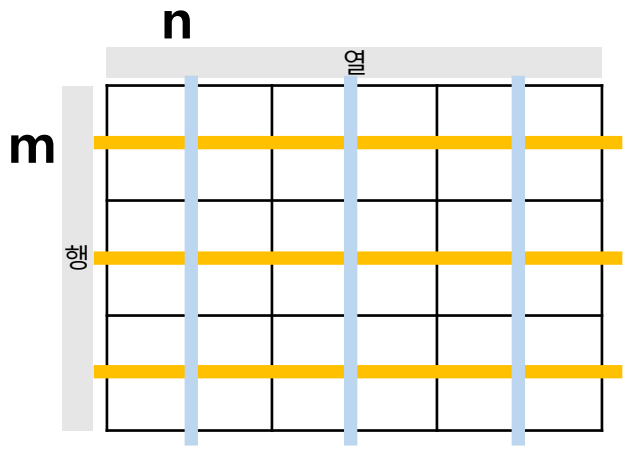
2차원 배열

▶ 2차원 배열

자료형이 같은 1차원 배열의 묶음으로 배열 안에 다른 배열 존재

2차원 배열은 할당된 공간마다 인덱스 번호 두 개 부여 (앞 번호는 행, 뒷 번호는 열 ([0][0]))

✓ 인덱스 값 이해



- m값이 올라가면
  행이 아래로 가고

- n값이 올라가면
  열이 옆으로 이동

▶ 2차원 배열

✓ 참조변수 선언과 배열의 생성

```
int[] arr1;
Arr1 = new int[4]
```


```
int[][] arr2;
arr2 = new int[3][4];
```

▶ 2차원 배열 선언과 할당

✓ 배열 선언

자료형[ ][ ] 배열명 ;

자료형 배열명[ ][ ] ;

자료형[ ] 배열명[ ];

✓ 배열 할당

자료형[ ][ ] 배열명 = new 자료형[행크기][열크기];

자료형 배열명[ ][ ] = new 자료형[행크기][열크기] ;

자료형[ ] 배열명[ ] = new 자료형[행크기][열크기] ;

ex)
```
int[][] arr = new int[3][4];
int arr[][] = new int[3][4];
```

▶ 2차원 배열

✓ 배열의 접근

```
int[][] arr = new int[3][3]
```



```
arr[0][0] = 1;
```



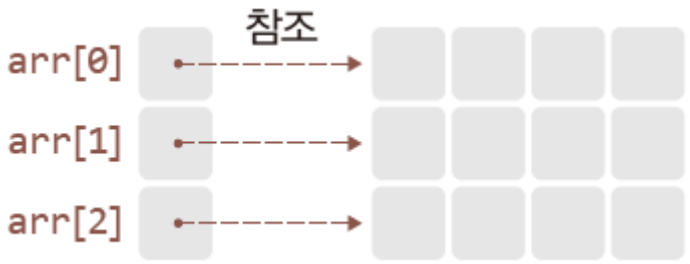
```
arr[1][0] = 5;
```

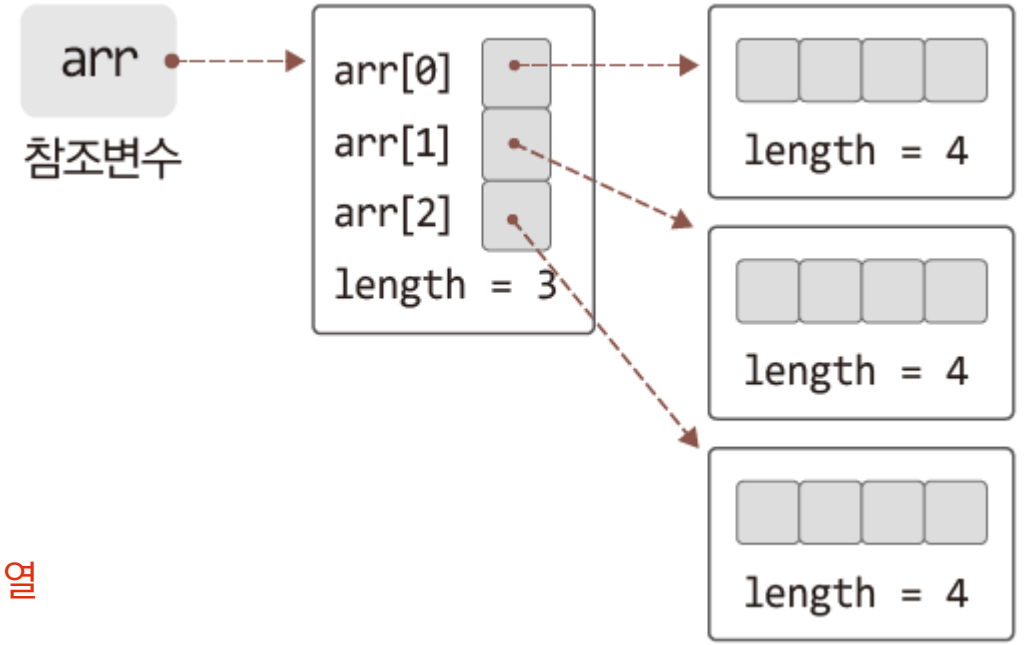

```
arr[2][2] = 9;
```


```
arr[0][1] = 7;
```

