|  |
| --- |
| **CS 164 Introduction to Computer Science Fall 2013**  **Study Guide for Final Exam Monday, December 9, 2013 10:30 AM - 12:30 PM Nesbitt 111** |

**Reading:**

From the Reed Text:

* Chapter 1 (Computer Basics)
  + Basic Functions of Computersp0-
  + Basic Components of a Computer, Computer System
* Chapter 2 (HTML and Web Pages)
  + HTML tags (see table 4.1) and attributes
  + Anchors
  + Images
  + Lists
  + Tables
  + You***do not*** need to know about frames
  + Absolute vs. Relative URL's
  + CSS -- default font-family and color only
  + XML
  + You***do not*** need to know about XSL
* Chapter 3 (Internet and the World Wide Web)
  + Network, Router
  + Internet Protocol (IP), dotted quad, Transmission Control Protocol
  + Universal Resource Locator, Hypertext Markup Language
* Chapter 4: JavaScript and Dynamic Web Pages
  + JavaScript objects (intrinsic objects, using objects to access form elements)
* Chapter 5: JavaScript Numbers and Expressions
  + Mathematical formulas and calculations in JavaScript
* Chapter 6 (History of Computing)
  + Analytical Engine, Babbage, Ada
  + Jacquard Loom and its Role in Computing
  + Antanasoff and Zuse
  + Mauchley, Eckert and ENIAC
  + First through Fourth Generation computers
  + Moore's Law
* Chapter 7: Event-Driven Pages
  + HTML forms (creation and use)
* Chapter 9: Abstraction and User-Defined Functions
  + JavaScript functions (creation and use)
* Chapter 10 (Computer Science as a Discipline)
  + Core areas of Computer Science
* Chapter 11: Conditional Execution
  + if statements, if/else statements, if/else if/else... statements
  + loops (while statements)
* Chapter 12: Data Representation
  + Binary and Hexadecimal
  + Floating-Point Representation
* Chapter 14: Inside the Computer - The von Neumann Architecture (*nothing from this chapter*)
* Chapter 15: JavaScript Strings

From the Brookshear Text:

* Chapter 6, Sections 6.1-6.2 -- Programming Languages
* Chapter 7, Sections 7.1-7.2 -- Software Engineering
  + Software Life Cycle
* Chapter 9, Sections 9.1-9.2 -- Databases
* Chapter 10, Sections 10.1-11.3 -- Computer Graphics
* Chapter 11, Sections 11.1-11.3 -- Artificial Intelligence

From the Decker and Hirshfield Text:

* Module 6:
  + Binary and Hexadecimal
  + PIPPIN and Rosetta

**Topics Covered:**

* Core Areas of Computer Science
  + **Algorithms and Data Structures**

This area deals with specific classes of problems and their efficient solutions. The performance characteristics of algorithms and the organization of data relative to different access requirements are major components.

* + **Architecture**

Methods of organizing efficient, reliable computing systems provide a central focus of this area. It includes implementation of processors, memory, communications, and software interfaces, as well as the design and control of large computational systems that are reliable.

* + **Artificial Intelligence and Robotics**

The basic models of behavior and the building of (virtual or actual) machines to simulate animal and human behavior are included here. Inference, deduction, pattern recognition, and knowledge representation are major components.

* + **Database and Information Retrieval**

The area is concerned with the organization of information and algorithms for the efficient access and update of stored information. The modeling of data relationships, security and protection of information in a shared environment, and the characteristics of external storage devices are included in this area.

* + **Human-Computer Communication**

The efficient transfer of information between humans and machines is the central focus of this area. Graphics, human factors that affect efficient interaction, and the organization and display of information for effective utilization by humans are included.

* + **Numerical and Symbolic Computation**

General methods for efficiently and accurately using computers to solve equations from mathematical models are central to this area. The effectiveness and efficiency of various approaches to the solution of equations, and the development of high-quality mathematical software packages are important components.

* + **Operating Systems**

This area deals with control mechanisms that allow multiple resources to be efficiently coordinated during the execution of programs. Included are appropriate services of user requests, effective strategies for resource control, and effective organization to support distributed computation.

* + **Programming Languages**

The fundamental questions addressed by this area involve notations for defining virtual machines that execute algorithms, the efficient translation from high-level languages to machine codes, and the various extension mechanisms that can be provided in programming languages.

* + **Software Methodology and Engineering**

The major focus of this area is the specification, design, and production of large software systems. Principles of programming and software development, verification and validation of software, and the specification and production of software systems that are safe, secure, reliable, and dependable are of special interest.

