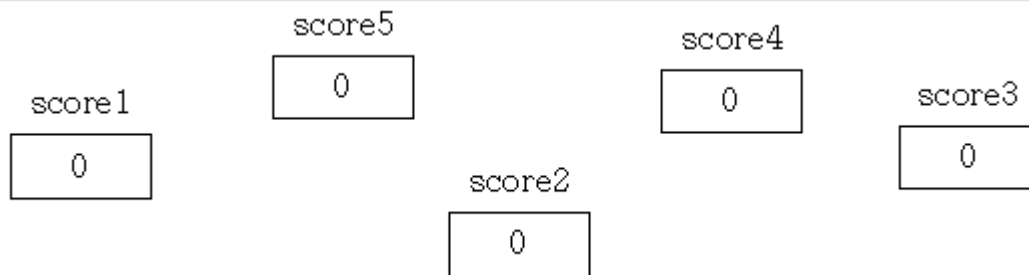


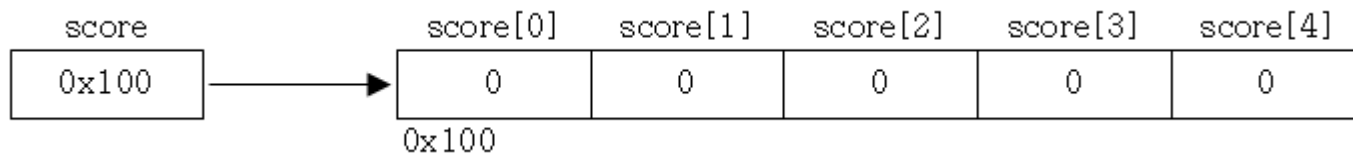
■ 배열(array)

- 같은 타입의 여러 변수를 하나의 묶음으로 다루는 것
- 많은 양의 값(데이터)을 다룰 때 유용하다.
- 배열의 각 요소는 서로 연속적이다.

```
int score1=0, score2=0, score3=0, score4=0, score5=0 ;
```



```
int[] score = new int[5]; // 5개의 int 값을 저장할 수 있는 배열을 생성한다.
```



■ 배열의 장점

- 많은 양의 데이터를 적은 코드로 처리(중복된 변수 사용을 최소화)
- 반복문 이용해 요소들을 쉽게 처리

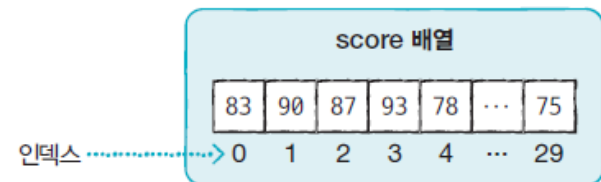
ex) 학생 30명의 성적을 저장하고 평균값을 구하려면?

```
int score1 = 83;
int score2 = 90;
int score3 = 87;
...
int score30 = 75;

int sum = score1;
sum += score2;
sum += score3;
...
sum += score30;
int avg = sum / 30;
```

반복문을 이용한 배열 처리

```
int sum = 0;
for(int i=0; i<30; i++) {
    sum += score[i];
}
int avg = sum / 30;
```



■ 배열의 선언과 생성

- 타입 또는 변수이름 뒤에 대괄호[]를 붙여서 배열을 선언한다.

자료형[] 배열이름 = new 자료형[개수];

int[] arr = new int[10];

자료형 배열이름[] = new 자료형[개수];

int arr[] = new int[10];

메모리 구조



■ 배열의 초기화

- 생성된 배열에 처음으로 값을 저장하는 것
- 선언과 동시에 초기화가 가능
- 초기화 할 때는 배열의 개수를 명시하지 않음

```
int[ ] studentIDs = new int[ ] {101, 102, 103}; //개수는 생략함
```

```
int[ ] studentIDs = new int[3] {101, 102, 103}; //오류 발생
```

