

2010-2011 Assessment for Master's Degree Program  
Fall 2010 - Spring 2011  
Computer Science Dept.  
Texas A&M University - Commerce

Program Objective #1 (PO1): Students will be able to demonstrate a broad knowledge of Computer Science which includes data structures, operating systems, computer programming skills, computer organization, algorithm design, and automata theory.

Program Objective #2 (PO2): Students will gain a substantial knowledge of one of the following Computer Science specialties: Database, Networking, Artificial Intelligence, Information Security, Computer Engineering.

Program Objective #3 (PO3): Students will demonstrate the ability to recognize, design and implement efficient software solutions to problems.

Program Objective #4 (PO4): Students will demonstrate knowledge and understanding of professional ethics and responsible behavior.

Program Objective #5 (PO5): Students will demonstrate the ability to communicate effectively and to work as a team.

Program Objective #6 (PO6): Students will become successful professionals able to gain Employment and/or to be accepted into a Computer Science Ph.D. program.

**Program Objective#1 (PO1): Students will be able to demonstrate a broad knowledge of Computer Science which includes data structures, operating systems, computer programming skills, computer organization, algorithm design, and automata theory.**

Assessment will be measured through testing the following course objectives:  
The first percentile is Fall 2010 and the second percentile is Spring 2011.

#### CSCI 515 Fundamental of Programming

(CO515.1): To understand the internal representation of the various data types.

(CO515.2): To examine the internal representation of two and three dimension arrays in C/C++.

(CO515.3): To understand dynamic memory allocation, parameter passing, the use of pointers.

#### CSCI 516 Fundamental Concepts of Computer and Machine Organization

(CO516.1) Numbering systems and conversions, Boolean functions.

(CO516.2) Intro to Computer Organization: design logic; digital diagrams, and basic circuits and gates, and the link between Boolean functions, circuits, processor and Micro code.

(CO516.3) Concepts of Machine Instructions, Assembly and linking, common interrupts.

(CO516.4): Concepts of Jumps, flags, subroutines, procedures, stacks, stack parameters and frames..

(CO516.5): Arrays, addressing modes and Floating Point memory management, indirect addressing.

(CO516.6): Advanced procedures, local variables, stack parameters, strings.

#### CSCI 520 Information Structure and Algorithm Analysis.

(CO520.1): To understand the concept of sparse matrices, stack and queues.

(CO520.2): To examine the differences between linear and linked representation of stacks, queues, and ordered data.

(CO520.3): To understand and implement tree structures and to compare various sorting algorithms.

CSCI528 Object Oriented Methods  
(CO528.1): Software Engineering Basics.

#### CSCI 530 Operating Systems

(CO530.1) Understand the concepts, structures, and mechanisms of operating systems.

(CO530.2) Understand memory management, virtual memory, swapping, paging algorithms, segmentation, and clock paging policies.

(CO530.5) Understand concurrent processes and associated deadlock prevention, avoidance, detection, recovery methods, and the use of semaphores.

#### CSCI 532 Algorithm Design

(CO532.1): To teach students how to analyze algorithms in order to determine their calculation complexity in the terms of Big Oh, Big theta and Omega. Recursions.

(CO532.2): To teach sorting algorithms (such as mergesort and quicksort) and their applications.

(CO532.3): Probabilistic Analysis and Randomized algorithms for sample problems from the following list (not limited to, and not necessarily including all): CS- Hiring, Longest Streaks, Bins and Balls problem, the Birthday paradox, and randomized quicksort.

(CO532.4): Binary search trees and optimal binary search trees, and their applications.

(CO532.5): Dynamic programming algorithms for problems such as line scheduling, matrix chain multiplication, longest common subsequence, and their practical applications.

(CO532.6): Greedy algorithms for problems such as the activity selection problem and its application to resource planning.

#### CSCI 540 Computer Architecture

(CO540.1): General purpose machines from different views. Instruction sets and classification of computers.

(CO540.2): Cost and performance of a computer: evaluation metrics, Amdahl's law, principle of locality, and benchmarks.

(CO540.3): Cache and memory organization: cache mapping and replace strategies, virtual memory and cache coherence.

(CO540.4): Pipelining: performance issues, and pipelining hazards.

(CO540.5): I/O system: hard drive, RAID technology, I/O performance and benchmarks.