* What *Is* and*Is Not*the Purview of Computer Science
  + can be programmable for different tasks
  + can store and retrieve information
  + is capable of performing millions/billions of instructions per second (without error)
  + **Accept and Store Information**  
    (Input, Storage)

INPUT: The process by which *the user* communicates *information* to *the machine*.

* + **Perform Mathematical Calculations**  
    (+, -, \*, /)
  + **Compare Values of Numbers**   
    (Logic - Are two values equal? If not, which is larger?)
  + **Retrieve and Display Information**  
    (Retrieval, Output)

OUTPUT: The process by which *the machine* communicates *information* back to *the user*.

* Detailed Study in Specific Core Areas
  + Databases
    - Attributes of a database:
      * ***Transactional Processing:*** The ability to add or remove data from the database, or modify existing data. For example, each year Drexel must add new students to its student database, and update existing student records.**Buzzword:** *OLTP: online transaction process*
      * ***Query:*** The ability to search a database for information. Usually only certain aspects of the data, called *index fields*, are used in the search. For example, retrieve a list of all Drexel students for whom the permanent address is in Florida.
      * ***Batch Processing:*** The ability to perform a repeated task for numerous elements of the database without involving human monitoring or interaction for each transaction. For example, Drexel prepares a bill, or grades, for each student every so often. Rather than have the registrar pull up each student's record individually and process the transaction, a "batch request" is created that contains specific instructions to be performed for each student processed. The students' ID's may be listed individually, or may be determined by a query.
      * ***Report Writing:*** The ability to create customized ways of displaying the data. The Drexel student database can create transcripts, class lists, list by major, list by class rank (freshman, senior, etc.), or any number of reports.
      * ***Data Analysis:*** Many databases provide tools to aid in the analysis of the data. Tools include the ability to create graphs, or perform advanced mathematical or statistical analysis (regression, line of best fit, etc.).
    - Types of databases:
      * ***Library or Flat-File Database:*** Think of a library: a huge pile of books (data) just sitting their waiting to be used. Although the books may be *organized* (sorted), there is no *relation* between the books. A flat file database contains all the information, as well as an index that is used to locate various data items. The index serves the same role as the library's card catalogue, hence the name "flat file." (Think of an index card).
      * ***Relational Database (RDBMS):*** In a relational database, several different types of data are stored in several different files (or tables), but the data items are *linked* through common *fields* or *attributes*. For example, Drexel maintains a database of students, but it also maintains a database of Professors. Although these two groups of data are usually kept separate, the can be related by classes taken/taught or through majors/departments.

Relations are further categorized as *one-to-one* (e.g. one office for one professor), *many-to-one* (e.g. each classroom is used by several classes), or *many-to-many* (e.g. many students are linked to many classes). In the example below, all relations are many-to-one.

* + - * ***Hierarchical/Network Database:*** In a hierarchical database, the data is organized in levels. For example, the files on your computer are arranged in a hierarchy (think pyramid). This type of organization is characterized by the many-to-one relationship, but unlike the relational database, this relation often has a definite "direction".
      * ***Object Databases:*** The newest member of the database family, object databases are based upon the *object oriented paradigm*. These databases mesh cleanly with object oriented programming languages, and allow for more *heterogeneous* collection of data. For example, an object database of students would not only allow for the inclusion of student photos as data, but those photos could also be *searchable* by the software.
    - SQL = Structured Query Language, used to manipulate databases
    - BLOB = Binary Large Object
  + Human-Computer Interaction
    - “Human-Computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.”
    - Types of majors included in HCI:
      * ***Computer Science*** - designing applications and user interfaces.
      * ***Psychology*** - applying theories of human behavior
      * ***Sociology and Anthropology*** - studying the interaction between technology, work and organization.
      * ***Engineering*** - designing products
    - Examples of HCI:
      * ***Interface Design***: the graphical user interface, windows, frames, application interfaces
      * ***Interface Hardware***: the mouse, touch screens, voice recognition, devices for the disabled
      * ***Ergonomics***: repetitive motion disorder, lower back pain, eye strain, fatigue, lighting issues, special needs (left handedness, color blindness, interfaces for children, etc.)
      * ***Embedded Computers***: microwaves, cell phones, cars, entertainment systems
      * ***Human Learning***: educational software, on-line help, information kiosks
      * ***Communication***: e-mail, instant messaging, group interfaces
      * ***Adaptation***: users adapting to machines, machines that adapt to users
      * ***Computer Timing***: does the computer respond too slow? or too fast?
      * ***Social Implications***: consequences of improved productivity demands, computers replacing humans in the work force, cool colors for my PC
  + Scientific Computation
    - NOT SURE WHAT THIS O CRAP.
  + Computer Hardware
  + Artificial Intelligence
    - That is, a machine is intelligent if it performs actions that are considered intelligent when performed by human beings.

