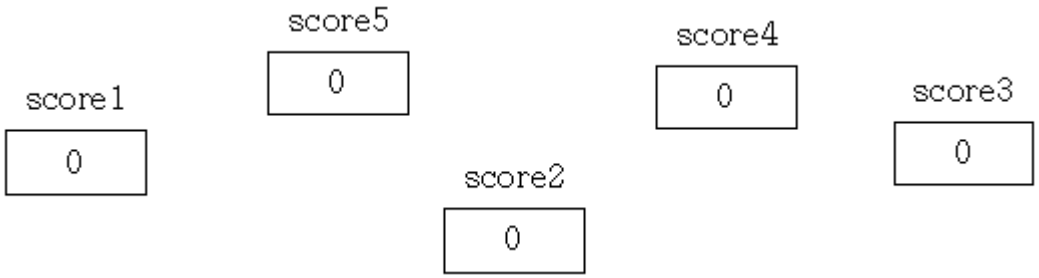
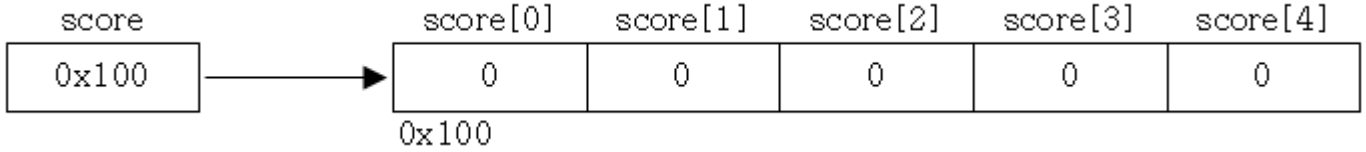
■ 배열(array)

- 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

● 중복된 변수 선언 줄이기 위해 사용

● 반복문 이용해 요소들을 쉽게 처리

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

```
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.

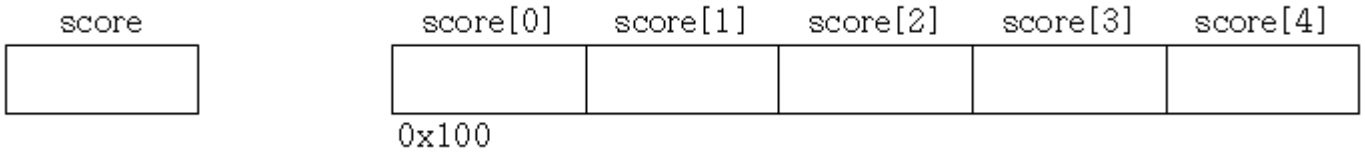
선언방법	선언 예
타입[] 변수이름;	**int[] score;** **String[] name;**
타입 변수이름[];	int score[]; String name[];

■ 배열의 선언과 생성

● 배열을 선언한다고 해서 값을 저장할 공간이 생성되는 것이 아니라 배열을 다루는데 필요한 변수가 생성된다.

```
int[] score;         // 배열을 선언한다.(생성된 배열을 다루는데 사용될 참조변수 선언)
score = new int[5]; // 배열을 생성한다.(5개의 int값을 저장할 수 있는 공간생성)
```

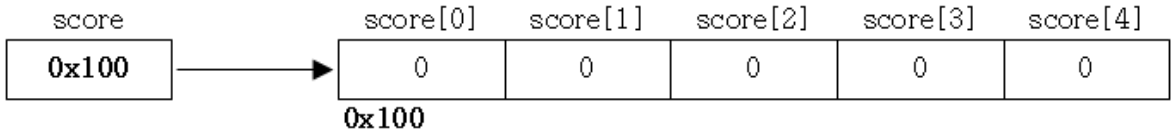
[참고] 위의 두 문장은int[] score = new int[5];와 같이 한 문장으로 줄여 쓸 수 있다.



■ 배열의 초기화

● 생성된 배열에 처음으로 값을 저장하는 것

```
int[] score = new int[5]; // 크기가 5인 int형 배열을 생성한다.
score[0] = 100;           // 각 요소에 직접 값을 저장한다.
score[1] = 90;
score[2] = 80;
score[3] = 70;
score[4] = 60;
```




```
int[] score = { 100, 90, 80, 70, 60};          // 1번
int[] score = new int[]{ 100, 90, 80, 70, 60}; // 2번
```

```
int[] score;
score = { 100, 90, 80, 70, 60}; // 에러 발생!!!

int[] score;
score = new int[]{ 100, 90, 80, 70, 60}; // OK
```

```
int add(int[] arr) { /* 내용 생략 */}

int result = add({ 100, 90, 80, 70, 60});          // 에러 발생!!!
int result = add(new int[]{ 100, 90, 80, 70, 60}); // OK
```

■ 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(socre[i]);
}
```

