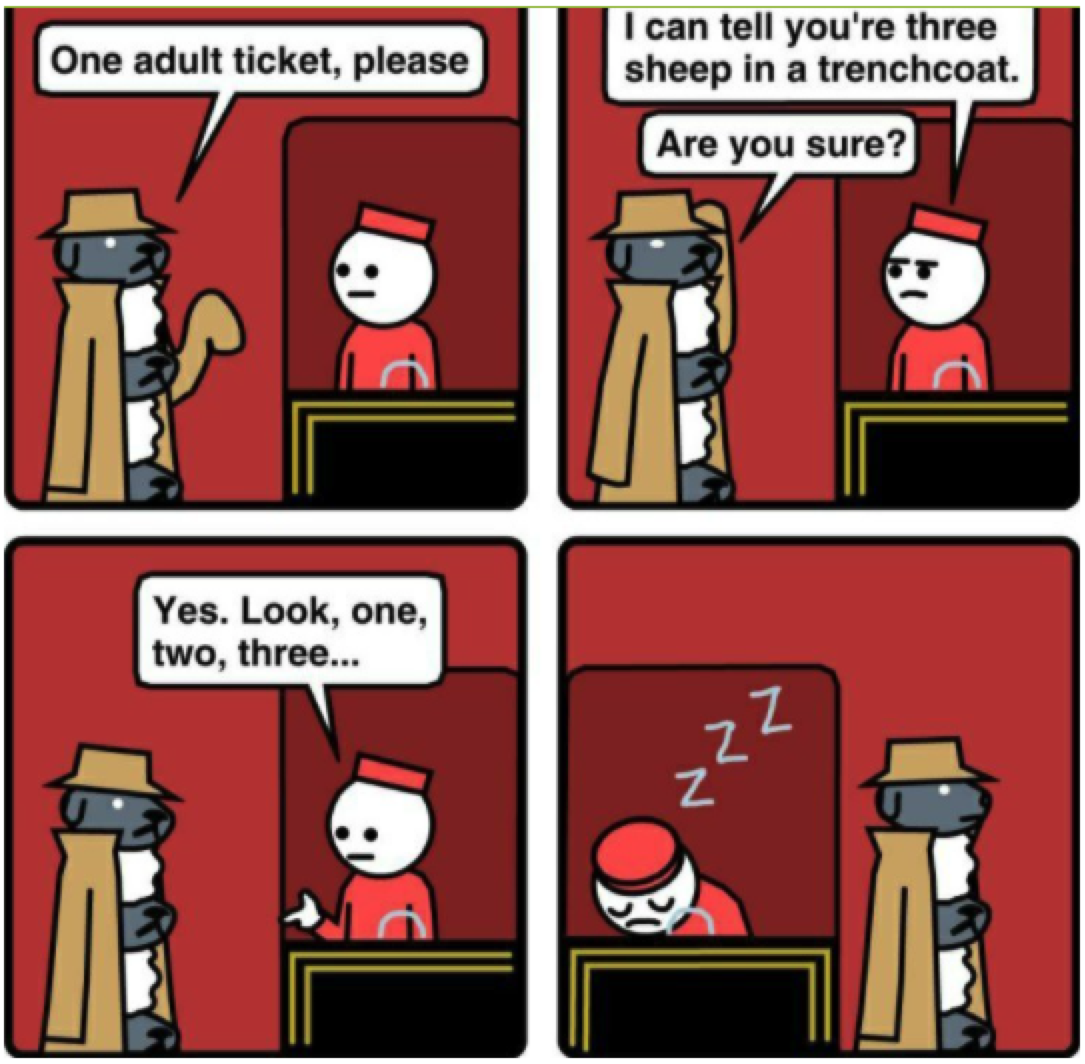
Week 6


One adult ticket, please
I can tell you're three sheep in a trenchcoat.
Are you sure?
Yes. Look, one, two, three...
z z z

Announcements

- Project 5
    - Monday, November 23rd
- Midterm creeping up
    - Tuesday, November 24th

Questions?

- Anything?

C Strings

```
// include a library
#include <cstring>

// initialize a c string for "sheep" (hint: think arrays)
char sheep1[6] = "sheep";
char sheep2[6] = {'s', 'h', 'e', 'e', 'p', '\0'};

// what does the empty string look like?
char empty1[1] = "";
char empty2[1] = { '\0' }

// loop through the string
for (int i = 0; i < strlen(sheep1); i++) {
for (int i = 0; sheep1[i] != '\0'; i++) {
  cout << sheep1[i];
  sheep.at(i) // can I do this? no
}

// can I get user input?
cin.getline(sheep1, 5);  // new way to use getline
// getline(cin, sheep1); // wrong for C Strings!

// if you leave out the null byte and cout, your program will have undefined
// behavior
// in memory: | s | h | e | e| p | ! |
cout << sheep1;
```

Helpful Functions

```
char t[10] = "sheep";
char s[10] = "";

s = t // will this work? no

// Copy
strcpy(s, t);
strcpy(s, "sheep"); // can do this
strcpy(s, 's');     // can't do this
```

```
// Length
strlen(s);

// more efficient way to iterate, if your string doesn't change length in the loop
int n = strlen(sheep1);
for (int i = 0; i < n; i++) {
  cout << sheep1[i];
}

// Concatenate
string s1 = "hi";
string s2 = s1 + "!";

char[10] s3 = "hi";
strcat(s3, "!!!!");
strcat(s3, "!!!!!!!!!"); // overwrites the null byte

// Compare
char a[50] = "friday the 13th";
char b[50] = "friyay the 13th";

// returns int
// result is with respect to a
int r = strcmp(a, b)
if (r > 0)
  cout << "a is greater" << endl;
else if (r < 0)
    cout << "a is less" << endl; // will output this because 'd' < 'y'
else if (r == 0)
    cout << "a is equal" << endl;
```

Arrays of C Strings

```
// get user input of 5 colors, stored as C-strings
// use constants so you get used to the +1 in the array size
const int NUM_STRINGS = 5;
const int MAX_LENGTH = 50;
char colors[NUM_STRINGS][MAX_LENGTH+1]

// get 5 strings as input, store them in colors
for (int i = 0; i < NUM_STRINGS; i++) {
  cin.getline(colors[i], MAX_LENGTH)
}

// if red is in the list, output sus and which position they're in
char sus[MAX_LENGTH+1] = "red";
int i = 0; // by initializing i here, we can access it outside the for-loop
for (; i < NUM_STRINGS; i++) {
  if ( strcmp(colors[i], sus) == 0 ) {
    break
  }
}

cout << "Red is position: " << i << endl;
```

Pass 2D Arrays to Functions

```
/*
Screen is 5 rows, 7 columns
0000000    0000000
0000000    00aaa00
0000000 -> 00a0a00
0000000    00aaa00
0000000    0000000
*/

// initialize a "screen" of all '0's (the character 0, not the integer)
const int ROWS = 5;
const int COLS = 7;
char screen[5][cols+1] = {"0000000", "0000000", "0000000", "0000000", "000000"};

// write the function prototype to take this as input
int drawSquare(char screen[][COLS+1], int nStrings);

// leave the first dimension size empty
// the size of all other dimensions needs to be known at compile-time
// pass an argument that represents how many elements (just like with 1D arrays)
void foo(char a[][5][5], int nStrings);
void bar(char a[], int nStrings);
```