Mundane Tasks: (for everyday use....)

* Robotics - automation, motion
* Perception - vision, speech

Formal Tasks: (following a fixed set of rules....)

* Games - "Would you like play a game of chess?"
* Mathematics - geometry, logic

Expert Tasks: (advanced decision making skills....)

* Engineering - design, manufacture
* Medical Diagnosis
* Financial Analysis - loan approval
  + Computer Graphics
    - A graphical image is made of up *picture elements* (or *pixels*), each of which has a RGB value. Dozens of pixels are arranged in a grid to make an image, such as a character that appears on the screen.
  + Public-Key Cryptography
    - The science of developing *encryption* and *decryption* schemes so that a sender and intended receiver are able to transmit and receive messages that others are unable to read.
    - a *message* **M**
    - the readable form of **M** is said to be in *plaintext*
    - the message is *encoded* with an *encryption scheme* **E**
    - the encoded message **C = E(M)** is said to be in *ciphertext*
    - the encoded message can be *decoded* with a *decryption scheme* **D**
    - **M = D(C) = D( E(M) )**
* History of Computing Devices and Languages
  + **1950's** : COBOL, FORTRAN
  + LGOL, Basic
  + **1970's & 1980's** : Pascal, C
  + **Object Oriented "Era"**: C++, Java
* Computing Devices and Capabilities (input, output, storage, retrieval, arithmetic, logic)
* Concepts of Programming Languages (stored program, delimiters, identifiiers, syntax, comments)
* Client-Server interaction
* Database Nomenclature (types of databases, database operations)
* HTML (HyperText Markup Language)
  + anatomy of a page
  + tags, attributes
  + comments
  + formatting elements
  + links (absolute vs. relative)
  + colors, RGB system
  + lists
  + forms
* Cascading Style Sheets (CSS - only to the extent covered in course notes)
* XML, XHTML (XSL not covered)
* JavaScript
  + what is it? (what it's not i.e. Java)
  + Embedding it in HTML
  + script
  + comments in JavaScript vs. comments in HTML
  + Intrinsic JavaScript Objects
  + document object
    - properties: (document.referrer, document.lastModified, ...)
    - methods: (document.write(), ...)
  + arithmetic expressions, assignment statements
  + string expressions
  + conditional expressions, if statements, if/else statements
  + functions, event handlers
  + rollovers
  + communication between forms and functions
  + random numbers
  + loops
  + arrays
* HTML forms - creation
* JavaScript functions (making, using, parameter passing)
* Number Representations and Systems (Binary, Decimal, Hexadecimal)
  + Binary Representation of Integers (including two's complement)
  + Binary Addition, Binary Subtraction
  + Binary Representation of Characters (ASCII)
  + Conversion between bases
  + Binary fractions
  + Colors in HTML and Hexadecimal numbers
* Computer Representation of Floating Point Numbers (exponent, mantissa, etc.)
  + Errors arising from representation of integers and floating point numbers
* Supporting Mathematics
  + Modular arithmetic
  + Euler's Totient function
* User Interface Design Principals (Golden Rules for Interface Design)
  + Transparency, Forgiveness, Visual, Intuitive
  + Highlights from "The Macintosh User Interface Guide"
* Graphics Formats
  + JPG, GIF, PNG
  + lossy vs. exact representations
  + bitmap vs. vector representations
* The Object Oriented Paradigm
  + Origins
  + Advantages and Disadvantages
  + Attributes and Methods
  + Object Oriented nomenclature in Computer Science (i.e. "who.what")

**From the Labs:**

* JavaScript!!!
* HTML forms!
* Binary representation and round-off errors.
* Regular Expressions.

**Types of Problems to Expect:**

1. You can expect to see short answer, fill in the blanks, TRUE/FALSE, and multiple choice questions
2. You can expect to see questions involving key terms and nomenclature from all areas of study
3. You can expect to see questions similar to those in the text on *number systems*, *program translation*, *hardware*,and *artificial intelligence/expert systems*.
4. You can expect to see problems dealing with various aspects of your homework assignments and labs.
5. You can expect to see problems of the form "What appears in the browser window when the following HTML code appears on a web page" and "What HTML code should appear on a web page in order to produce the following output in the browser window"
6. You can expect to see problems of the same form for JavaScript
7. You can expect to see problems that involve communication between HTML forms and JavaScript scripts.