#### CSCI 549 Automata Theory

(CO549.1): Understand the concept of languages and recursive definitions

(CO549.8): Construct a context free grammar to define a context free language

(CO549.11): Construct a push down automata for a language

(CO549.12): Design and construct a Turing machine for any language

(CO549.13): Design and construct a LR(1) parser for SmallG language

**Program Objective#2 (PO2): Students will gain a substantial knowledge of one of the following Computer Science specialties: Database, Networking, Artificial Intelligence, Information Security, Computer Engineering.**

Assessment will be measured through testing the following course objectives.

The first percentile is Fall 2010 and the second percentile is Spring 2011.

### CSCI 525 Introduction to Local Area Networking

(CO525.1): To define and understand basic Data Communications, networking topologies, the OSI Model and the IEEE 802 standards.

### CSCI 526 Databases Systems

(CO526.1): Obtain current status of the state-of-the-art database design methodology in industry and academics.

(CO526.5): Write SQL programs for effective data definition and manipulation.

(CO526.6): Develop ER diagrams for logical design of database systems.

(CO526.7): Implement a small scale database development project using commercially available DBMS tools.

### CSCI 534 Networking II Routers and Switches

(CO534.1): Using subnets and routing protocols, design and configure a router network.

(CO534.2): Design and configure a switched network and VLANs.

(CO534.3): Understand the concepts of an Access Control List and learn how to configure a router for ACLs.

### CSCI 538 Artificial Intelligence

(CO538.1): To learn about general concepts in the field of artificial intelligence.

(CO538.2): To learn about the current fields of research in artificial intelligence.

(CO538.3): To work on an on-going class project to create a computer program that learns from its users.

### CSCI 539 Expert Systems

(CO539.1): To learn about the general concepts and deployment of expert systems.

(CO539.2): To create an expert systems project using a pre-developed software tool (environment) or in any language of your choice.

### CSCI 553 Networking III – Unix Based Networks

(CO553.5): Become familiar with sockets, including programming both connection-oriented TCP and connectionless UDP sockets.

(CO553.6): Be able to create simple TCP Client/Server applications using sockets in a High-level language/toolbox such as Java, Perl, Python or C++.

### CSCI 563 Fundamentals of Information Security & Assurance

(CO 563.1): State the basic concepts in information security, including security policies, security models, and various security mechanisms.

(CO563.2): Understand the issues of network communications such as service, confidentiality, authentication, reliability, access control, and availability.

(CO563.3): State threats and sources of attacks in network security.

(CO563.4): Explain how to use cryptography to protect information and how to choose an appropriate encryption method.

(CO563.5): State main strategies to secure Windows and Linux computers.

(CO563.6): Understand limitation of the current security technology and able to choose proper security mechanisms.

### CSCI 581 Computer and Network Security

(CO581.1): Students will be able to describe and discuss information security and network security basics.

(CO581.2): Students will be able to describe and discuss cryptography basics.

(CO581.3): Students will be able to describe and discuss authentication in network applications.

(CO581.4): Students will be able to describe and discuss electronic mail security.

(CO581.5): Students will be able to describe and discuss IP security.

(CO581.6): Students will be able to describe and discuss network security applications that implement the above capabilities.

**Objective#3 (PO3): Students will demonstrate the ability to recognize, design and implement efficient software solutions to problems.**

Assessment will be measured through testing the following course objectives.

The first percentile is Fall 2010 and the second percentile is Spring 2011.

**CSCI 520 Information Structure and Algorithm Analysis.**

(CO520.1): To understand the concept of sparse matrices, stack and queues.

(CO520.2): To examine the differences between linear and linked representation of stacks, queues, and ordered data.

(CO520.3): To understand and implement tree structures and to compare various sorting algorithms.

**CSCI527 Advanced Databases and Data Mining**

(CO527.1): Understand current status of the state-of-the-art data mining methodology in industry and academics.

(CO527.3): Learn and use effective tools for web navigation and program integration management.

(CO527.5): Construct programs for capturing association rules.

(CO527.6): Write programs for trend analysis using statistical data mining techniques.

**CSCI 528 Object Oriented Methods**

(CO528.2): Classes basics/advanced.

(CO528.3): Overloading.

(CO528.4): Polymorphism/Virtual function.

(CO528.5): Template, Exception.

(CO528.6): UML.

**CSCI 532 Algorithm Design**

(CO532.1): To teach students how to analyze algorithms in order to determine their calculation complexity in the terms of Big Oh, Big theta and Omega. Recursions.

(CO532.2): To teach sorting algorithms (such as mergesort and quicksort) and their applications.

(CO532.3): Probabilistic Analysis and Randomized algorithms for sample problems from the following list (not limited to, and not necessarily including all): CS- Hiring, Longest Streaks, Bins and Balls problem, the Birthday paradox, and randomized quicksort.

(CO532.4): Binary search trees and optimal binary search trees, and their applications.

(CO532.5): Dynamic programming algorithms for problems such as line scheduling, matrix chain multiplication, longest common subsequence, and their practical applications.

(CO532.6): Greedy algorithms for problems such as the activity selection problem and its application to resource planning.

**CSCI 581 Computer and Network Security**

(CO581.2): Students will be able to describe and discuss cryptography basics.

**Objective #4 (PO4): Students will demonstrate knowledge and understanding of professional ethics and responsible behavior.**

Assessment will be measured through testing the following course objectives.  
The first percentile is Fall 2010 and the second percentile is Spring 2011.

CSCI 563 Fundamentals of Information Security & Assurance

(CO 563.1): State the basic concepts in information security, including security policies, security models, and our security mechanisms.

(CO563.2): Understand the issues of network communications such as service, confidentiality, authentication, reliability, access control, and availability.

(CO563.3): State threats and sources of attacks in network security.

CSCI 581 Computer and Network Security

(CO581.1): Students will be able to describe and discuss information security and network security basics.

**Objective #5 (PO5): Students will demonstrate the ability to communicate effectively and to work as a team.**

Assessment will be measured through testing the following course objectives.  
The first percentile is Fall 2010 and the second percentile is Spring 2011.

CSCI 526 Database Systems

(CO526.2): Master the technique for team play and teamwork for small scale database projects through brain storming and joint requirement planning.

(CO526.10): Be able to demo and present the initial, intermediate, and final delivery of the database design project.

CSCI 527 Intelligent Database Systems

(CO527.2): Obtain the technique for team play and teamwork for large intelligent database projects through brain storming and joint requirement planning.

CSCI 528 Object Oriented Methods

(CO528.7): Integration Project.

**Objective #6 (PO6): Students will become successful professionals able to gain employment and/or to be accepted into a Computer Science Ph.D. program.**

Assessed by on-going follow-up surveys and letters of feedback from students.

## APPENDIX A

### **Assessment Plan Details and Examples Graduate and Undergraduate Degree Programs Department of Computer Science & Information Systems Texas A&M University-Commerce**

#### I. Program Embedded Assessment

The following is a description of the assessment program for the Computer Science Master's Degree Program, the Computer Science Bachelor of Science Degree, and the Computer Information Systems Bachelor of Science Degree. The assessment program contains: a) an overall comprehensive testing component; b) a course-embedded assessment component; c) a non-quantitative data gathering component.

#### A. Overall Comprehensive Testing Component

Currently, the Computer Science Master's program requires the successful completion of a comprehensive exam as a prerequisite for the Master's Degree. This exam is given three times a year (in the Fall, Spring, and Summer semesters). The exam is developed and administered by a committee of graduate faculty. A set of the most important course objectives are selected (see Course Embedded Assessment, Section B) for examination. Testing for these objectives are translated into a set of questions and administered. The results are translated into a percentile of those passing and failing by objective, and a list of those students passing or failing the exam. The criteria for passing is determined by the committee. Such criteria is based on the overall performance of students taking the exam. The report of the percentile of those passing or failing by objective will be used as feedback in re-evaluating techniques used in teaching these objectives.

Starting in the Spring Semester, 2002, a comprehensive exam will be also given to students graduating from the Computer Science and the Computer Information Systems Undergraduate Programs. The exam will be constructed by a committee of undergraduate faculty. It shall be a one or two hours exam and shall be given in the final days of the senior course, CSCI 440. Unlike the graduate comprehensive exam, the undergraduate comprehensive exam will not be required for graduation. It will be used only for program evaluation. The comprehensive exam will be developed and administered by a committee of undergraduate faculty. A set of the most important course objectives will be selected (see Course Embedded Assessment, Section B) for examination. Testing for these objectives will be translated into a set of questions and administered. The results will be translated into a percentile of those passing and failing by objective and will be used as feedback in re-evaluating techniques used in teaching these objectives.

