Introduction to Computer Science and Programming in C

Session 14: October 16, 2008

Columbia University

Announcements

- Midterm Exam on 10/21

Today

- Homework 2 solutions
- Midterm review: Every topic from beginning to C Library

Homework 2 Solutions

- http://www.cs.columbia.edu/~bert/courses/1003/homework2_soln.txt

1. Introduction

- Algorithm – systematic method to solve a problem.
  - Handwritten Addition
- Characteristics of C:
  - high-level: similar to English (low-level would be more similar to machine language)
  - compiled: convert to machine language

2. History and Architecture

- Analog vs. Digital.
  - Analog - numbers represented by analogy
  - Digital - numbers represented by symbols
- volatile memory vs. non-volatile

2. History and Architecture

- Binary representation:
  - bit: 0 or 1
  - byte = 8 bits
  - Base-2 representation
- ASCII: standardized table of mapping from characters to numbers

3. Cunix Tutorial

- Mostly irrelevant for midterm.

4. Variables and Basic Types

- Variables are declared and initialized:
```
int x = 3;
```
- Basic types: int, char, float
- C arithmetic operators: + - * / (not ^)
- Casting: (<new type>) variable:
```
float y = (float) x;
```
- Casting float to int truncates

5. Arrays, strings, i/o

- Array: an ordered group of variables. Also often called a **vector.**

```
int scores[10];
```

  - individual entries are accessed with **index**, which begins at 0 and ends at **size**-1.

```
int x = scores[4];
```

- String: an array of characters, used to store text.

5. Arrays, strings

- The end of a string is marked with a NULL character, written '\0'
  'S' , 'a', 'm', '\0'
- Strings can be read from standard input (stdin) and from command line
- See 5th lecture slides or book for syntax

6. If, loops

- **Control flow**: instead of a linear path through your code, if statements and loops allow you to design multiple paths

- if (<Boolean statement>)
  ...do stuff...
  else
  ...do something else...

- while (<Boolean statement>)

6. If, loops

- `for (<initialization>; <Boolean>; <count>)`

- ```
  switch(<variable>) {
      case <value>:
          ...do stuff...
          break;
      case <another value>:
          ...do stuff...
          break;
      default:
          ...do default stuff...
          break;
  }
  ```
  ```

7. Functions, scope

- Functions allow us to abstract repeated code.
- **arguments:** input values to function
- **return value:** output value of function
- When we **call** a function, we give it **arguments** and it **returns** a response.

7. Functions, scope

- **scope:** area of program where variable is valid
- Variables are only valid within **block**
- **block:** area of code designated by curly-braces

8. Recursion

- When a function calls itself
- Towers of Hanoi: to move N discs,
  1) move N-1 discs out of the way
  2) move bottom disc to target peg
  3) move N-1 discs onto target
- Produces elegant algorithms that are easier to understand

9. More types

- Struct: data structure holding multiple **fields**– Any assortment of other variables.
- Union: block of memory that can hold variables of different types. “multi-purpose
- enum: type with discretized settings, represented with numbers, but numerical value is meaningless (like chars)

10. File I/O

- stdio.h provides the FILE type
- fopen(<FILE>, <mode>);
- fclose(<FILE>);

10. File I/O

Name	Input	Output
fprintf()	formatted text + args	file
printf()	formatted text + args	stdout
sprintf()	formatted text + args	string
fputc(), fputs()	char, string	file
fscanf()	file	formatted text + args
scanf()	stdin	formatted text + args
sscanf()	string	formatted text + args
fgetc(), fgets()	file	(char) int, string

11. C Preprocessor

- Commands that modify your code text before compilation
- #include – copies text from external file
- #define – find and replace
- #ifdef – conditional compilation

12. Bit operations

- Hexadecimal: base-16 counting. One symbol for every four bits.
- bitwise operations perform same operation on each bit independently
- and `&` , or `|` , xor `^` , not `~`
- left shift `<<` fills with zeros
- right shift `>>` fills with sign bit

13. C Libraries

- C libraries provide standardized functions, types macros for portability
- We've used: stdio.h, string.h
- time.h, stdlib.h, ctype.h, math.h, assert.h,
- ...and some more