Strings

- Strings are not a built-in data type.
- C provides almost no special means of defining or working with strings.
- A string is an array of characters terminated with a “null character” ("\0")
String literals

char *name = "csc209h";
printf("This is a string literal\n");

• String literals are stored as character arrays, but you can't change them.

name[1] = 'c'; /* Error */
• The compiler reserves space for the number of characters in the string plus one to store the null character.
String Variables

- arrays are used to store strings
- strings are terminated by the null character ("\0") (That's how we know a string's length.)
- Initializing strings:
  - char course[8] = "csc209h";
  - char course[8] = {'c','s','c','2','0','9','h','\0'};
  - course is an array of characters
  - char *s = "csc209h";
  - s is a pointer to a string literal
Warning!

• Big difference between a string's length and size!
  – **length** is the number of non-null characters currently present in the string
  – **size** if the amount of memory allocated for storing the string

• Eg., `char s[10] = "abc";`
  – length of `s = 3`, size of `s = 10`
  – ensure length+1 ≤ size!
String functions

• The library provides a bunch of string functions which you should use (most of the time).

$ man string

int strlen(char *str)

– returns the length of the string. Remember that the storage needed for a string is one plus its length
Copying a string

char *strncpy(char *dest,
               char *src, int size)
– copy up to size bytes of the string pointed to by src
   in to dest. Returns a pointer to dest.
– Do not use strcpy (buffer overflow problem!)

char str1[3];
char str2[5] = "abcd";
/*common error*/
strncpy(str1, str2, strlen(str2));/*wrong*/
Concatenating strings

char *strncat(char *s1, const char *s2,
size_t n);
– appends the contents of string s2 to the end of
s1, and returns s1.
– only appends up to n bytes to s1

• Watch out! It is easy to forget how much space is
left.
– char str1[6] = "abc";
– strncat(str1, "def", 6); /*wrong*/
Comparing strings

int strcmp(const char *s1,
           const char *s2)

• compares s1 and s2, returning a value less than, equal to, or greater than 0 depending on whether s1 is less than, equal to, or greater than s2.

if( strcmp(str1, str2) <= 0)
   /* is str1 <= str2? */
NAME
strchr, strrchr - locate character in string

SYNOPSIS
#include <string.h>

char *strchr(const char *s, int c);
char *strrchr(const char *s, int c);

DESCRIPTION
The strchr() function returns a pointer to the first occurrence of the character c in the string s.

The strrchr() function returns a pointer to the last occurrence of the character c in the string s.

RETURN VALUE
The strchr() and strrchr() functions return a pointer to the matched character or NULL if the character is not found.