Introduction to Computer Science and Programming in C

Session 13: October 14, 2008
Columbia University
Announcements

- Homework 2 is due.
- Midterm Review next class (10/16).
  Exam on 10/21
- Bert’s office hours today moved to
  Wednesday 10/15, 1-3 PM (or by appointment)
Review

- Bit operators:
  - & , | , ^ , ~ , << , >>
  - and, or, xor, not, left-shift, right-shift
  - Using **masks** to manipulate individual bits
Today

- C Libraries
C Libraries

- In addition to built-in C commands, operators, C installations provide a **standard library** of functions, types, macros.

- The standard library is not considered part of C itself, but all **ANSI C installations** have it.

- **ANSI** - American National Standards Institute
C Libraries

- The standard library allows us to abstract away many machine-specific implementations.

- Using the library, we don’t have to worry about how to implement common functions on different computers.

- We will see specific examples of this as we go through the libraries.
Library Headers

- We access the standard libraries by using the `#include` preprocessor command to include the header of the library.

- For example, our favorite library header:
  ```
  #include <stdio.h>
  ```
stdio.h

- Standard input and output
- FILE, printf(), fprintf(), fscanf(), etc.
- Provides access to keyboard input, terminal output, and file system on any computer
string.h

- strcpy(A,B); /* copy string B into A */
- strcat(A,B); /* put B in A after A (concatenate)* /
- strcmp(A,B); /* check if A is equal to B (compare)* /
- strlen(A); /* returns length of A */
- strtok(A,B);
  /* Useful for splitting long strings into pieces, or tokens. The usage is complicated, so don’t worry about this one for now. */
/* Utility functions to check for types of char's */

isalpha(c); /* check if c is an alphabet character 'a'-'z', 'A'-'Z' */

isdigit(c); /* check if c is digit '0'-'9' */

isalnum(c); /* isalpha(c) or isdigit(c) */

iscntrl(c); /* control char (i.e. \n, \t, \b) */

islower(c); isupper(c) /* lowercase/uppercase */

d = tolower(c); d = toupper(c) /* convert to lowercase or uppercase */
math.h

- Provides the basic scientific calculator functions.

- Often needs to be specially linked when compiling because takes advantage of specialized math hardware in processor.

- gcc -lm myProgram.c -o myProgram
math.h

- sin(x); cos(x); tan(x);
- asin(x); acos(x); atan(x); /*{sin, cos, tan}^(-1)*/
- exp(x); log(x); log10(x);
  /* e^x, natural and base-10 log */
- pow(x,y); /* x^y */
- sqrt(x); /* square root */
- ceil(x); floor(x); /* closest int above or below */
- fabs(x); /* absolute value */
stdlib.h

- Lots of utility functions
  - `atof(<string>); /* convert string to float */`
  - `atoi(<string>); /* convert string to int */`
  - `x = rand(); /* returns a (pseudo) random int between 0 and constant RAND_MAX */`
  - `srand(<unsigned int>); /* seeds rand generator */`
  - `malloc(); free(); /* memory management */`
  - `system(<string>); /* runs string in OS */`
assert.h

- Provides a macro to check if critical conditions are met during your program:
  
  ```
  assert(<boolean expression>);
  ```

  /* if the expression is false, the program will print to stderr:
  Assertion failed: <expression>, file <filename>, line <line number>
  */

- Provides a nice way to test programs.
limits.h + float.h

- Contain various important constants such as the minimum and maximum possible values for certain types, sizes of types, etc.

- CHAR_BIT (bits in a char)
- INT_MAX, CHAR_MAX, LONG_MAX
- INT_MIN, CHAR_MIN, LONG_MIN
- FLT_DIG (decimal digits of precision)
- FLT_MIN, FLT_MAX (min. and max. value of float)
- DBL_MIN, DBL_MAX (and of double precision float)
time.h

- Provides new type to represent time, `time_t`

- `time_t time(NULL); /* returns current time */`

- `time_t clock();
  /* returns processor time used by program since
  beginning of execution */`

- `strftime(A, sizeof(A), "formatted text", <time_t>);
  /* format text with placeholders:
  %a weekday
  %b month
  %c date and time
  %d day of month
  %H hour ...and many more */`
A few more

- stdarg.h - allows you to create functions with variable argument lists (such as printf).
- signal.h - provides constants and utilities for standardized error codes for when things go wrong
- setjmp.h - allows you to jump to anywhere in your code. NEVER use this.
The user’s manual for all the functions are in *The C Programming Language, Appendix B*

(Flip through it to get a feel. Don’t try to read it all)