CS 361 – Advanced Data Structures and Algorithms
SYLLABUS

College: **College of Science**
Department: **Department of Computer Science**
Syllabus Title: **CS 361 – Advanced Data Structures and Algorithms**

1. Meet the Professor

1.1. Instructor Contact Information:

Title: **Assistant Professor**
Name: **Tamer Nadeem**
Office Location: **3204 E&CS Building**
Office Hours: **Monday – 2:30-4:00pm**
or by appointment
Email Address(es): **nadeem@cs.odu.edu**
Telephone Number(s): **757-683-7718**

1.2. About the Professor

Teaching and Education Background

Dr. Tamer Nadeem joined the department of computer science at ODU as an assistant professor in January 2011. He received his Ph.D. degree in Computer Science from the University of Maryland, College Park in 2006. Prior to joining ODU, he spent 5 years as a research scientist at Siemens Corporate Research (SCR) in Princeton, USA in which I led several research projects in the general areas of smart mobility for optimized intelligent transportation systems and dynamic radio management for enterprise and industrial wireless networks. Since I joined ODU, I have taught the wireless networked systems course once a year. I also introduced new course on application development for smart devices focusing on programming for Android devices and have been teaching it once a year. I also introduced new seminar course in Spring 2013 on physical cyber systems. The success of the seminar course encourage me to continue the trend and plan to introduce another seminar course on smart sensing in Fall 2016.

Research Interests

While at SCR, Dr. Nadeem led the development of several projects in the area of dynamic radio management for enterprise wireless networks, efficient cross layer protocols for vehicular networks, intelligent transportation systems, location estimation and tracking of WLAN devices, and statistical characterization of VoWLAN. From 2004-2005, Dr. Nadeem was at Fujitsu Labs of America (FLA) in College Park, USA. While he was at FLA, he participated in designing, analyzing, and evaluating new paradigms for 802.11 wireless networks such as access points

with sectorized antennas. Dr. Nadeem holds two US patents (#7,171,558, 2007 & #7,630,343, 2009) and has 15 pending patents. He has over 45 publications in peer reviewed scholarly journals and conference proceedings. He serves on the organizing committees of several conferences including recently ACM MobiCom 2013, ACM MobiSys 2013, and ACM HotMobile 2012. Recently, he served as program chair of the International Wireless Communications & Mobile Computing Conference (IWCMC) - Mobile Computing Symposium for years 2013, 2014 & 2015. He serves as a member of the technical committees of various conferences in the areas of wireless management and vehicular networking. He also serves on several panels on vehicular networking. Dr. Nadeem's technical interests include wireless management for smart devices and enterprise networks, vehicular networks, intelligent transportation system, cyber physical systems, smart grid communication, network security, mobile and pervasive computing, and location determination systems.

Selected Papers and Publications

My ODU CS website (http://www.cs.odu.edu/~nadeem/) has more information about me, including my research interests, research projects, and publications.

2. Course Information

2.1. Course Description

This course explores data structures, algorithms for manipulating them, and the practical problems of implementing those structures in real programming languages and environments. Heavy emphasis is placed upon the analysis of algorithms to characterize their worst and average case requirements for running time and memory.

Perhaps more than any other course, CS361 should expand the students "toolbox" of basic techniques for manipulating data at both the conceptual and the concrete level. At the conceptual level, the student will see a broad selection of standard practices and approaches used in program design. At the concrete level, the student will begin what should be a career-long practice of accumulating useful, reusable code units

2.2. Course Overview

In this course you learn different types of data structures and how to implement them using C++. Moreover, you will learn how to analyze these algorithms to characterize their worst and average case requirements for running time and memory.

Course requirements include (1) Class participation, (2) Homework and programming assignments, (3) Mid-term exam, and (4) Final exam.

2.3. Course Meeting

Meeting place: Dragas 1117
Meeting Time: Monday 4:20pm - 7:00pm

3. Course Readings

3.1. Required Materials

REQUIRED TEXT BOOK

Data Structures and Algorithm Analysis in C++ (4th Edition), Mark A. Weiss, Pearson; 4 edition (June 23, 2013), ISBN-13: 978-0-132-84737-7

Other readings are course slides listed on the course page.

3.2. Optional Materials

Another good optional textbook is: "Data Structures with C++ Using STL", William Ford and William Topp, Pearson; 2 edition (July 27, 2001). ISBN-13: 978-0-130-85850-4.

4. Course Prerequisites

The prerequisites for this course are:

- CS 250, Problem Solving and Programming, or CS 333, Problem Solving and Programming in C++.

