CSSE 132 – Introduction to Systems Programming
Rose-Hulman Institute of Technology

Exam 2 Review Guide

This exam measures your mastery of these learning objectives:

1. Describe the functions of common computer system hardware elements including CPU, memory hierarchy and input/output devices.
2. Implement and analyze software in the C programming langauge using:
   - Standard C data types
   - Binary arithmetic, boolean and logical operations
   - Functions
   - Arrays
   - C Strings
   - Pointers and Pointer Arithmetic
   - Static and Dynamic memory allocation techniques
   - User and file input/output
   - Command-line arguments
3. Discuss why certain abilities such as information representation, network communication, input/output, and security require support from multiple layers of a computer system.
4. Demonstrate ability to perform tasks like these in a variety of operating environments including the Linux system environment:
   - compile software
   - debug software
   - secure files
   - leverage a version control system
   - manipulate data
   - command-line (shell) navigation and manipulation

1 Format

Two-part exam. 2h50m total time (lab period)

- Part 1 (paper). Written problems. **Closed resources**. You are allowed to use one single-sided 8 1/2 by 11 inch sheet of hand-written notes only.
- Part 2 (coding). You are allowed to use only these acceptable resources:
  - Your computer
  - Your assignments and labs submitted in your individual repository for this term
  - The CSSE 132 course website and things directly linked from it

You are not allowed to use other Internet resources, instant messaging, your smartphone, or other communication means during any part of the exam. Use of other resources is considered academic dishonesty and will result in a penalty grade.

2 Topics to study

- Representing data in a binary computer
  - C data types `float`, `int`, `char`, etc.
  - Floating point `float` standard, range and precision
  - C arrays, pointer arithmetic, address (`&`) and dereferencing (`*`)
  - C strings, null-termination, string library functions (`strlen`, `strcpy`, `strcmp`, etc.)
  - C structs: define, allocate, set members, use members
- Functions in C and procedure calls in ARM assembly, conventions for caller and callee in use of registers and stack
- Allocating and using memory on the stack, in C and ARM assembly
- Allocating, using, and freeing memory on the heap (in C), memory leaks
- Creating a C program from scratch that can take command-line arguments
- Input/Output
  - Getting input from a user in C (especially `fgets`)
  - Opening, closing, seeking, reading, and writing files in C

3 Problem-by-Problem Review

Paper Part (50 pts)

1. Problem 1: Multi-choice (8 problems × 2 = 16 pts)
   - 1.1 Floating Point *[Quiz 15]*
   - 1.2 Struct and memory *[Quiz 08]*
   - 1.3 Stack/Heap *[Quiz 12/16]*
   - 1.4 String/pointer *[Quiz 08]*
   - 1.5 `fgets` *[Quiz 20]*
   - 1.6 `argc/argv` *[Quiz 13]*
   - 1.7 String/`malloc` *[Quiz 08/16]*
   - 1.8 Pointer *[Quiz 07]*
2. Problem 2: Stack & Heap (10 pts) *[Sample Exam/Quiz 12/Quiz 16/HW4]*
3. Problem 3: Procedure Call (4 pts) *[Quiz 12 - last page]*
4. Problem 4: `argc/argv` + pointer (10 pts) *[Quiz 13/HW 4 - APPLETAX problem]*
5. Problem 5: Fill in blanks (10 pts) *[Quiz 18/ Lab 5]*

Coding Part (50 pts)

1. `part1.c` (20 pts) *[Sample Exam/HW 5 Coding/HW 6 - expand_str]*
   - 1.1 (8 pts) string and array, conversions
   - 1.2 (6 + 6 pts) Implement one method to generate a new string
     - Write two functions: `xxxx_len, xxxx`
     - Hint: using `memcpy, strncmp`
2. `part2.c` (15 pts) *[Practice Exam/Lab 6 - `getALine`, HW 4/5 coding part]*
   - Create a new file to write a program
   - Read file(s) and output some results
3. `part3.c`: (15 pts) *[Practice Exam/HW 6 - `platin.c`/Lab 6]*
   - Create a new file to write an *interactive* program
   - File I/O: read and/or write to a file