9. 배열

3주차

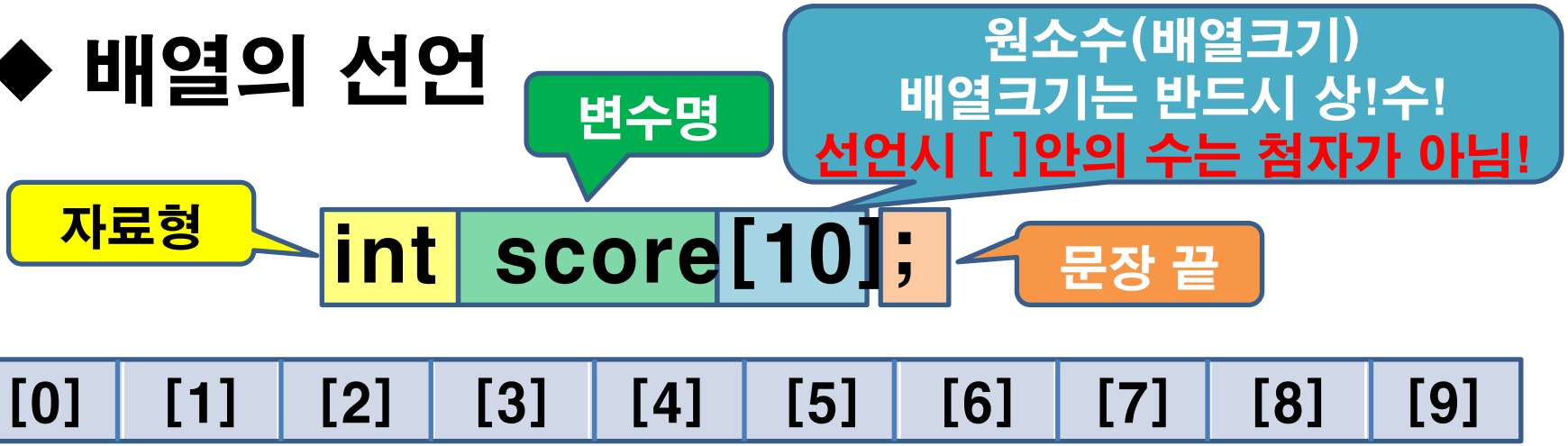
배열
(필요성은?)

◆ 배열이란

❖ 동일한 자료형의 여러 변수들을 일괄 선언

❖ 저장장소의 연속된 위치에 저장됨.

◆ 배열의 선언



배열의 첨자(index)

- 첨자(index)란
  - 배열 내의 각 원소를 참조하기 위한 수.
  - [ ] 안에 번호로 표시

- 첨자의 범위
  - 원소수가 N 이면 첨자의 범위는 0 ~ N-1

    int score[10];
    - 첫 원소 : score[0]
    - 마지막 원소 : score[9]

  - 유효한 값의 범위를 벗어난 첨자 사용은 절대 금지! 프로그래머의 책임!

배열원소 일괄 출력
(declarearray.c)

```
#include <stdio.h>
#define SIZE 5
int main(void) {
    int score[SIZE];//int score[5];
    score[0] = 78;
    score[1] = 97;
    score[2] = 85;
    score[4] = 91;
    //score[5] = 50; //문법오류는 발생하지 않으나 실행오류 발생

    for (int i = 0; i < SIZE; i++)
        printf("%d  ", score[i]);
    printf("₩n");
    return 0;
}
```

[결과]
78 97 85 -858993460 91

배열 4번째 원소에 값 저장하지 않아 쓰레기 값 저장

배열원소 출력

배열선언시의 초기화

- 배열을 함수내부에서 선언 후 원소에 초기값을 저장하지 않으면 쓰레기 값을 가짐.
  -> 항상 초기화 해야 함

- 중괄호 이용하여 각 원소값을 순서대로 적음

```
int  a[4] = { 10, 30, 50, 40 };
```

- 배열 초기화의 여러 예
  - 초기화된 개수만큼 자동으로 원소 수 정해 짐

```
int a1[] = { 10, 30, 50, 40 };
```

