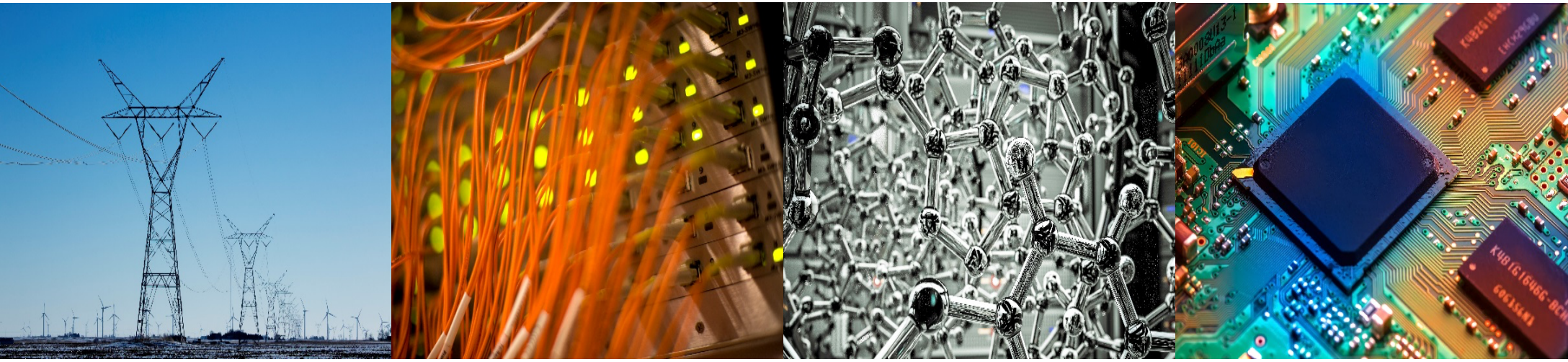
ECE 220 Computer Systems & Programming

Lecture 16 – Pointers and Arrays

July 8, 2020



ILLINOIS
Electrical & Computer Engineering
GRAINGER COLLEGE OF ENGINEERING

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

Pointers and Arrays

Pointer

- Address of a variable in memory
- Allows us to indirectly access variables (in other words, we can talk about its **address** rather than its **value**)

‘ & ’ address operator: &x - returns the address of variable x

‘ * ’ indirection (dereference) operator: *ptr - returns the value pointed to by ptr

Array

- A list of values arranged *sequentially* in memory
- Example: a list of telephone numbers
- Indices **start from 0**: **a[4]** refers to the 5th element of the array **a**

Arrays

- Allocate a group of memory locations: character string, table of numbers

- Declaring and using Arrays

```
int grid[10] = {0,1,2,3,4,5,6,7,8,9};
grid[6] = grid[3] + 1;
int i;
for(i=0;i<2;i++){
   grid[i+1] = grid[i] + 2;
}
```

Passing Array as Argument

Arrays are passed by reference in C

- the address of the array (i.e., address of the first element) is written to the function's activation record
- otherwise, would have to copy each element

```
int main(){
  int array[10];
  int result;
  result = average(array, 10);
  return 0;
}

int average(int array[10], int size);
/* int average(int array[], int size); */
/* int average(int *array, int size); */
```

Pointer Array Duality

```
char word[5];
char *cptr;
cptr = word;
```

cptr	word	&word[0]
(cptr + n)	word + n	&word[n]
*cptr	*word	word[0]
*(cptr + n)	*(word + n)	word[n]

Exercise: implement a function to reverse an array

/*array_reverse(): reverses an integer array, such that the first element will become the last element, the second element will become the second to last element and so on. This function takes two arguments: a pointer to an integer array and its size.*/

```
void array_reverse(int array[], int n){
```

```
}
```