Instead of using an in-house comprehensive exam, the option is reserved for a utilizing a nationally administrated exam that would nationally rank participants.

## B. Course Embedded Assessment Component

The purpose of embedded assessment is to measure the degree of success in which each course objective has been met. Starting in the Spring 2003 semester, the comprehensive exam for both graduate and undergraduate students will contain embedded assessment of program objectives.

1. The most important course objectives (see Section II below) will be selected from objectives listed in core courses for both the graduate and undergraduate programs. These important objectives will comprise the overall program objectives. The selection will be made by the Graduate Curriculum Committee and Undergraduate Curriculum Committee.

2. These overall program objectives will be embedded in one or more questions given in the Graduate Comprehensive Exam and in the Undergraduate Comprehensive Exam. A passing grade on the Graduate Comprehensive Exam, as determined by the Graduate Curriculum Committee, is required for student graduation. The Undergraduate Comprehensive is not a requirement for graduation. The assessment for each objective is a percentile of the total maximum score in a similar manner that course objectives are measured. For example, if Program Objective #1 is measured by two 10 point questions and a total of 10 students take the exam, then a total maximum score for Program Objective #1 will be 200 points. If a total of 175 points is scored for Objective #1, then the overall assessment for Objective #1 is 87.5%. See Appendix A for a more detailed example.

3. Program Objectives will be divided into two categories: Category (A) 75% or greater; and Category (B) less than 75%. Objectives in Category A will be considered successful while objectives in Category B will be considered unsuccessful. The report submitted by the instructor shall include steps being taken to better emphasize and teach objectives falling into Category B.

## C. Non-quantitative Data Gathering Component

A non-quantitative data gathering component shall also be part of the program assessment. This component consists of the following:

1. Exit interviews with graduating students.
2. Compilation of job positions and salaries offered to graduating students.
3. Interviews with employers of graduates.
4. Interviews with potential employers of graduates.

The purpose of these interviews are to keep informed of changes in employment patterns as we continually adjust our computer science and information systems curriculum to meet the needs of the workplace.

## II. Course Embedded Assessment Plan

The purpose of embedded assessment is to measure the degree of success in which each course objective has been met. Starting in the Spring 2003 semester, selected courses taught within the Department of Computer Science & Information Systems will be designed to contain embedded assessment of course objectives on an experimental basis. Please note that while embedded course objective are similar to embedded **program objectives**, the embedded program objectives are designed to meet the requirements of accreditation. Assessment of **course objectives** are designed to spot potential problem areas within a particular course. Thus, embedded course objective assessment may be more flexible than assessment of program objectives simply because of the nature of differing course material.

Procedure for implementing this course embedded assessment plan are as follows:

(Note: this procedure plan is evaluated on a continuous basis and is subject to change.)

1. Course objectives are designated for each course taught in the department. The number of objectives should be more than 10 and probably less than 30 objectives. Multiple sections of the same course will designate identical course objectives, as agreed to by those teaching the course. On a yearly basis, all course objectives will be examined and re-evaluated by members of the Undergraduate Curriculum Committee and the Graduate Curriculum Committee. Course objectives should appear in the syllabus for that particular course. A course objective may have sub-objectives, however, only the assessment of course objectives, and not the sub-objectives, will be reported.

2. A quantitative assessment of the success of each objective shall be measured by questions given to students in regular examinations, mid-term examinations, and final examinations. This assessment measurement will exist as a percentile of the total score for each objective. (See example below).

3. Each faculty member teaching a course shall report the result of student performances of each embedded objective. The report shall be given to the department head no later than 3 weeks after the final exam. The report shall list each course objective and the percentile score of those successfully achieving this objective. Sub-objective scores need not be reported. For classes that have different instructors teaching multiple sections, the report shall be given to a designated class coordinator who will, in turn, prepare a consolidated report. The consolidated report along with each individual report shall be turned in the department head.

4. Objectives will be divided into two categories: Category(A) 75% or greater; and Category (B) less than 75%. Objectives in Category A will be considered successful while objectives in Category B will be considered unsuccessful. The report submitted by the instructor shall include steps being taken to better emphasize and teach objectives falling into Category B.