  - 선언한 원소수보다 많이 초기화할 수는 없음

```
int a2[2] = { 10, 30, 50, 40 }; // 오류!
```

  - 선언한 원소수보다 적게 초기화하면 나머지 값은 0

```
int a3[10] = { 10, 30, 50, 40 };
```

- 중괄호 이용한 초기화는 선언시에만 가능

```
int a[4];
a = {10, 30, 50, 40 };          // 오류!
```

배열 초기화 예제

[출력결과]

총점=517
평균=74

```
#include <stdio.h>

int  main(void)
{
   int  score[7] = { 87, 65, 78, 77, 47, 75, 88};
   int  i, sum = 0;

   모든 점수를 sum에 더하기

   printf( "총점=%d\n", sum );
   printf( "평균=%d\n",                                    );
   return 0;
}
```

실수계산하여
소수점아래 첫째자리에서 반올림

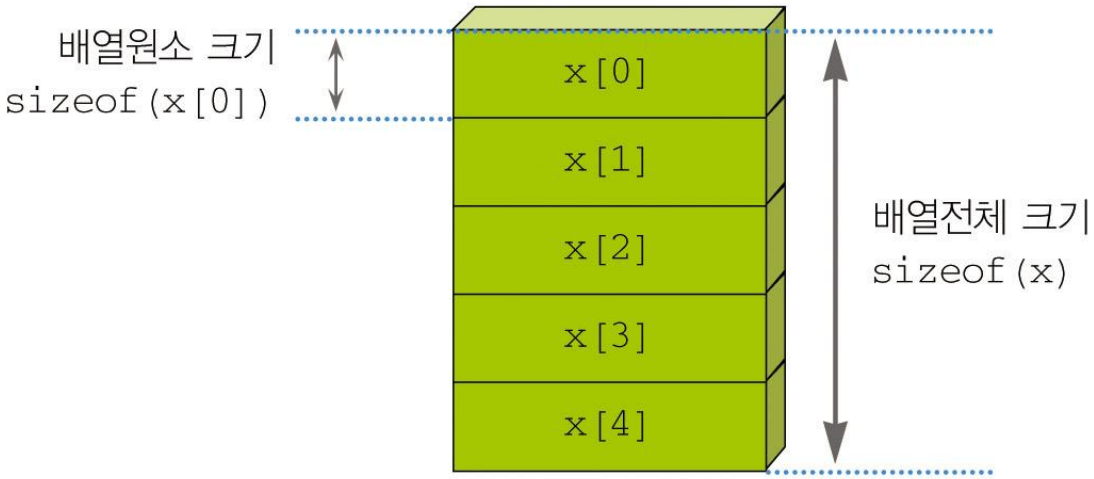
연산자 sizeof 로 배열 원소 수 구하기

◆ 아래와 같은 배열이 있을 때

```
int  a[] = { 10, 20, 30 };
```

❖ 배열의 크기 : sizeof(a) : 4x3=12
❖ 배열 원소의 크기 : sizeof(a[0]) :4

◆ 배열의 원소 수
❖ sizeof(s) / sizeof(a[0])



배열의 합 구하기 예제

◆ 아래 실수로써 배열 초기화 하여 이들의 합을 출력하는 프로그램을 작성하시오.
10.5, 5.7, 12.9, 4.56

```
#include <stdio.h>

int main(void)
{
        double x[] = { 10.5, 5.7, 12.9, 4.56 };
        int i, cnt;
        double sum = 0.0;

        cnt = sizeof(x) / sizeof(x[0]);
        for (i = 0;i < cnt;i++)
                sum += x[i];
        printf( "sum=%g\n", sum );
        return 0;
}
```

이차원 배열 선언과 사용

- 이차원 배열 개요 (행 우선 배열)
  - 평면 테이블 형태
  - 1차원 배열(행)이 여러 개(열) 있는 것
- 이차원 배열선언
  - 2개의 대괄호가 필요 [행 개수][열 개수]
  - 배열선언 시 초기값을 저장하지 않는다면 반드시 행과 열의 크기를 모두 명시해야 함.
  - 배열선언 시 초기값을 준다면 행의 개수 생략 가능.
- 이차원 배열 원소 access
  - 2개의 인덱스 사용
  - arr[row][col]
    - row+1번째 행의 col+1번째 열 원소

