CSC161 2011S *Imperative Problem Solving*

Class 35: Strings

Held: Wednesday, 6 April 2011

Summary: We begin to consider the design of strings in C.

Related Pages:

- EBoard.
- Lab: Strings.
- Reading: K&R 5.5, B3.

Notes:

- Responses to your questions on the make reading have been posted. Class time discussing them will be limited.
- Missing questions from DG, MH, JTL.
- Reading for Friday: Debugging with DDD.
- EC for today's 's M.C. Escher talk [academic]
- EC for Thursday's nonvo on Iowa Water (or other part of that symposium) [academic]
- EC for Friday's CS Table on Watson [academic]
- EC for Friday's presentation of *Sita Sings the Blues* [academic]
- EC for any activity in the animation festival, including Tiny Circus [peer]
- EC for either part of the baseball doubleheader against Knox [peer]

Overview:

- About strings.
- Important string functions.
- Thinking like a C programmer.
- Lab.

Strings

- In C, strings are null-terminated arrays of characters.
  - "null" means "0".
- Since arrays are (mostly) interchangable with pointers, strings are also pointers to characters.
- While this sounds simple, it's easy to screw things up.

Important String Functions

- `strlen (str)` - find the number of characters in str.
- `strcpy (target, source)` - copy string from source to target.
- `strncpy (target, source, n)` - copy up to n characters from source to target.
- `strcat (target, source)` - put source on the end of target.
- `strcmp (str1, str2)` - compare two strings.

Thinking Like a C Programmer

While I've put `strcpy` in the outline, a reading question suggests that I should do `strcmp` instead.

```
/**
 * strcpy: Copy source (s) to target (t).
 */
void
strcpy (char *t, char *s)
{
  char *result = t;
  while (*t++ = *s++)
    ;
  return result;
} // strcpy
```

Lab

- Do the lab.
