Department of Computer Science

Study Guide for Graduate Comprehensive Exams

Spring 2015

A PHOTO ID IS REQUIRED FOR ENTRANCE TO THE EXAM.

YOU WILL ONLY BE ALLOWED TO

TAKE WRITING INSTRUMENTS TO YOUR SEAT DURING THE EXAM.

YOU WILL BE REQUIRED TO LEAVE ALL BACKPACKS, NOTES, AND

ELECTRONIC DEVICES AT THE BACK OR FRONT OF THE EXAM ROOM.

The content exam will be given 14:00-17:00 on Friday, Mar. 13 in DERR 113. The software engineering exam will be given 14:00-17:00, Mar. 13 in DERR 113. The programming exam will be given 15:30 -- 17:00 on Friday, Mar. 27 in DERR 113. The communication exam will be given 14:00 -- 15:00 on Friday, Mar. 27 in DERR 113.

The programming exam covers the "content/skill level of CS3358."

The communication exam covers the ability to write clear technical English on Computer Science topics.

The content exams cover the core courses.

So be sure to check that you have the right set of topics.

You will be examined over the courses listed on your exam registration.

Software Engineering Study Guide

--------------------------------

CS 5391

- software life cycle models

- software measurement

CS 5392

- verification by proof

- model checking

- satisfiability based verification

- data flow analysis

- symbolic execution

- model checking

- satisfiability based verification

CS 5393

- software inspections

- code coverage measures for testing

- coverage criteria for testing

- syntax based testing

- regression testing

Computer Science Study Guide

----------------------------

1. CS5306 Advanced Operating Systems

a. mutual exclusion solutions in distributed os

b. deadlocks solutions in distributed os

c. clock synchronization in distributed os

(physical clock, logical clock)

d. consistency models of distributed shared memory

2. CS5310 Network and Communication Systems

a. Significances of the Fourier series with

regard to data transmissions

b. The seven layers of the OSI reference model

c. Definitions and comparisions of CSMA/CD, Token

Bus, and Token Ring LAN protocols

d. Basic concepts, structures, and programming of

BSD Socket programming

3. CS5318 Design of Programming Languages

a. role of context-free grammars in programming

language design

b. role of abstraction in programming language

design

c. computational paradigms

d. the use of attribute grammars to decorate

parse trees

4. CS5329 Algorithm Design and Analysis

a. Recurrences including recursion trees,

substitution and master method, with examples.

b. Algorithm design strategies,including

characteristics and examples.

c. Insertion, Selection, Quick, Shell, Bucket,

Radix sorting. Know the definitions, properties

running times for the best and worst running

cases in terms of big O notation. Be able to

apply to specific inputs.

d. Hash functions definitions, collision

resolution schemes, linear, quadratic probing,

double hashing. Be able to provide and work

with specific examples

e. AVL, Splay trees, Heap definitions,

properties, examples. Be able to add and

delete nodes from each of those entities.

5. CS5332 Data Base Theory and Design

a. elementary concepts of DBMS

b. data analysis for table formation

c. normalization of tables(relations):

1NF, 2NF, 3NF, and BCNF

d. E/R modeling

e. SQL, database tools and database

implementation for real world problems

f. relational algebra

6. CS5338 Formal Languages

a. Mathematical foundations including countable

and uncountable infinity and diagonalization

b. Definition, construction and properties of

finite state machines, pushdown automata and

Turing machines

c. Halting behavior of deterministic and non-

deterministic machines

d. Language hierarchy and closure properties of

regular and context free languages

e. Complexity classes and NP-completeness proof

techniques

f. Church-Turing thesis, decidability and semi-

decidability

7. CS5346 Advanced Artificial Intelligence

a. Conventional and Informed Search Methods

b. Minimax Search Procedures including Alpha-Beta

Pruning

c. Representation of Knowledge and Inference in

Propositional and First-Order Logic

d. Use of Forward and Backward Chaining in Expert

Systems

e. Building knowledge bases and reasoning for

applications like the testing of the design of

electronic circuits

f. Rule generation through machine learning

methods

g. Intelligent planning

8. CS 5351 Parallel Processing

a. Performance metrics (speedup, efficiency,

overhead, scalability, complexity)

b. Parallelization approaches (data parallelism,

task parallelism, divide and conquer,

partitioning, agglomeration, scans)

c. Shared-memory machines and programming

(consistency models, cache coherence, false

sharing)

d. Message-passing machines and programming

(network topologies, bandwidth, routing)

e. Synchronization primitives (barriers, mutexes,

locks, read/write locks, semaphores, condition

variables)

f. Communication primitives (send, receive,

broadcast, reduce, scatter, gather, point to

point vs. collective, blocking vs. non-

blocking)

g. Parallelism issues (data races, livelock,

deadlock, indeterminacy, load balance,

termination detection, reentrancy)

h. Parallel programming (OpenMP, MPI)

9. CS5391 Survey of Software Engineering

a. Software Development Lifecycle Models

b. Capability Maturity Model

c. Design Methods

d. Software Testing Strategies

