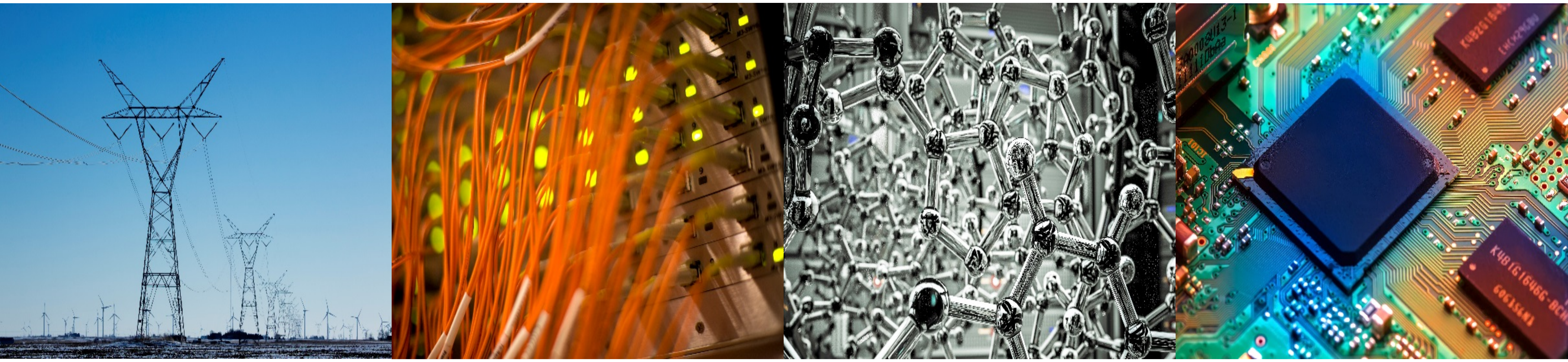
ECE 220 Computer Systems & Programming

Lecture 12 – Strings and Multidimensional Arrays

March 9, 2021



- MT1 is released on Gradescope, regrade request due this Thursday at 10pm CT
- Quiz 2 is available on PrairieLearn through Wednesday
- Informal Early Feedback

ILLINOIS
Electrical & Computer Engineering
GRAINGER COLLEGE OF ENGINEERING

Pointers & Array Recap

- Pass by Value vs. Pass by Reference in C (assume x and y are declared as int)

```
swap(x,y);    swap(          ,          );
```

- Double Pointer – a pointer to another pointer

```
int var=3;
int *var_ptr;
int **var_pptr;
var_ptr =            ;
var_pptr =           ;
```

- Array of size **n** has indices **0, 1, … n-1**
- Array is **passed 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;
}
```

Pointer Array Duality

```
char word[10];
char *cptr;
cptr = word; /* assign cptr to point to 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: pointer to an integer array and array size. */

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











}
```

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.

I/O with Strings

printf and scanf use "%s" format character for string

printf -- print characters up to terminating zero

```
char outputString[20] = ”ECE 220 Roster";
printf("%s", outputString);
```

scanf -- read characters until whitespace,
store result in string, and terminate with zero

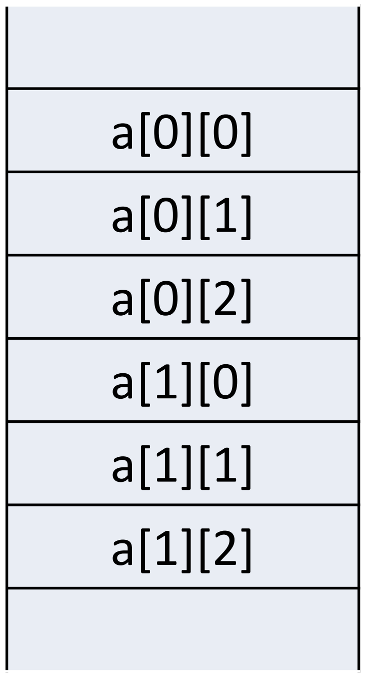
```
char inputString[20];
scanf("%s", inputString);
```

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 in memory

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 an NxN matrix. The functions takes three arguments: pointer to the matrix, row number x and row number y.