이차원 배열 원소 참조와 출력 (twodarray.c)

```
#include <stdio.h>
#define ROWSIZE 2
#define COLSIZE 3
int  main(void) {
    int  td[ROWSIZE][COLSIZE];

    td[0][0] = 1; td[0][1] = 2; td[0][2] = 3;
    td[1][0] = 4; td[1][1] = 5; td[1][2] = 6;

    printf("반목문 for를 이용하여 출력₩n");
    for (int i = 0; i < ROWSIZE; i++){
        for (int j = 0; j < COLSIZE; j++)
            printf("td[%d][%d] == %d ", i, j, td[i][j]);
        printf("₩n");
    }
    return 0;
}
```

[결과]
반목문 for를 이용하여 출력
td[0][0] == 1 td[0][1] == 2 td[0][2] == 3
td[1][0] == 4 td[1][1] == 5 td[1][2] == 6

2차원 배열 원소에 값 저장

이차원 배열 초기화 (inittwodarray.c)

```
#include <stdio.h>
#define ROWSIZE 2
#define COLSIZE 3
int main(void) {
    int td[][3] = { { 1 },{ 1, 2, 3 } };
    int td2[2][3] = { 1, 2, 3, 4 };
    printf("반목문 for를 이용하여 출력₩n");
    for (int i = 0; i < ROWSIZE; i++) {
        for (int j = 0; j < COLSIZE; j++)
            printf("%d ", td[i][j]);
        printf("₩n");
    }
    printf("td2₩n");
    for (int i = 0; i < ROWSIZE; i++) {
        for (int j = 0; j < COLSIZE; j++)
            printf("%d ", td2[i][j]);
        printf("₩n");
    }
```

중괄호를 중첩하여 사용. 값을 주지 않은 원소들은 0.

하나의 중괄호를 사용. 값을 주지 않은 원소들은 0.

[결과]
반목문 for를 이용하여 출력
1 0 0
1 2 3
td2
1 2 3
4 0 0

이차원 배열 이용한 성적처리 (tdscore.c)

```
#include <stdio.h>
#define ROWSIZE 4
#define COLSIZE 2
int main(void) {
    int sum = 0, midsum = 0, finalsum = 0;
    int score[][COLSIZE] = { 95, 85, 90, 88, 86, 90, 88, 78 };
    printf("        중간        기말₩n");
    printf("------------------------₩n");
    for (int i = 0; i < ROWSIZE; i++) {
        for (int j = 0; j < COLSIZE; j++) {
            printf("%10d ", score[i][j]);
            sum += score[i][j];
            if (j == 0) midsum += score[i][j];
            else finalsum += score[i][j];
        }
        puts("");
    }
    printf("------------------------₩n");
    printf("평균: %7.2f %7.2f₩n",
        (double)midsum / ROWSIZE, (double)finalsum / ROWSIZE);
    printf("₩n성적의 합은 %d이고 ", sum);
    printf("평균은 %.2f이다.₩n", (double)sum / (ROWSIZE * COLSIZE));
```

[결과]

```
 중간       기말
------------------------
        95         85
        90         88
        86         90
        88         78
------------------------
평균:    89.75    85.25

성적의 합은 700이고 평균은 87.50이
```

삼차원 배열
(threedary.c)

```
#include <stdio.h>
#define ROWSIZE 4
#define COLSIZE 2
int main(void) {
    int score[][ROWSIZE][COLSIZE] = {
        { { 95, 85 },{ 85, 83 },{ 92, 75 },{ 90, 88 } },
        { { 88, 77 },{ 72, 95 },{ 88, 92 },{ 93, 83 } } };
    for (int i = 0; i < 2; i++) {
        if (i == 0) printf("[강좌 1]");
        else printf("[강좌 2]");
        printf("%11s%7s₩n", "중간", "기말");
        for (int j = 0; j < ROWSIZE; j++) {
            printf("%10s%2d", "학생", j + 1);
            for (int k = 0; k < COLSIZE; k++)
                printf("%6d ", score[i][j][k]);
            printf("₩n");
        }
        printf("₩n");
```

3차원 배열 초기화, 첫 번째 크기는 생략 가능.

```
[결과]
[강좌 1]        중간    기말
      학생 1    95     85
      학생 2    85     83
      학생 3    92     75
      학생 4    90     88

[강좌 2]        중간    기말
      학생 1    88     77
      학생 2    72     95
      학생 3    88     92
      학생 4    93     83
```