CS 35L
Discussion 1A
Week 8

Final Review

Contents

1.1. Logistic

- When?
  - Check on MyUCLA
- Where?
  - Check on MyUCLA
- Professor will mention more details in his lecture
- Resource
  - Test is open notes
  - **NO** Computer, **NO** Internet
  - **NO** collaboration / communication with others! Finish by yourself !!

1.2. Strategy

- Make use of the 3 hrs you have!
    - Check how many points (minutes) the exam says for each questions
        - Usually Qs with fewer points are okay to have shorter questions
        - Usually Qs with more points will take longer time
        - Explain more details may earn you more points if you are not that confident of your answers
    - Easy ones first:
        - First answer as many questions as you easily can
        - Then come back later to the hard ones
    - Don't worry about the perfect answers:
        - Try to move on and get as many points as you can
- For hard questions:
    - Try to focus more on the overall approach for more points, then come back for small details (which might be worth fewer points)
    - If Qs ask you to **explain**, try to focus more on **concrete justification** for your answer
    - If Qs ask you to give examples, try to make your example concrete on input/output and give some explanation if the Qs are of high points

1.2. Strategy

- Review your HWs/Labs
- Make use of the resources (personal suggestions)
  - Print out / summarize some resources
    - Summaries: basic **concepts** Prof. mentioned in his lecture
    - **Simple** examples for Pipe / IO redirection operators
    - Summarize some **common** linux shell commands (with some easy examples, e.g. for sed, tr, ...)
    - Regular Expression patterns
    - (Most)Basic Syntax for shell scripting
    - Emacs reference card
    - (Most)Basic Syntax for Lisp
    - (Most)Basic Syntax for python
    - (Most)Basic Syntax for HTML
    - (Most)Basic Syntax for Javascript
    - (Most)Basic Syntax for C and Makefile
    - Git
      - Basic commands
      - How Gits organize objects
    - Client/Server related techs
      - TCP/UDP (feature, advantage/disadvantage)
      - Node.js, React, ...
  - Basically basic concepts and ideas are more important than concrete syntax, especially for paper-based exams
  - Don't bring too many different kinds of resources; don't bring heavy books (it will distract you; make sure you can locate what you need in a fast manner)

1.3. Examples

1. Logistic
2. Strategy
3. **Examples**

F20: CS 97 Final Fall 2020 (Open Computer)
S20: CS 35L Final Spring 2020 (Open Computer)
-k: Question No. k

https://docs.google.com/document/d/16m_zqE5aRoxVHyAcMMbYooCPtkyZ3y3ISlNzvbNM1SA/edit?usp=sharing
(There could be some errors/typos in this doc, so please leave a comment here if you find any mistakes.
Also please feel free to leave comments if you have any questions about anything, and any followed comments/discussions are welcomed.)

1.3.1. Files, Editing and Shells

Files, editing, and shells

- Multiuser and multiprocess operating systems
- CLI basics (e.g., Bash)
- Unix file system organization
- Basic shell commands: ls; pwd, cd, mkdir, rmdir; echo, cat; cp, mv, ln, rm; chmod, kill, ps, grep, find
- Documentation and man pages
- Emacs basics: introduction, online tutorial (C-h t), help (C-h ?), basic editing, directory editing, running shell commands, building programs, Emacs Lisp.

Examples:

- File System & Basic commands: F20-1, F20-3, S20-1 (6pts)
- Emacs:

1.3.2. Commands and Basic Scripting

Commands and basic scripting

- The Unix shell as a scripting language
- Pattern matching, wildcards and regular expressions
- More advanced commands (e.g., grep, find) (sed, tr ...)
- Pipelines and redirection
- Scripting for interactive applications, with Emacs Lisp as a case study

Examples:

- Shell scripting: F20-2
- Regular expressions: S20-2 (8pts), S20-5a,5b,5c (6pts)
- Advanced commands: S20-3 (4pts),
- Lisp: M-3 (10pts)

1.3.3. Scripting and Construction

Scripting and construction

- Basics of Python
- Java as a compromise between interpreted and compiled languages
- Building from source
  - Dependencies (e.g., make, Apache Ant)
  - Makefile basics
  - Programming with dependencies (e.g., automake, autoconf)

Examples:

- Python: S20-5d (4pts)
- Interpreted/Compiled Language:

1.3.4. Client-server Apps and User Interfaces

Client-server apps and user interfaces

- Scriptable user interfaces via JavaScript
- HTML and web protocols
- Case study: Node.js and React.