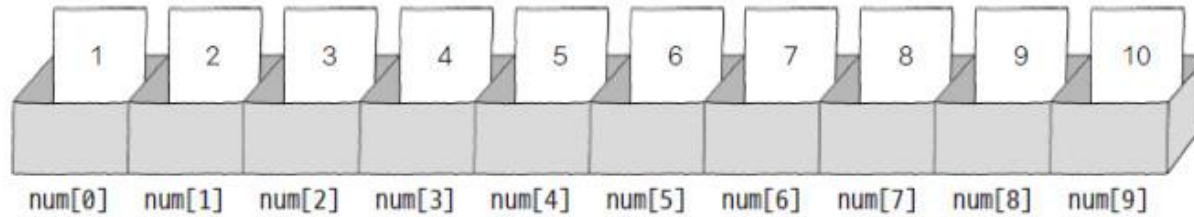
```
int[ ] studentIDs = {102, 102, 103}; //int형 요소가 3개인 배열 생성
```

아무런 초기화 값이 없이 선언만 한 경우, 정수는 0, 실수는 0.0 객체 배열은 null 로 초기화 됨

■ 배열의 활용

- 배열의 위치를 지정하여 자료를 가져오거나 변경 할 수 있음
- 배열의 순서를 0부터 시작
- N 개의 배열은 0 부터 N-1 위치까지 자료가 존재

예) `int[] num = new int[]{1,2,3,4,5,6,7,8,9,10};`



■ 배열의 활용

● 배열에 값을 저장하고 읽어오기

```
score[3] = 100;          // 배열 score의 4번째 요소에 100을 저장한다.  
int value = score[3];    // 배열 score의 4번째 요소에 저장된 값을 읽어서 value에 저장.
```

● '배열이름.length'는 배열의 크기를 알려준다.

```
int[] score = { 100, 90, 80, 70, 60, 50 };
```

```
for(int i=0; i < 6; i++) {  
    System.out.println(score[i]);  
}
```



```
for(int i=0; i < score.length; i++) {  
    System.out.println(score[i]);  
}
```

■ 배열 사용 – 1 (for)

```
public class ArrayExam1 {  
    public static void main(String[] args) {  
        int[] arr1 = new int[3];  
        int[] arr2 = new int[] {10, 20, 30};  
        int[] arr3 = {100, 200, 300};  
  
        arr1[0] = 1;  
        arr1[1] = 2;  
        arr1[2] = 3;  
  
        for(int i = 0; i < 3; i++) {  
            System.out.println(arr1[i]);  
        }  
        for(int i = 0; i < arr1.length; i++) {  
            System.out.println(arr1[i]);  
        }  
    }  
}
```

■ 배열 사용 – 2 (for-each)

```
public class ArrayExam2 {  
    public static void main(String[] args) {  
        int[] arr = {100, 200, 300};  
        for(int a : arr) {  
            System.out.println(a);  
        }  
    }  
}
```


■ 배열 사용 – 3 (최소값 구하기)

```
public class ArrayExam3 {  
    public static void main(String[] args) {  
        int[] numbers = {3, 2, 1, 7, 4};  
  
        int min = 0;  
  
        for(int i = 0; i < numbers.length; i++) {  
            if(numbers[i] < min || min == 0) {  
                min = numbers[i];  
            }  
        }  
  
        System.out.println("최소값 : " + min);  
    }  
}
```

■ 배열 사용 – 4 (글자 정렬)

```
public class ArrayExam4 {  
    public static void main(String[] args) {  
        char[] chars = {'b', 'e', 'z', 'a', 'w'};  
        char temp = ' ';  
        for(int i = 0; i < chars.length - 1; i++) {  
            for(int j = i + 1; j < chars.length; j++) {  
                if(chars[i] > chars[j]) {  
                    temp = chars[i];  
                    chars[i] = chars[j];  
                    chars[j] = temp;  
                }  
            }  
        }  
        System.out.println(chars);  
    }  
}
```

■ 배열의 활용

● 배열 복사

- 배열은 한 번 생성하면 크기 변경 불가
- 더 많은 저장 공간이 필요하다면 보다 큰 배열을 새로 만들고
이전 배열로부터 항목 값들을 복사

● 배열 복사 방법

- for문 이용
- `System.arraycopy()` 메소드 이용

■ 배열 복사 – 배열 길이 확장

```
public class ArrayCopy {  
    public static void main(String[] args) {  
        int[] arr = new int[3];  
  
        arr[0] = 0;  
        arr[1] = 1;  
        arr[2] = 2;  
  
        int[] arr2 = new int[5];  
  
        System.arraycopy(arr, 0, arr2, 0, arr.length);  
  
        arr2[3] = 3;  
        arr2[4] = 4;  
  
        for(int i = 0; i < arr2.length; i++) {  
            System.out.println(arr2[i]);  
        }  
    }  
}
```

■ 다차원 배열의 선언과 생성

● []의 개수가 차원의 수를 의미한다.

선언방법	선언예
타입[] [] 변수이름;	int[] [] score;
타입 변수이름[] [];	int score[] [];
타입[] 변수이름[];	int[] score[];

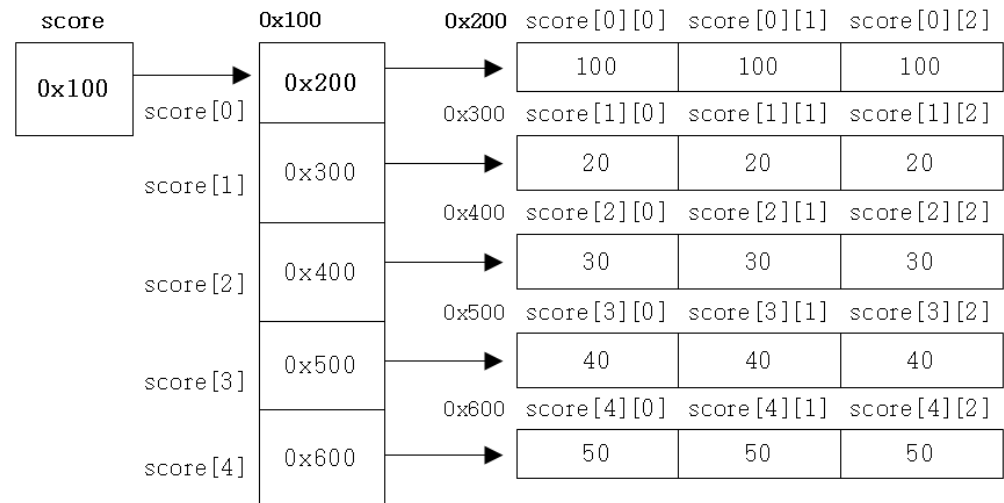
[표5-3] 2차원 배열의 선언

```
int[] [] score = {
    {100, 100, 100},
    { 20, 20, 20},
    { 30, 30, 30},
    { 40, 40, 40},
    { 50, 50, 50},
};
```

```
int[] [] score = new int[5][3];    // 5행 3열의 2차원 배열을 생성한다.
```

	국어	영어	수학
1	100	100	100
2	20	20	20
3	30	30	30
4	40	40	40
5	50	50	50

```
for (int i=0; i < score.length; i++) {
    for (int j=0; j < score[i].length; j++) {
        score[i][j] = 10;
    }
}
```



[그림5-2] 2차원 배열

■ 다차원 배열 사용 - 1

```
public class ArrayExam5 {  
    public static void main(String[] args) {  
        int[][] arr1 = new int[][]{  
            {1, 2, 3},  
            {4, 5, 6},  
            {7, 8, 9},  
            {10, 11, 12}  
        };  
  
        int[][] arr2 = {  
            {10, 20, 30},  
            {40, 50, 60},  
            {70, 80, 90},  
            {100, 110, 120}  
        };  
    }  
}
```

```
int[][] arr3 = new int[2][2];  
arr3[0][0] = 100;  
arr3[0][1] = 200;  
arr3[1][0] = 300;  
arr3[1][1] = 400;  
}  
}
```

■ 다차원 배열 사용 – 2 (배열 이동)

```
public class ArrayExam6 {  
    public static void main(String[] args) {  
        int[][] arr = {  
            {1, 2, 3},  
            {4, 5, 6},  
            {7, 8, 9}  
        };  
  
        int[] last = arr[arr.length - 1];  
  
        for(int i = arr.length - 1; i >= 0; i--) {  
            if(i == 0) {  
                arr[i] = last;  
            } else {  
                arr[i] = arr[i - 1];  
            }  
        }  
    }  
}
```

```
for(int[] tempArr : arr) {  
    for(int temp : tempArr) {  
        System.out.print(temp + " ");  
    }  
    System.out.println();  
}  
}
```

7	8	9
1	2	3
4	5	6

가변배열

- 다차원 배열에서 마지막 차수의 크기를 지정하지 않고 각각 다르게 지정.