```
#define N 4 /* size */
void row_interchange(int matrix[N][N], int row_x, int row_y){















}
```

String Example

- Roll three six-sided dice
- Use the numbers as indices to pick characters from a string
- User will guess three random characters
- Check how many characters matched

Support Functions:

roll_dice /* generate three random numbers */

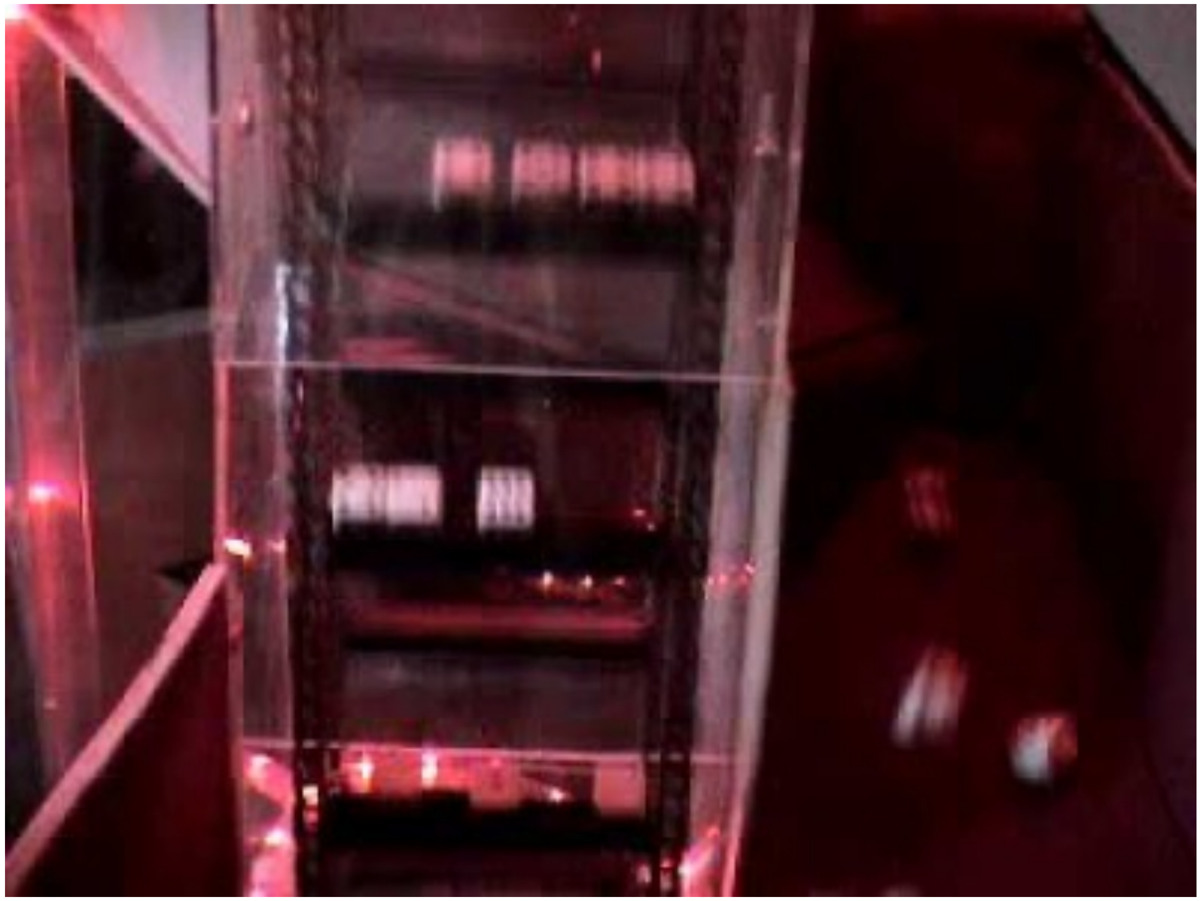
pick_char /* use generated random numbers to pick chars from string */

comp_char /* compare user guesses with solution */

➢ How do we generate random numbers between 1 and 6? How about between 14 and 22?

How to generate random numbers

- If you are really serious
- Pseudorandom numbers generators compute a *deterministic* sequence of numbers that looks random from a starting seed
    - 5 7 1 1 6 4 …
- Important to set a different seed for the pseudorandom generator


Scott Nesin's Dice-O-Matic from GamedByEmail generates 1.3 million rolls per day