ECE 220 Computer Systems & Programming

Strings and Multi-dimensional Arrays

ECE ILLINOIS

ILLINOIS

Strings

Allocate space for a string just like any other array:

```
char outputString[16];
```

Space for string must contain room for terminating zero.

Special syntax for initializing a string:

```
char outputString[16] = "Result = ";
```

…which is the same as:

```
outputString[0] = 'R';
outputString[1] = 'e';
outputString[2] = 's';
...
```

Null terminating strings – '\0' special sequence that corresponds to the null character.

String Literals

- "This is a string literal"
- Also acts as a null-terminated character array
- You may not change it (type is `const char *`)
- Multiple copies of the same string **may** be co-located

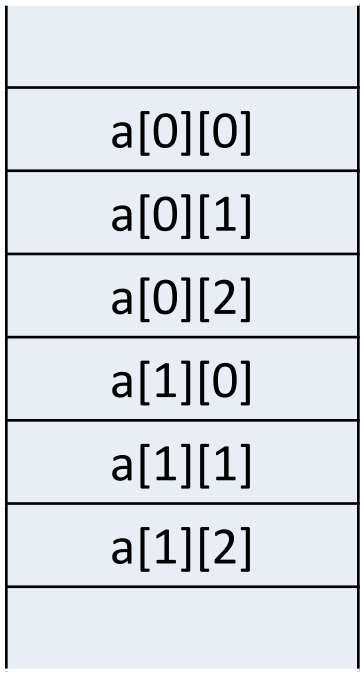
- Except for initialization, cannot assign char[] from string literal
- `==` is pointer comparison, not content

Multi-dimensional Arrays

int a [2][3];

	Column 0	Column 1	Column 2
Row 0	a[0][0]	a[0][1]	a[0][2]
Row 1	a[1][0]	a[1][1]	a[1][2]

In memory



* multi-dimensional array is stored in **row-major** order

Initialize Multi-dimensional Array

```
int a[2][3] = {{1, 2, 3}, {4, 5, 6}};
```

or

```
int a[][3] = {{1, 2, 3}, {4, 5, 6}};
```

or

```
int a[2][3] = {1, 2, 3, 4, 5, 6};
```

Initialize Multi-dimensional Array

```
int a[2][3] = {{1, 2, 3}, {4, 5, 6}};
```

or

```
int a[][3] = {{1, 2, 3}, {4, 5, 6}};
```

or

```
int a[2][3] = {1, 2, 3, 4, 5, 6};
```

Note: when you declare an array, you must either specify a size or provide an initializer.
Multi-dimensional arrays must provide a size for all dimensions except the most significant one.