Examples:

- User Interface:
- TCP/UDP:
- HTML & Web Applications:
- Node.js and React: F20-8, F20-11

1.3.5. Package Management

Package management

- Dependencies and packages
- Installing a package
- Creating a package
- Package management case study: (e.g., Python packaging)

Examples:

- NULL

1.3.6. Change Management

Change management

- Creating and using patches (e.g., diff and patch)
- File-based version control, e.g., RCS
- Package-based version control, e.g., CVS
- Decentralized version control, with Git as a case study
    - Commits
    - Branching and merging
    - Rebasing
    - Branching workflows
    - Searching
    - Advanced merging and conflict resolution
    - Remote branches
    - Submodules
    - Git internals

Examples:

- Diff and patches: S20-4 (4pts),
- Git: F20-4, F20-5, F20-9, F20-10, S20-10 (7pts), S20-11 (3pts), S20-13 (4pts)

1.3.7. Low-level Construction and Debugging

Low-level construction and debugging

- Introduction to C
- The C compilation and linking process
- Debuggers and debugging tools, e.g., GDB, Valgrind, strace.
- C and system programming
- Library calls vs. system calls
- Ways in which a program can go wrong at the low level (e.g., buffer overruns)
- Defenses against low-level misbehaviors

Examples:

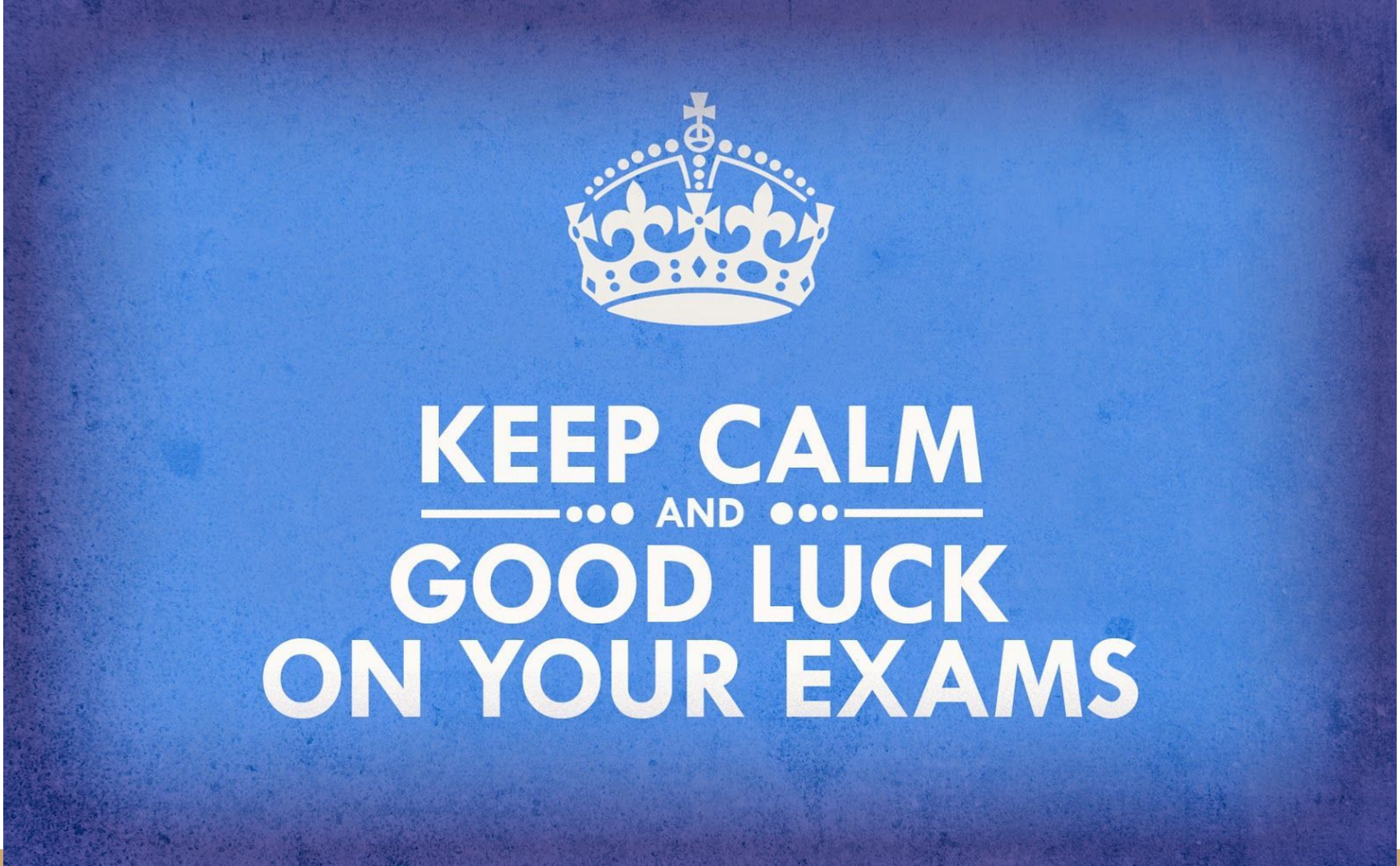
- C: F20-6, S20-6 (10pts), S20-7 (4pts)
- Debugging tools: F20-7

1.3.8. Open Questions

- Combine different kinds together, what we have learned + open thinkings
- Generally takes longer time (more pts)
- Tricks:
    - Enumerate your knowledge about each mentioned tech
        - List in items, make it easier for the grader to catch your points
    - Prepare before the exam: rethink about the advantages/disadvantages and the use cases of each tech we mentioned in this quarter

Examples:

- F20-12, S20-8 (6pts)


KEEP CALM
AND
GOOD LUCK
ON YOUR EXAMS