Example:

Suppose a given set of objectives for CSCI 151 are as follows:

1. Construct appropriate comments inside a C++ program.
2. Declare valid identifiers using appropriate data types in a C++ program.
3. Evaluate and construct assignment statements in a C++ program.
4. Input and output data in a C++ program.
5. Evaluate and construct selection structures using C++.
6. Evaluate and construct repetition structures using C++.
7. Construct programs consisting of multiple functions.
8. Understand the concepts of scope and lifetime.
9. Understand how and when to use value and reference parameters with functions.
10. Effectively use one dimensional arrays.

11. Create algorithmic solutions to programming problems and implement them into C++.
12. Perform the steps, without assistance, to enter, run and debug a C++ program using a C++ compiler.

Further consider that objective #4 is tested by 3 questions (10 points each) on the mid-term exam and by 2 questions (10 points each) on the final exam. Thus, a total of 50 points per student will test the achievement of objective #4. If there are 10 students in the class, then there will be a total of 500 points which measure the effectiveness of objective #4. Given the final score per student as follows:

#### Objective #4

Student #1 45 points  
Student #2 40 points  
Student #3 50 points  
Student #4 10 points  
Student #5 0 points  
Student #6 30 points  
Student #7 15 points  
Student #8 25 points  
Student #9 40 points  
Student #10 45 points

-----

300 points total

Percentile success rate = 300 points out of a possible 500 points =  $300/500 = 60\%$

Thus we have an overall success rate of 60% with objective #4.

EXAMPLE: Given the following objectives for CSCI 151

#### CSCI 151 Course Objectives

1. Understand a brief history of computing, overview of programming languages and the compilation process.
2. Create algorithmic solutions to programming problems and implement them into.
3. Declare valid identifiers using appropriate data types in a program.
4. Construct and evaluate assignment statements in a program.
5. Perform data Input and output data in a program with file processing and stream I/O
6. Construct and evaluate selection structures.
7. Construct and evaluate repetition structures.
8. Construct programs consisting of multiple functions.
9. Understand the concepts of scope and lifetime.
10. Understand how and when to use value and reference parameters with functions.
11. Effectively use one dimensional arrays.
12. Construct appropriate comments/documentation to a program.
13. Perform the steps, without assistance, to enter, run and debug a program using an IDE.

Suppose the evaluation for Objective #1 was embedded in two questions in exam #1 (20 points) and in one question in the final exam (10 points). Thus the total number of points used for the assessment of Objective #1 is 30 points. There are 21 students in the class or a total of 630 points for the overall assessment of Objective #1. Further suppose that students achieved a grand total of only 300 points on the two questions in exam #1 and on the one question in the final exam. Thus the overall assessment of Objective #1 is 47.6%. This overall assessment will be noted in a report to the Department Head.

## Overall Assessment of Objectives

Course: CSCI 151.001 Spring 2002

Instructor: Will McWhorter

- \*\*1. Understand a brief history of computing, overview of programming languages and the compilation process.
- 2. Create algorithmic solutions to programming problems and implement them into.
- \*\*3. Declare valid identifiers using appropriate data types in a program.
- \*\*4. Construct and evaluate assignment statements in a program.
- \*\*5. Perform data Input and output data in a program with file processing and stream I/O
- 6. Construct and evaluate selection structures.
- 7. Construct and evaluate repetition structures.
- 8. Construct programs consisting of multiple functions.
- 9. Understand the concepts of scope and lifetime.
- 10. Understand how and when to use value and reference parameters with functions.
- \*\*11. Effectively use one dimensional arrays.
- 12. Construct appropriate comments/documentation to a program.
- 13. Perform the steps, without assistance, to enter, run and debug a program using an IDE.

\*\* denotes unsuccessful objectives (< 75%)

| Objectives | Percentile Achievement |
|------------|------------------------|
| 1)         | 48%                    |
| 2)         | 88%                    |
| 3)         | 20%                    |
| 4)         | 58%                    |
| 5)         | 67%                    |
| 6)         | 98%                    |
| 7)         | 78%                    |
| 8)         | 88%                    |
| 9)         | 90%                    |
| 10)        | 88%                    |
| 11)        | 69%                    |
| 12)        | 99%                    |
| 13)        | 98%                    |

Category A (Successful  $\geq 75\%$ )

2 6 7 8 9 10 12 13

Category B (Unsuccessful < 75%)

1 3 4 5 11