Goals for today

- Everything you wanted to know about C-strings (but were afraid to ask)
- Char/ascii
- String constants, string as “abstraction” (albeit a leaky one)
- C library functions <string.h>
- Under the hood: C-string as pointer/array of char
- Array indexing vs pointer arithmetic
- Ask questions today (and every day!)
Character data

**char is 1 byte**
(by definition in standard, never need to compute sizeof(char))

**char may be signed or unsigned by default**
Of consequence when promoting to larger type but not much else

**Standard ASCII maps 0x00 - 0x7f to letters, digits, punct**
man ascii to display table
8th bit used as parity/error check in some situations
No consensus for characters mapped to 0x80-0xff

**Unicode**
more robust/flexible but more complex, larger storage needs

**char operations**
single byte integer, man isdigit
int my_isdigit(int ch) {
    return ch >= '0' && ch <= '9';
}

int my_isxdigit(int ch) { // from musl
    return my_isdigit(ch) || ((unsigned)ch|32)-'a' < 6;
}

How to represent sequence of chars?

Stuff into a 4-byte or 8-byte word?
Array/vector/list of char?

C language features generally map straight to machine structures

We saw this with behavior/operations of integers
How does underlying system support sequence of anything?

C-string is a simply a pointer!

Characters stored in sequential memory locations
Null char used for termination

char *str = "binky";

What can you do with a C-string?

Read/write individual chars, e.g. str[index]
Man string for library functions
size_t my_strlen(const char *s)
{
    size_t count = 0;
    while (s[count] != '\0')
        count++;
    return count;
}

char *my_strcpy(char *dst, const char *src)
{
    char *result = dst;
    while ((*dst++ = *src++));
    return result;
}

char *my_strncpy(char *dest, const char *src, size_t n)
{                                               // from man page
    size_t i;

    for (i = 0; i < n && src[i] != '\0'; i++)
        dest[i] = src[i];
    for ( ; i < n; i++)
        dest[i] = '\0';
    return dest;
}
Let's code!

/afs(ir/class/cs107/samples/lect4/pig.c

Pig latin: be => ebay
    trash => ashtray
    one   => oneway

Helpful string.h functions to consider:
    strcspn
    strcat, strncat