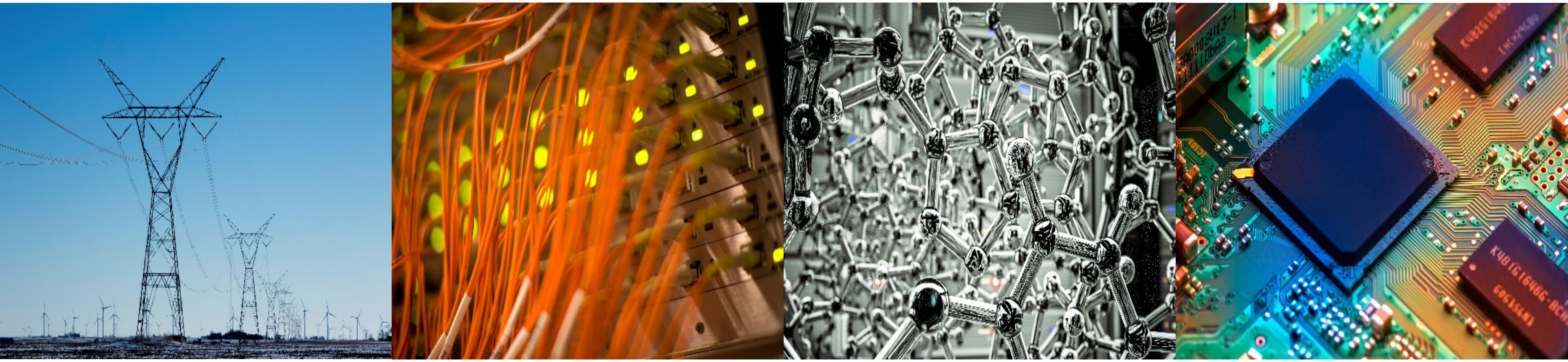
ECE 220 Computer Systems & Programming

Lecture 17 – Strings and Multi-dimensional Arrays

July 9, 2020



ILLINOIS
Electrical & Computer Engineering
GRAINGER COLLEGE OF ENGINEERING

- MT1 score is released on Gradescope
- Regrade deadline: 10pm Central Time on July 10th

Arrays Recap

- Array of size **n** has indices **0, 1, … n-1**
- Array is **pass by reference** (pointer to the first element) in C
- Importance of array bounds checking

```
int array[3] = {1,3,5};
int *ptr = array; /* same as ‘int *ptr = &array[0];’ */
int i;
for (i=0; i<3; i++, ptr++){
   *(ptr + 1) = *ptr + 1;
}
```

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';
...
```

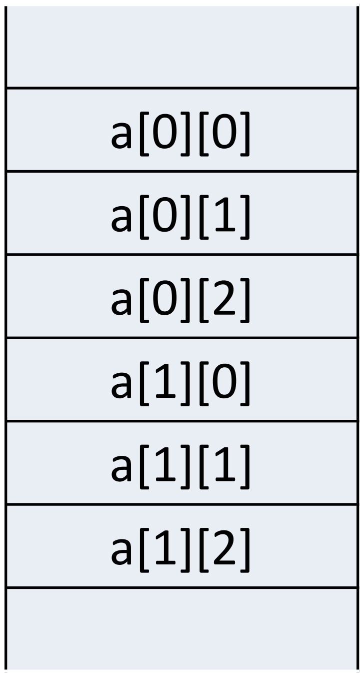
Null terminating strings – '\0' special sequence that corresponds to the null character.

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



a[0][0]
a[0][1]
a[0][2]
a[1][0]
a[1][1]
a[1][2]

* 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};
```

Exercise: implement a function that interchanges two rows of a 5x5 matrix. The functions takes three arguments: pointer to the matrix, row number x and row number y.

```
#define N 5
void row_interchange(int matrix[N][N], int row_x, int row_y){











}
```