2차원 배열의 실제 구조

```
int[][] arr = new int[3][4];
```

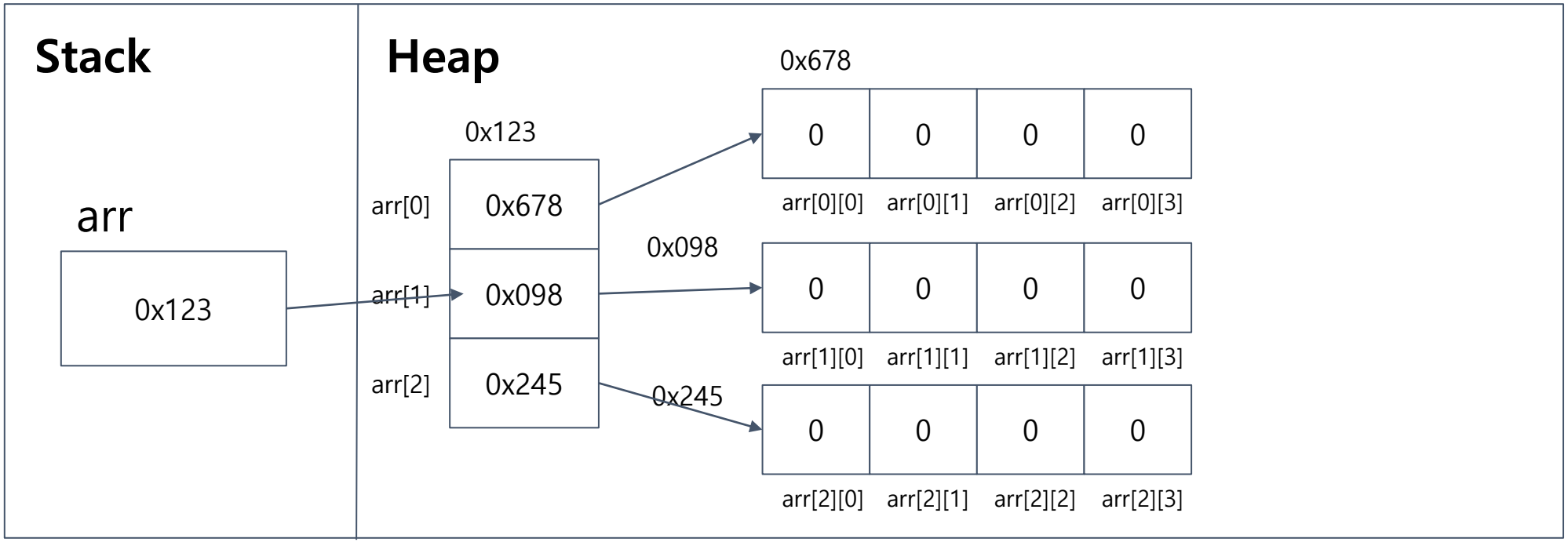




다수의 1차원 배열을 엮어서 구성이 되는 2차원 배열

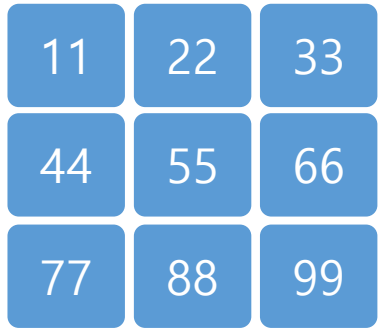
2차원 배열의 실제 구조

```
int[][] arr = new int[3][4];
```

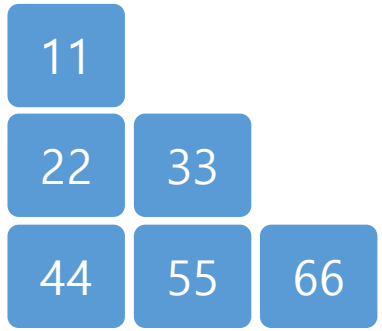


2차원 배열의 초기화

```
int[][] arr = {
   {11, 22, 33},
   {44, 55, 66},
   {77, 88, 99}
};
```




```
int[][] arr = {
   {11},
   {22, 33},
   {44, 55, 66}
};
```



▶ 가변 배열의 선언과 할당

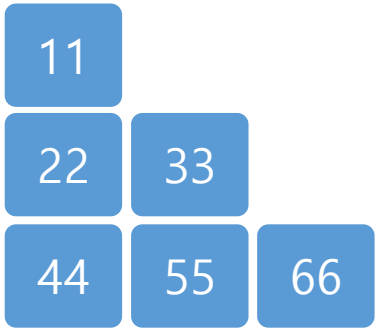
2차원배열선언시 마지막 열크기를 지정하지 않고, 추후에 각기 다른 길이의 배열을 생성함으로써, 고정된 형태가 아닌 보다 유동적인 가변 배열을 구성할 수 있다.

✓ 배열 선언 및 할당

자료형[][] 변수이름 = new 자료형[행크기][];

ex)
```
int[][] arr = new int[3][];
    arr[0] = new int[3];
    arr[1] = new int[2];
    arr[2] = new int[5];
```

```
int[][] arr = {
    {11},
    {22, 33},
    {44, 55, 66}
};
```



2차원 배열의 실제 구조

```
ex) int[][] arr = new int[3][];
    arr[0] = new int[3];
    arr[1] = new int[2];
    arr[2] = new int[5];
```