RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Electrical and Computer Engineering

# Course Outline (W2011)

COE428: Engineering Algorithms and Data Structures


**Instructors**

Dr. Reza Sedaghat
Office: ENG-431
Phone: (416) 979-5000 ext 6083
Email: rsedagha@ee.ryerson.ca
Office hours: *(if known)*

Dr. Olivia Das
Office: ENG-464
Phone: (416) 979-5000 ext 6114
Email: odas@ee.ryerson.ca
Office hours: *(if known)*


**Prerequisites** COE318 and COE328 (Corequisite: MTH314)

**Course Website** www.ee.ryerson.ca/~courses/coe428

**Compulsory Text** Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest. Introduction to Algorithms, MIT, 2002, ISBN: 0-07-013151-1 (McGraw-Hill).

**Reference Texts**

1. Knuth, Donald Ervin, The Art Of Computer Programming (3 volumes) Addison-Wesley, 1977. This is the classic book on computer programming, algorithms, and data structures. It is very mathematical and also has extensive problems and solutions.

2. Brian W. Kernighan and Rob Pike, The Practice of Programming, Addison-Wesley, 1999, ISBN:0-201-61586-X, 267 pages. This excellent book will help you hone your programming skills in any language (although the emphasis is on C).

3. Standish, Thomas A., Data Structures, Algorithms and Software Principles in C, Addison-Wesley 1995, ISBN: 0-201-59118-9

4. Brian W. Kernighan, Dennis Ritchie, The C Programming Language, Prentice-Hall, 2nd edition 1988. This is the classic book on C, written by the language developers.

**Calendar Description** The main topics covered in this course include basic data structures (arrays, pointers), abstract data structures (trees, lists, heaps), searching, sorting, hashing, recursive algorithms, parsing, space-time complexity, NP-complete problems, software engineering and project management, object-oriented data structures. