2D Array Initialization

The 2D array `a2d` could be declared and initialized like this:

```
int a2d[2][3] = {
    {5, 19, 3},
    {22, -8, 10}
};
```

Each row appears is in its own set of curly braces. Because, the array is actually laid out in a linear fashion, you may **omit the inner braces** all together, but that is not as clear:

```
int a2d[2][3] = {5, 19, 3, 22, -8, 10};
```

The rules for partial initialization are similar to **1D** arrays: any uninitialized elements are **value initialized to `0`**. With embedded braces, partial initialization is on a **row-by-row** basis; if you omit them, the rows are ignored. These examples use the **same initial values**, but produce quite different results.

```
int a2d[2][3] = {
    {5},
    {22, -8}
};
```

	0	1	2
0	5	0	0
1	22	-8	0

```
int a2d[2][3] = {
    5, 22, -8
};
```

	0	1	2
0	5	22	-8
1	0	0	0

When initializing, you **may omit the first explicit dimension**, but the second dimension **`[3]` is required**. The compiler must **know how big** each element of `a2d` is.

```
int a2d[ ][3] = {
    {5, 19, 3},
    {22, -8, 10}
};
```