CS 271 – Data Structures
Fall 2018

Instructor: George Thomas **Office:** Halsey 218
Email: thomasg@uwosh.edu **Phone:** 424-2069

Office Hours:	MW	9:30-11:00
	TR	12:30-1:30
	Or by appointment	
Lectures:	MTW	11:30-12:30, HS 202
Lab:	R	11:30-12:30, HS 101C

Prerequisites: CS 262 with a grade of C of better AND, for Computer Science majors, either completion of or concurrent enrollment in CS 251 with a grade of C or better.

Required Textbook: *COMP SCI 271: Data Structures*, Online book by *Zybooks*.

Subscription Instructions:

1. Create an account at `learn.zybooks.com`. You are required to use your `uwosh` email as your login. You may need to also provide your ID number.
2. Enter zyBook code:
   `UWOSHCOMPSCI271ThomasFall2018`
3. Subscribe. A subscription is $58 and will last until Dec 28, 2018.

Course Website: UWO D2L

Note: If you have special needs, please come and talk to me at the end of the first class, or as soon as you can.

Course Description

A course surveying the fundamental methods of representing data and the algorithms that implement and use those data representation techniques. Data structures and algorithms include: linked lists, stacks, queues, trees, heaps, priority queues, hashing, searching, sorting, data compression, graphs, recursion. Analysis topics include: elementary big-O analysis, empirical measurements of performance, time/space trade-offs, and identifying differences among best, average, and worst case behaviors.

Learning Outcomes:

- Given a non-recursive algorithm, the student will be able to examine its loop structures, infer its asymptotic runtime, and express its efficiency using big-O notation.
- Given a recursive algorithm, the student will be able to examine its recursive structure, determine and solve the corresponding recurrence relation, and infer the asymptotic runtime of the algorithm using big-O notation.

- Given the description of a computational problem requiring a mixture of search, insertion, and/or deletion operations on collections of data, the student will be able to compare the relative advantages of using arrays, vectors, and linked lists in solving the problem efficiently.
- Given a classical computational problem (e.g., infix-to-postfix conversion, postfix-expression evaluation, Huffman data compression, path planning, minimum-spanning tree computation), the student will be able to trace a solution to the problem using appropriate data structures (e.g., stacks, queues, binary trees, binary search trees, red-black trees, graphs) and to predict the asymptotic runtime of the solution based on the selected data structures.
- Given a collection of unordered data, the student will be able to trace the execution of an advanced sorting algorithm (such as quick sort and heap sort) on this data set.
- Given a set of data keys, the student will be able to trace through a sequence of key insertions, searches and deletions on a balanced tree structure. The student will also be able to discuss the relationship between the number of keys and the execution time of these operations.
- Given a set of data keys, a hash function, a table size, and a collision-handling strategy, the student will be able to trace through a sequence of key insertions and searches, and to discuss how varying the table size, hash function or collision-handling would affect the execution time of these operations.
- Given a graph data structure, the student will be able to implement it using either adjacency lists or an adjacency matrix, to traverse it using either a depth-first or breadth-first strategy, to identify its structural properties (whether it is directed, cyclic, connected, complete), and to trace the execution of one or more classical graph algorithms (e.g., Dijkstra's, topological sort or minimum-spanning tree computation).
- Given a problem requiring the efficient use of a variety of data structures, the student will be able to apply object-oriented design principles in implementing and testing a solution to that problem in an appropriate object-oriented language.

Course Grading Policy: Your final grade for this course will be based on five components, namely exams, assignments, labs, participation activities in the Zybooks online textbook, and unannounced quizzes. Your overall numerical grade for the course will be computed as the weighted sum of the component grades using the following weights:

Component	Weight
Exams (4)	45%
Assignments (4-6)	25%
Labs (10)	18%
Participation Activities (Zybooks)	6%
In class quizzes (9-12)	6%

Tentative Exam Dates are as follows:

- **Exam 1 – Thursday, 10/04**

- **Exam 2 – Thursday, 10/25**
- **Exam 3 – Thursday, 11/15**
- **Exam 4 – Wednesday, 12/12**

Your letter grade for the course will be computed as follows:

Numerical Score	Grade	Numerical Score	Grade
>=92	A	72-78	C
90-92	A-	70-72	C-
88-90	B+	68-70	D+
82-88	B	62-68	D
80-82	B-	60-62	D-
78-80	C+	<60	F

While this overall grading scheme is fixed, I will be happy to discuss any issue you may have with individual grades. If you notice a mistake or have a question regarding a specific grade, please come and talk to me *as soon as possible*. Do not wait until the end of the semester to bring up grading issues.

Attendance and Participation: You are expected to not only attend **every** class meeting on time but also to come **prepared** for and **participate** actively in it. Necessary preparation requires you to have studied and assimilated the material covered in previous sessions, to have met with me outside of class to discuss any questions you may have, to have done any assigned labs or assignments on time. I **strongly encourage you to ask any question** or raise any issue you have with the course either during or at the end of class, or during my office hours. I will also gladly meet with you by appointment.

Assignment, Lab and Participation Activity Deadlines: Each lab and participation activity will come with a deadline (day and time) by which it must be submitted. **Late lab or participation activity submissions will NOT be accepted.**
Each assignment will also come with a deadline (day and time) by which it must be submitted. You are allotted *three* assignment credit days you can use through the semester. A credit day is exactly 24 hours or less. You can use unused credit days to submit an assignment after its deadline, without penalty. **Any assignment submitted after the deadline, plus any credit days you have unused, will receive a zero.**
For example, if you have 2 unused credit days available and an assignment is due on Tuesday at 5:00PM, you can submit it anytime by exactly Thursday at 5:00PM without penalty. Do note that if you submit your assignment on Thursday at 5:01PM, you will be penalized 100% of the score of the assignment and thus receive a zero! Note also that if you submit your assignment on Wednesday at 5:01PM, you will be charged two credit days (but no penalty, obviously).

Extensions on deadlines may be granted at the discretion of the instructor if you provide a valid justification (in the form of a written excuse from a medical doctor or the Dean of Students Office) **before** the due date. If you miss a scheduled exam (tentative dates are

provided), you **may** be able to take a make-up exam provided you give the instructor a valid justification (see above) ahead of time if possible. Only one make-up exam will be given. It will be a comprehensive exam scheduled at the end of the semester. Similarly, there will be no make-up quizzes unless the instructor is provided with a valid justification (see above) for your absence on the day of the quiz.

Collaborating versus Cheating: Unless otherwise stated in the assignment or project, all submissions must be entirely your own work. While it is acceptable to discuss the assignments and labs at a high level (for example, at the design level) with others, you must submit your own work. You may not "borrow" any piece of code or design of any length from someone else, the internet, or any other source, unless you can live with a zero and the other potential academic sanctions of cheating (see UWO Student Discipline Code 2007, Chapter UWS 14).