Character Arrays

(a.k.a. C-style strings)

(a.a.k.a. those things that Portland State made me use for the entirety of CS161 while you guys get to use strings 😒)

NULL

- NULL just means 0 or, more specifically, **0x000000000000**
- Fittingly, NULL has **ASCII value 0**.
- All of your pointers should be pointing to NULL when you aren’t using them

```
void* ptr = NULL;
```

A brief detour: Assert statements

- Assertions are kinda like if statements…

```
#include <assert.h>
```

- ...only they stop your whole program, mid-execution, if they are false.

```
assert(arr != NULL);
```

Terminating NULL `== '\0'`

- Signals the end of a character array.
- It must be there in order for your char array to behave properly with the iostream library.

```
char charray[6] = {'M', 'e', 'm', 'e', 's'};
```

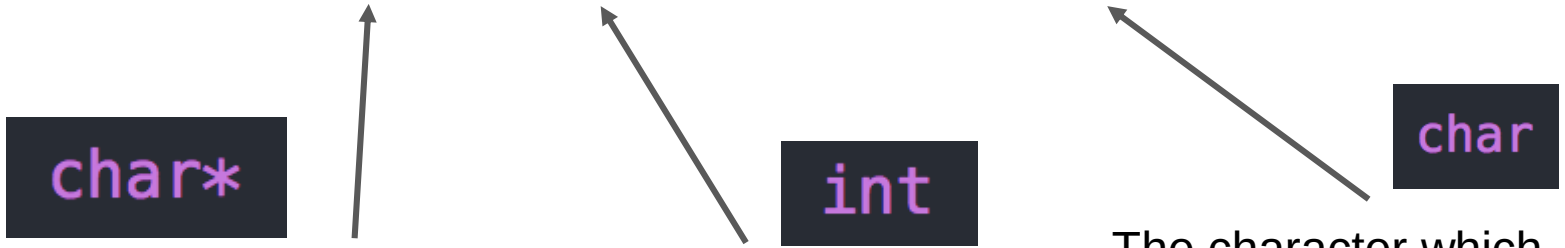
‘M’	‘e’	‘m’	‘e’	‘s’	‘\0’

You must define your char arrays to be 1 longer than the number of elements that you want to use!!!

cin.get();

- Technically speaking: This function gets a single char from the input buffer and returns it.
- If the input buffer is empty, cin.get() will hang your program and wait for a keystroke followed by <enter>.
  - Unless there is already something in the input buffer, in which case it will just grab that.
- (‘\n’)s ARE LEFT IN THE INPUT BUFFER BY CIN.GET()!

cin.get(*array*, *length*, *delimiting character*);



The array you wish to read data into, preferably initialized to all \0s.

The length of the array, including space for \0 (so # elements + 1)

The character which terminates reading from input buffer. (Usually '\n', but could be anything.)

- Unlike cin >>, cin.get(*array*, *length*, *delim*) will allow you to enter a string with spaces!

Input Buffer

- The input buffer holds all incoming information before it is processed by the CPU.
- In layman’s terms, it’s where your keystrokes are stored (or at least their ASCII values).

- Clearing the input buffer:

```
while (cin.get() != '\n');
```