CSCI 311 : Algorithms and Data Structures

Fall 2017

Department of Computer Science & Engineering
Bucknell University

I. Instructor

Fahmida Hamid
Visiting Assistant Professor
Office: D313
Phone: 570 - 577 - 2343
Office Hours: F (1:30 PM - 3:30 PM)
Email: fh017@bucknell.edu

II. Class

- CSCI 311-01: MWF (8:00 AM - 8:52AM) Room #D116
- CSCI 311-01: MWF (11:00 AM - 11:52AM) Room #D137

III. Recitation

- CSCI 311R.41 Thursday (10:00 AM - 10:52 AM) Room #D116
- CSCI 311R.40 Thursday (11:00 AM - 11:52 AM) Room #D116

IV. Textbook

Introduction to Algorithms, third edition, Cormen, Leiserson, Rivest, and Stein, 2009.
MIT Press. ISBN: 978-0-262-03384-8 (hardcover), ISBN 978-0-262-53305-8 (paperback).

V. Description

In this course, you will study algorithms and data structures. You will learn techniques to analyze algorithms and determine time and space complexity of programs. You will also cover some algorithms for solving combinatorial problems. Major topics include time and space complexity, abstract data types, linear data structures, sets, trees and graphs, searching, and sorting.

This course involves programming assignments as well as problem sets. Please think carefully about program design and apply good software engineering practices. Written design specs may be required for programming projects.

VI. Course Outcome

- Students will be able to analyze the asymptotic behavior of algorithms using measures such as big-O, big-Omega, and big-Theta.
- Students will be able to write programs to solve computational problems, choosing appropriate algorithms and data structures beyond the introductory level.

VII. Course Objective

- **Gaining factual knowledge (terminology, classifications, methods, trends):** In particular, in this course you will learn definitions for asymptotic analysis of functions, methods for formulating and solving recurrence relations, classifications of algorithms and data structures in terms of abstract data types (ADTs), and the details of various algorithms.
- **Learning fundamental principles, generalizations, or theories:** The course will cover the motivation for using asymptotic analysis on algorithm performance. It will also cover general algorithmic techniques such as dynamic programming.
- **Learning to apply course material (to improve thinking, problem solving, and decisions):** In this course you will learn to solve simple recurrence relations, and you will implement algorithms covered in class. We will also discuss where asymptotic analysis can be applied, and where other considerations will outweigh it in choosing the best algorithm.

VIII. Prerequisite

CSCI 205 and MATH 241

IX. Attendance

Attendance at all lectures and recitations is expected. While attendance won't always be taken, instructors appreciate knowing why students are absent. Please let your lecture or recitation instructor know in advance of planned absences. Note that regularly missing class is one of the potential special circumstances that could lead to a discretionary reduction in grade.

X. Homework and Programs

You will enjoy (and benefit from) this class more thoroughly if you successfully complete all the assignments. Some of these assignments will be problem sets and some will ask you to write programs. There will be roughly **six** assignments and you will always have at least one week to complete them. Please be neat when writing solutions to problem sets and, with programming assignments, make sure to document your code well (that is, write comments where appropriate).

All assigned work is due on the date and at the time specified. Each student will be given **3 late days** to use during the semester (a weekend counts as one day). When you hand in something late, indicate how many late days you are using and how many you think you have left. Once your late days are exhausted, any further late assignments will be accepted only at the discretion of the instructor, and with a penalty to be negotiated. For example, very late assignments have sometimes been accepted in past years at a substantial (e.g., 50%) penalty. Under certain circumstances, assignments may be accepted late without penalty. For example, written medical excuses from a doctor may be grounds for granting an extension. Also, it is sometimes possible to make prior arrangements for late submissions due to anticipated absences; be sure to talk to the instructor well before the anticipated absence.

If a grade needs to be adjusted, please see your instructor as soon as possible after the return of the assignment.

XI. Reading Assignment

Complete the reading assignments by the date indicated in the lecture schedule. Be prepared to discuss and raise questions about the material.

XII. Grading Policy

Your grade will be distributed as follows:

- HW & Programming Assignments: 30%
- Exam I: 20%
- Exam II: 20%
- Final Exam: 30%

Grades may be adjusted upwards or downwards at the discretion of the instructor in special circumstances. For example, an extremely bad failing grade on the final exam could be the basis for a reduction of the final grade, as could excessive absences from class.

XIII. Co-operative Work

You are expected to **work individually on the problem sets**, and **work on pairs for the programming assignments**. Discussion of high-level design issues (e.g., how the data structures and algorithms discussed in class work and general ideas about how they can be applied and implemented) is allowed. Design of the code itself and the actual coding should be done only by the group members.

Pseudocode for algorithms used in the programming assignments will usually be presented in lecture and/or in the course text. You are strongly encouraged to follow this pseudocode in writing your code because you should have a clear idea of how it works from class. There are numerous ways to implement most of these algorithms, however, and we have observed that students often go to other sources when writing their programs. If you do go to another source for information about how to implement an algorithm, be sure to reference that source in a comment in your code. That includes page numbers for books and URLs for web references.

As is typical in programming, you may at some point require assistance in finding errors in your program. After you have made a good-faith effort to solve such a problem, ask a friend for help or see your instructor. "Help" in this context means assistance in determining what is wrong. You (and your team-mate) are responsible for fixing the problem.