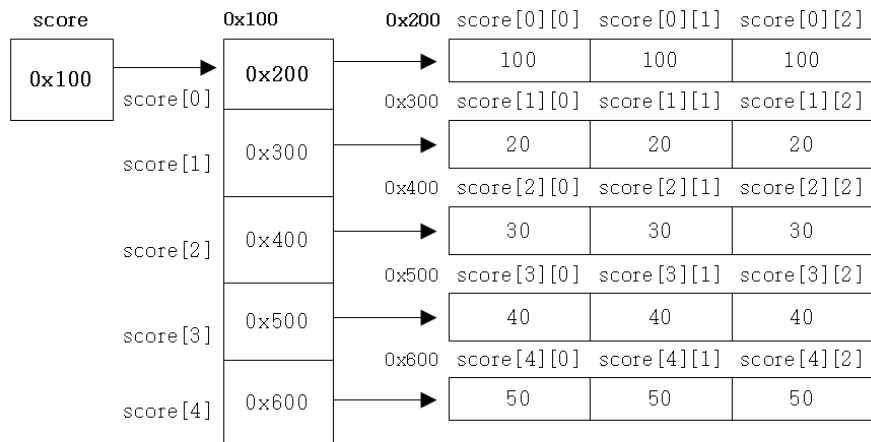
```
int[][] score = new int[5][3];    // 5행 3열의 2차원 배열을 생성한다.
```

```
int[][] score = new int[5][];
score[0] = new int[3];
score[1] = new int[3];
score[2] = new int[3];
score[3] = new int[3];
score[4] = new int[3];
```

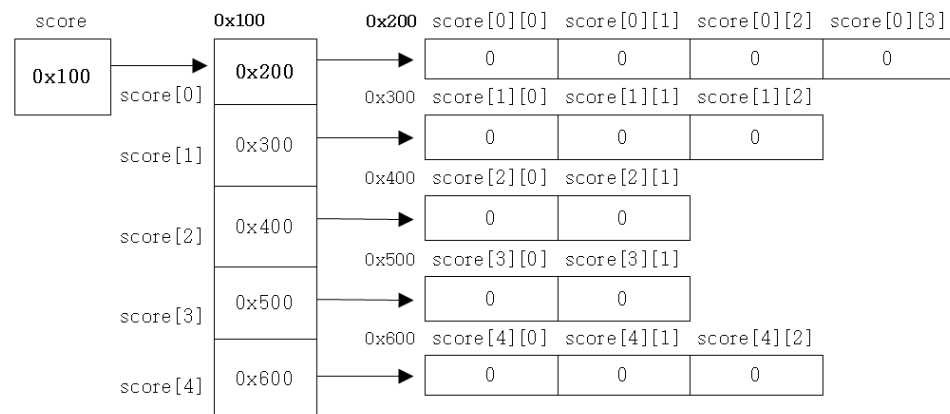
```
int[][] score =
{
    {100, 100, 100},
    { 20, 20, 20},
    { 30, 30, 30},
    { 40, 40, 40},
    { 50, 50, 50},
};
```

```
int[][] score = new int[5][];
score[0] = new int[4];
score[1] = new int[3];
score[2] = new int[2];
score[3] = new int[2];
score[4] = new int[3];
```

```
int[][] score =
{
    {100, 100, 100, 100},
    { 20, 20, 20},
    { 30, 30},
    { 40, 40},
    { 50, 50, 50},
};
```



[그림 5-2] 2차원 배열



[그림 5-3] 가변배열