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

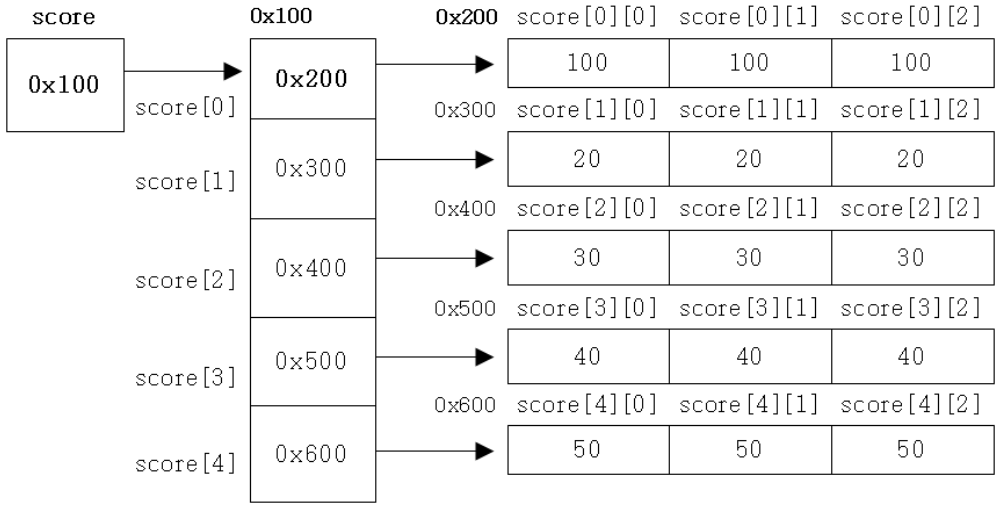
[표5-3] 2차원 배열의 선언

```
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;
    }
}
```



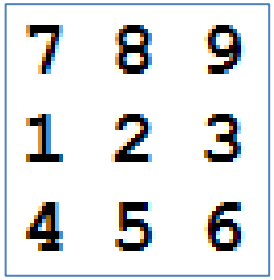
[그림5-2] 2차원 배열

■ 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();
        }
    }
}
```



■ 가변배열

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

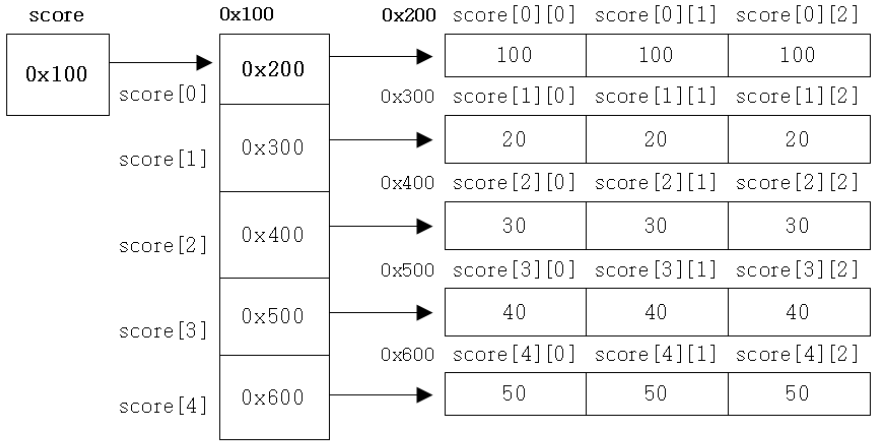
```
int[][] score = new int[5][3];      // 5행 3열의 2차원 배열을 생성한다.
```

```
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];
```

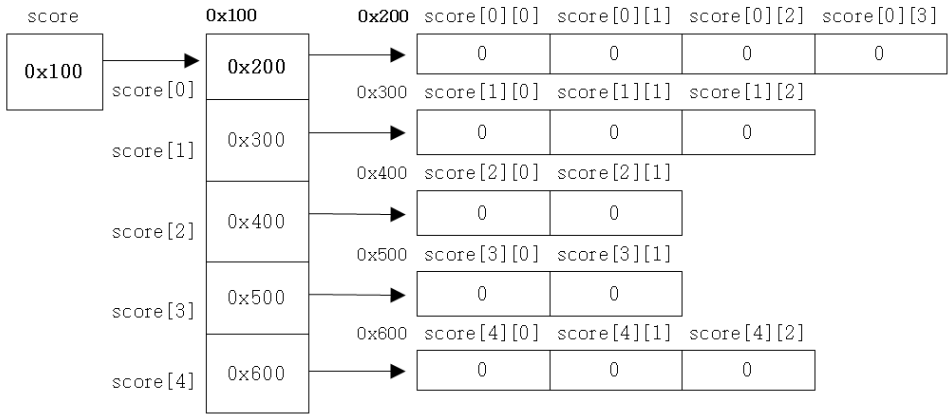
```
int[][] score =
    {
        {100,100,100},
        { 20, 20, 20},
        { 30, 30, 30},
        { 40, 40, 40},
        { 50, 50, 50},
    };
```

```
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},
    };
```



[그림5-2] 2차원 배열



[그림5-3] 가변배열