Course Introduction

Bryce Boe

2013/06/25

CS24, Summer 2013 C

About Me (Bryce Boe)

- Ph.D. Candidate in Computer Science Education
  - Focus on automated assessment
  - Not (yet) deserving of the Doctor/Professor title
    - Just call me **Bryce** or **BBoe**
- B.S. in Computer Science from UCSB 2008
- Background in networking and security
- Second time teaching (CS32 last summer)

How to make class smoother

- Feedback, feedback, feedback
  - “Bryce, *X* doesn’t make sense”
  - “It might be better if Y”
  - “I can’t read your handwriting”
  - “Your going way too fast”

Outline for today

- Review the syllabus
- Piazza Demo
- Review CS16 Material
- Command Line Arguments
- Overview HW1 (Fizzbuzz)
- Demonstrate the submission and feedback process

SYLLABUS REVIEW

Problem Solving with Computers II

- Intermediate building blocks for solving problems using computers. Topics include data structures, object-oriented design and development, algorithms for manipulating these data structures and their runtime analyses. Data structures introduced include stacks, queues, lists, trees, and sets.

Course Info

- Instructor: Bryce Boe
  - – Office Hours
    - Tuesday 2:15 – 3:15 PM, GSL
    - Thursday 11:15 – 12:15 PM, GSL
- TA: Jane Iedemska
  - – Office Hours to be announced
- Website: http://cs.ucsb.edu/~cs24

Required Text

- C++ Plus Data Structures, 5th edition
  - Nell Dale
- Content will supplement lectures

What do you already know?

What you *should* already know

- C
  - Primitive types (int, float, double, char)
  - Loops and conditionals
  - Functions (declaring and calling)
  - Arrays
  - Structures
  - Memory allocation and de-allocation
  - Pointers

Grading Distribution

- 36% Projects (3 or 4)
- 24% Final (Thursday August 29)
- 16% Labs (8)
- 16% Midterm (Thursday July 25)
- 04% Homeworks (2)
- 04% Participation

Participation

- Earned by:
  - Participating in class
  - Answering questions on Piazza
  - Responding to questions on Piazza
  - (Maybe) editing questions and answers for clarity on Piazza
- Participation points are relative to the overall class effort (less positive outliers)

Late Submission Policy

- Grading based off your latest (most recent) submission
- 1% off every 5 minute interval late
- Examples:
  - Submission at 00:00:00-00:04:59, 1% off
  - Submission at 00:45:00-00:49:59, 10% off
  - Submission at 04:05:00-04:09:59, 50% off
  - Submission on or after 08:15:00, 0%

Grading Petitions

- Applies only to tests
- Not required for grading “mistakes”
- Must meet the following conditions:
  - Wait 24 hours after the test was returned to you
  - Provide a written argument that:
    - Clearly states why your answer is suitable for the question
    - Acknowledges your understanding of the *expected* answer
    - Compares the two

Attendance

- Lectures:
  - Strongly encouraged, not required
- Labs:
  - Required for the first lab (unless already notified)
  - Encouraged but not required for subsequent labs

Academic Integrity Discussion

- Break into groups of 4 or 5
- Discuss the following questions:
  - What constitutes a violation of academic integrity?
  - What sort of collaboration between students are acceptable?
  - Why are we having this discussion?

Course Syllabus

- The official course syllabus is viewable on the course website:
  - http://cs.ucsb.edu/~cs24
- It will be updated as necessary

Online Interaction

- Avoid class-related emails
- Class discussion and online interaction to take place on Piazza
  - https://piazza.com/class#summer2012/cs32
- Piazza allows:
  - You to ask questions anonymously
  - Ask questions privately to the instructor and TA
  - You to respond to questions
  - Edit questions and answers

Piazza Demo

- https://piazza.com/class#summer2013/cs24/home

CS16 REVIEW

Arrays

- Reference cs16_review_arrays.c

- Unitialized:
  - int foo[16];
  - char bar[1024];
- Fully initialized
  - int blah[] = {1, 2, 3, 0xDEADBEEF, 0b1010};
  - char msg[] = “hello world”;
  - char other[] = {‘a’, ‘b’, 99, ‘d’, ‘e’};

Structures

- Structures (struct keyword) allows you to define your own types (see cs16_review_struct.c)

```
struct Point {
  int x;
  int y;
  char *name;
};

struct Point p1;  // Uninitialized
struct Point p2  = {16, 32, “some point”};  // Initialized
```

Pointers

- A primitive type that stores the address to where the *data* is actually stored in memory
- When accessing elements of a **struct**, use `->` to automatically dereference the object

```
struct Point *p1 = malloc(sizeof(struct Point));
(*p1).name = “some name”;
p1->name = “some name”;  // the same as above
free(p1);  // Always free the memory when done
```

Memory Layout

- Function-level is stored on the **stack** in local memory (created when the function is called, and destroyed when the function returns)
- Memory returned by *malloc*/*calloc*/*realloc* comes from the heap (dynamic memory) and must be managed manually by calling *free*

C-strings

- C-strings are an array of characters followed by ‘\0’ (0b0000)
- char local_string[] = “hello world”;
- char manual_string[] = {‘a’, ‘b’, ‘c’, ‘\0’};
- char not_a_cstring[] = {‘x’, ‘y’, ‘z’};
- char *dynamic_string = “hello world”;

Exit Statuses

- The exit status is used to indicate success (0) or failure (non-zero)
- Exit status is set from the return value from main
- Exit status is non-zero when the program segfaults or crashes for other reasons
- After executing a command on the terminal (./a.out) run `echo $?` to get the previous command's exit status

COMMAND LINE ARGUMENTS

int main(int argc, char *argv[])

- argc – contains the number of arguments to the program, i.e., the length of argv
- argv – an array of c-strings (char*), contains argc elements
- Examples
  - ./a.out (argc ← 1, argv ← {“a.out”}
  - ./submit -p CS24_m13:29 fizzbuzz.c
    - argc ← 4, argv ← {“submit”, “-p”, “CS24_m13:29, fizzbuzz.c}

Homework 1, FizzBuzz

- http://cs.ucsb.edu/~bboe/cs24_m13/p/hw1

Submitting your work and receiving feedback

- Refer to http://cs.ucsb.edu/~bboe/cs24_m13/p/submission

For Thursday

- Complete HW1
- Begin reading Chapter 1 in the textbook

Questions?