Miami University
College of Engineering and Computing
Course Syllabus
Fall 2014

CSE 274 B/D - Data Abstraction and Data Structures

Instructor

Pierre St Juste, 205J Benton Hall
Phone: 513-529-0353
Email: stjustpt@miamioh.edu

Office Hours

Tuesday	1:00 pm – 5:00 pm
Thursday	1:00 pm – 5:00 pm
Friday	by appointment

Meeting Times and Location

Section B: Tuesday/Thursday	10:00 am – 11:20 am	006 Benton Hall
Section D: Monday/Wednesday	4:00 pm – 5:20 pm	008 Benton Hall

Important Dates

Monday, September 1	Labor Day (no classes)
Friday, September 12	Last day to drop the class without a grade
Friday, October 10	Fall Mid-term Break (no classes)
Monday, October 27	Last day to drop the class with with a "W"
Thursday, November 27	Thanksgiving
Saturday, December 6	Last day of classes at Oxford Campus

Section B

Tuesday, September 30	Exam 1
Tuesday, November 4	Exam 2
Tuesday, December 9	Final Exam (10:15 am – 12:15 pm)

Section D

Wednesday, October 1	Exam 1
Wednesday, November 5	Exam 2
Wednesday, December 10	Final Exam (3:00 pm – 5:00 pm)

Course Description (from Catalog)

Abstract data types and their implementation as data structures using object-oriented programming. Use of object-oriented principles in the selection and analysis of various ADT implementations. Sequential and linked storage representations: lists, stacks, queues, and tables. Nonlinear data structures: trees and graphs. Recursion, sorting, searching, and algorithm complexity.

Prerequisite Courses: CSE 271 (with C- or above) and MTH 231 ***or***
CSE 271 (with C- or above) and MTH 222, MTH 251, and ECE 287.

Required Text

Open Data Structures (in C++) by Pat Morin. This is an online book and you can download the PDF at http://opendatastructures.org/.

Learning Outcomes

The online course descriptions on the department's website elaborates the learning outcomes of this course. Please read it for more details, it can be accessed via the link below:
http://miamioh.edu/cec/academics/departments/cse/academics/course-descriptions/cse-274/index.html

The main course objectives are to be able to:

- To use appropriate data structures/data types, and algorithmic methods in problem solving
- To create implementations of data structures and abstract data types
- To apply object-oriented principles when implementing data structures
- To apply the use of C++ in the development of object-oriented programs
- To determine time and space requirements of data structure implementations and algorithms

Course Website

This course will use the Niihka course management system which can be accessed through the myMiami portal. All updates for syllabus, homeworks, exam dates, and announcements will be posted there, so please check regularly.

Grading Scale

A+	100 – 97	A	96.9 – 92.5	A-	92.4 – 90
B+	89.9 – 87	B	86.9 – 82.5	B-	82.4 - 80
C+	79.9 – 77	C	76.9 – 72.5	C-	72.4 – 70
D+	69.9 – 67	D	66.9 – 62.5	D-	62.5 – 60
		F	59.9 – 0		

Measures of Evaluation

Exam 1	15%
Exam 2	15%
Final Exam	20%
Homeworks	40%
In-class Labs	10%

Course Policies

Exams: All exams will be cumulative and closed-book. You are not allowed any outside material on the exams. This includes (but is not limited to) books, notes, calculators, smart phones, and laptops. Any student requiring the rescheduling of exams for foreseeable reasons (e.g. religious holidays) *must notify the instructor by the second week of classes*.

Homeworks: All homeworks will be submitted through Niihka by 11:59 pm of the deadline. Late homeworks will not be accepted under any conditions. The lowest homework grade will be dropped.

In-class Labs: There will be graded in-class labs given at the end of most lectures. These labs will not be announced. You cannot make-up lab assignments. The lowest two lab grades will be dropped.

Attendance: The in-class labs will serve as the attendance record, so plan on attending every lecture.

Statement of Community and Non-Discrimination

Miami University is committed to fostering a supportive learning environment for all students irrespective of individual differences in gender, race, national origin, religion, handicapping condition, or age. Students should expect, and help create, a supportive learning environment free from all forms of prejudice. Disparaging comments, sexist or racist humor, or questioning the academic commitment of students based upon these individual differences are behaviors that undermine our learning community. If such behaviors occur in class, please seek the assistance of your instructor or department chair.

Academic Integrity Policy

You must read and understand the CSE department expectations for Academic Integrity appended at the end of this syllabus. You can access the University's policy which can be found in the Student Handbook at http://www.miamioh.edu/handbook.

Students with Disabilities

If you have a documented disability and need special accommodations in this course, you must contact the Office of Disability Resources, 19 Campus Avenue Building. Once you submit the required documentation, they can determine what accommodations, if any, you will be given by your instructor. You will also receive paperwork with which to notify your instructor. For more information, refer to Chapter 3 (Part 4: Health and Safety) of the Student Handbook at http://www.miamioh.edu/handbook.

Acknowledgements

Portions of this document were written by Daniela Inclezan, Norm Krumpe, John Karro, Bo Brinkman, Mike Helmick, Al Sanders and the staff of Miami's CSE Department.

Computer Science and Software Engineering
Academic Integrity Expectations for
Individual and Group Problem Solving Assignments

The Department of Computer Science and Software Engineering is committed to maintaining strict standards of academic integrity. The department expects each student to understand and comply with the University's Policy on Academic Integrity: http://www.miamioh.edu/handbook. **Students may direct questions regarding academic integrity expectations to their instructor or to the department chair.** All work submitted must be **original** for that class. Submitting the same project for two different classes is grounds for charging a student with academic misconduct unless prior written permission is received from **both** instructors.

"Problem Solving Assignments" are assignments that involve **programming, math, proofs, derivations, and puzzles**. The purpose of a problem solving assignment is for you to develop the skills necessary to solve similar problems in the future. To learn to solve problems you must solve the problems and write your solutions independently.

It is worth reiterating that the important aspect of the assignment is that you actually create the solution from start to finish; simply copying a solution and then **understanding it after the fact is not a substitute** for actually developing the solution.

The notion of academic integrity can be confusing in courses with substantial problem solving because certain forms of collaboration and investigation are permitted, but you are still required to complete your assignment independently. The following scenarios are meant to help distinguish between acceptable and unacceptable levels of collaboration and research, but are not all-inclusive:

ACCEPTABLE:

- Consulting solutions from the current course textbook, but not from other published sources.
- Seeking help on how to use the programming environment such as the editor, the compiler, or other tools.
- Seeking help on how to fix a program syntax error or how a certain language feature works.
- Discussing strategies with a fellow student on how to approach a particular problem. This discussion should not include significant sections of completed work or source code (including printouts, email, viewing on a monitor). Discussions should begin with a clean sheet of paper and end with conceptual drawings and/or pseudo-code.

UNACCEPTABLE:

- Looking at another solution including those written by current students, past students, or outside sources such as code or solutions found on the Web, or in publications other than the current class textbook.
- Using another solution as a starting point and then modifying the code or text as your own work.
- Providing a copy of your solution or a portion of your solution, in any form (electronic, hard copy, allowing another student to view your code on a monitor), to another student.
- Giving or receiving code fragments to fix a problem in a program.

If you are stuck on a problem and you are tempted to search for a solution on the Web or to look at another student's solution **STOP** and email or ask your instructor for help.