  I will assume that you are familiar with the basics of C++, including:
  - the various C++ statements and control-flow constructs,
  - the built-in data types,
  - the use of arrays, pointers, pointers to arrays, and linked lists,
  - the use and writing of functions, and
  - the basic use of structs and classes for implementing abstract data types.

  In addition, students should be familiar with certain basic programming techniques that are largely independent of any specific programming language:
  - software design (i.e., top-down design, also known as "stepwise refinement")
  - testing software, including the use of scaffolding code (stubs and drivers) and the selection of test data for functional testing, special values testing, and boundary value testing.
  - debugging, including the use of debugging output, of using automatic debuggers to set breakpoints and trace program execution, and the general process of reasoning backwards from failure locations to the faulty code responsible for the failure.

  If you are uncertain about your preparation in any of these areas, you can review them at the CS333 website.

- CS 252 Introduction to Unix for Programmers
- MATH 163, Pre-Calculus II, or equivalents.

5. Software Requirements

C++ compiler: The "official" compiler for this course is the Free Software Foundation's g++ (also known as gcc or GNU CC), version 4.5 or higher. This is the compiler that the instructor and/or grader will use in evaluating and grading projects. If you have access to other compilers, you may use them, but you are responsible for making sure that their projects can be compiled by the instructor and/or the course's grader using the official compiler.

You may want to develop your programs on the most convenient compiler and then port it over to the Dept's Linux machines for final testing. Please don't underestimate the amount of time that may be involved in coping with subtle differences among compilers.

You can do all work in this course using g++ on the CS Dept Liniux servers via ssh/X. If you like, however, you can obtain the g++ compiler for free from a variety sources.

6. Course Schedule

Table shows class meeting days, topics, and readings. Typically, a weakly assignment will be posted by next day of the class with a due date on Saturday. Dates and topics are subject to change during the course

Class Meeting Days	Topics	Readings
Monday, August 24	Course Logistics: Course structure, grading, assignments, etc. Course Introduction & Object Oriented Design: Abstraction and Abstract Data Types (ADTs), ADTs and Classes, Implementation of ADTs in C++, Composition, and Overloading.	Slides & Weiss - Chapter 1
Monday, August 31	Algorithms & Complexity: Analysis of Algorithms, Best/Worst/Average Cases, Big Oh-notation, and Recursion.	Slides & Weiss - Chapter 2
Monday, September 7	No Lecture - Labor Day Holiday	
Monday, September 14	Sequences: C++ STL, Vectors, Lists, Stacks and Queues.	Slides & Weiss - Chapter 3
Monday, September 21	Trees: General Trees, Tree Traversing, Binary Search Trees.	Slides & Weiss - Chapter 4
Monday, September 28	Trees II: Balanced Trees	Slides & Weiss - Chapter 4
Monday, October 5	Hashing: Sets, Maps, Hash Function, and Hash Tables.	Slides, Weiss - Chapter 4 & Weiss - Chapter 5
Monday, October 12	No Lecture - Fall Holiday	
Monday, October 19	Midterm Exam	
Monday, October 26	Hashing: Hash Tables. Priority Queues: Binary Heaps, Advanced Heaps	Slides, Weiss - Chapter 5 & Weiss - Chapter 6
Monday, November 2	Priority Queues: Binary Heaps, Advanced Heaps Sorting: Insertion, ShellMerge.	Slides, Weiss - Chapter 6 & Weiss - Chapter 7
Monday, November 9	Sorting: Merge, Quick, Heap.	Slides, Weiss - Chapter 7 &

	Graphs: Graphs.	Weiss - Chapter 9
Monday, November 16	Graph (Cont'd).	Slides & Weiss - Chapter 9
Monday, November 23	Graph (Cont'd). Algorithm Design Techniques: Greedy, Divide and Conquer, etc.	Slides, Weiss - Chapter 9 & Weiss - Chapter 10
Monday, November 30	Advanced Data Structures and Analysis, and Review: Red-Black Trees, Garbage Collection, and NP.	Weiss - Chapter 12 (section 12.5)
Monday, December 7	Final Exam - 3:45-6:45pm - Dragas 1117	

7. Course Grading Criteria

7.1. Grading

Your grade in this class will be based on the following:

(Note that these percentages are only approximate and are subject to change, but by no more than 10%.)

Homework/Programming Assignments	**40%**	**These are to be completed individually.**
Midterm Exam	**25%**	
Final Exam	**35%**	

7.2. Grading Scale

The grading scale is as follows:
(+ and - modifiers will be applied as appropriate)

90-100 =	**A**
80-89 =	**B**
70-79 =	**C**
0-69 =	**F**

7.3. Late Assignments

Any assignment submitted after its deadline is considered late. Late assignments are **NOT accepted**. No credit is given for late assignments.