0
4 5 0
7 8 9

1 2 3
4 5 6
7 0 0
```

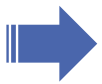
2차원 배열

◎ 2차원 배열의 선언과 초기화 (cont.)

```
int arr[][]={1, 2, 3, 4, 5, 6, 7, 8};
```

- 2차원 배열 선언 시, 행의 개수(세로의 길이)만 생략이 가능

```
int arr1[][4]={1, 2, 3, 4, 5, 6, 7, 8};
int arr2[][2]={1, 2, 3, 4, 5, 6, 7, 8};
```



```
int arr1[2][4]={1, 2, 3, 4, 5, 6, 7, 8};
int arr2[4][2]={1, 2, 3, 4, 5, 6, 7, 8};
```

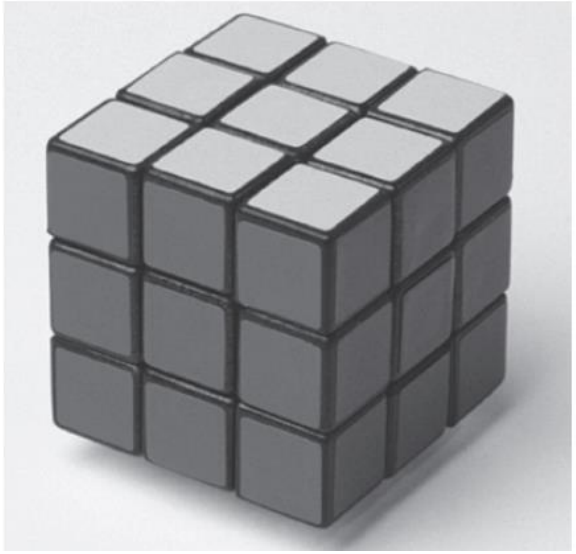
3차원 배열

◎ 3차원 배열의 선언

- 2차원 배열을 쌓은 형태
- TYPE name[높이][세로길이][가로길이];

```
int main(void)
{
    int arr1[2][3][4];
    double arr2[5][5][5];
    printf("높이2, 세로3, 가로4 int형 배열: %d \n", sizeof(arr1));
    printf("높이5, 세로5, 가로5 double형 배열: %d \n", sizeof(arr2));
    return 0;
}
```

```
높이2, 세로3, 가로4 int형 배열: 96
높이5, 세로5, 가로5 double형 배열: 1000
```



3차원 배열

◎ 3차원 배열의 선언과 접근

- ThreeDimArrayAccess.c

```
int main(void)
{
    int mean=0, i, j;
    int record[3][3][2]={
        {
            {70, 80},   // A 학급 학생 1의 성적
            {94, 90},   // A 학급 학생 2의 성적
            {70, 85}    // A 학급 학생 3의 성적
        },
        {
            {83, 90},   // B 학급 학생 1의 성적
            {95, 60},   // B 학급 학생 2의 성적
            {90, 82}    // B 학급 학생 3의 성적
        },
        {
            {98, 89},   // C 학급 학생 1의 성적
            {99, 94},   // C 학급 학생 2의 성적
            {91, 87}    // C 학급 학생 3의 성적
        }
    };
    for(i=0; i<3; i++)
        for(j=0; j<2; j++)
            mean += record[0][i][j];
    printf("A 학급 전체 평균: %g \n", (double)mean/6);

    mean=0;
    for(i=0; i<3; i++)
        for(j=0; j<2; j++)
            mean += record[1][i][j];
    printf("B 학급 전체 평균: %g \n", (double)mean/6);

    mean=0;
    for(i=0; i<3; i++)
        for(j=0; j<2; j++)
            mean += record[2][i][j];
    printf("C 학급 전체 평균: %g \n", (double)mean/6);
    return 0;
}
```

```
A 학급 전체 평균: 81.5
B 학급 전체 평균: 83.3333
C 학급 전체 평균: 93
```