If cooperation is allowed on a problem set, the instructions will specify this, and will explain what level of cooperation is allowed. Typically you will be allowed to discuss issues with other teams or individuals. When the time comes to write down the solutions, however, each team or individual must produce its own document. Please observe Bucknell's Academic Responsibility guidelines and always consult your instructor when in doubt.

Warning: Code from programming projects and problems from assignments will likely show up in exam questions. When working as a team, make sure that both members are involved in all parts of the assignment rather than splitting up the work and each doing half!

XIV. Exams

There will be two one-hour exams and a comprehensive final. The one-hour exams will be **Monday 25 September** and **Monday 30 October**. The final examination will be on **Monday DD MON (8:00 AM - 11:00 AM)**. Missing exams because of illness will require an excuse from a doctor. Make-up exams for excuses other than illness will be given only in extraordinary circumstances and only at the discretion of the instructor. If you expect that you will need a make-up exam, contact your lecture instructor at least one week in advance. If a grade needs to be adjusted, please see your instructor as soon as possible after the return of the exam. All the exams will be closed-book unless I decide it to be otherwise.

XV. Academic Responsibility

Students are expected to read and abide by the principles clearly explained in the Student Handbook in Section V. Additional information can be found in the information on academic responsibility on the Bucknell website. When in doubt, talk to your professor.

XVI. Access Statement

Any student who may need an accommodation based on the impact of a disability should contact his or her instructor privately to discuss the specific needs. Please contact Heather Fowler, Director of the Office of Accessibility Resources at 570-577-1188 or hf007@bucknell.edu, for help in coordinating reasonable accommodations for students with documented disabilities.

XVII. Tentative Schedule

Weeks	Topic	Reading	Assignments
8/21	Administrative Issues and Introduction	Ch 01	
8/23	Basic Assumptions and Approach	Ch 2.1, 2.2	
8/24	Recitation 1: **Summations, Functions, and proof techniques**		
8/25	Insertion Sort		
8/28	Analysis of Iterative Code	Ch 2.3	
8/30	Divide and Conquer, Recurrences, Merge Sort	Ch 3.1	
8/31	Recitation 2: **Running Time of an Iterative Code**		
9/01	Asymptotic Bounds		
9/04	Little-o Notation	Ch 3.1	
9/06	Solving Recurrences	Ch 4.1, 4.4	**HW 01**
9/07	Recitation 3: **Practice Solving Recurrence Relations**		
9/08	Sorting Properties, Elementary Sorting Techniques		
9/11	Elementary Sorting Techniques		
9/13	Priority Queues & Heaps	Ch 6	
9/14	Recitation 4: **Stoogesort**		
9/15	Building Heaps		**HW 02**
9/18	Heapsort, Heapsort vs. Mergesort		
9/20	Quicksort	Ch 7	
9/21	Recitation 5: **Review for Exam I**		
9/22	Quicksort, Median-of-3 Partitioning		
9/25	**Exam I**		
9/27	Sorting in Linear Time: Counting Sort	Ch 8	
9/28	Recitation 6: **Exam I: Post-mortem**		
9/29	Sorting in Linear Time: Radix Sort		
10/02	Dictionaries	Ch 11	
10/04	Hashing: Hash Functions		
10/05	Recitation 7: **Hash Functions and Linear Probing**		
10/06	Hashing: Hash Functions		**HW 03**
10/09	**Fall Break: No Class**		
10/11	Hashing: Open Addressing		

Weeks	Topic	Reading	Assignments
10/12	Recitation 8: **Binary Search Trees**		
10/13	Binary Search Trees (BSTs)	Ch 12	
10/16	Rotations and Randomized BSTs	Ch 13	**HW 04**
10/18	Red-Black BSTs		
10/19	Recitation 9: **Rotation and Insertion in BSTs**		
10/20	Red-Black BSTs		
10/23	Recursion and Dynamic Programming	Ch 15	
10/25	Dynamic Programming		
10/26	Recitation 10: **Review for Exam II**		
10/27	Dynamic Programming		
10/30	**Exam II**		
11/01	Graphs: Background and Representation	Ch 22	
11/02	Recitation 11: **Exam II Postmortem**		
11/03	Graphs: Breadth-First Search and Depth-First Search		
11/06	Graphs: Minimum Spanning Tree: Prim	Ch 23	
11/08	Graph: Minimum Spanning Tree: Kruskal		**HW 05**
11/09	Recitation 12: **Prim's and Kruskal's Algorithm**		
11/10	Graph: Dijkstra's Single Source Shortest Path Algorithm	Ch 24	
11/13	Union-Find Algorithm for Equivalence Relations		
11/15	String Search: Rabin-Karp	Ch 32	**HW 06**
11/16	Recitation 13: **Dijkstra's Algorithm**		
11/17	String Search: Hash function for Rabin-Karp		
11/20	Thanks Giving Holiday		
11/22	Thanks Giving Holiday		
11/24	Thanks Giving Holiday		
11/27	String Search: using Finite Automata	Ch 32	
11/29	String Search: Knuth-Morris-Pratt Algorithm		
11/30	Recitation 14: **String Search**		
12/01	Review Class		
12/04	Review Class		

XVIII. Special Instructions

Laptop I encourage the students to bring their laptop in the class and in the recitations so that we all can practice some problems in class.

Syllabus This syllabus may be modified as the course progresses. Notice of such changes will be announced in class or through Moodle.

Have a Great Semester!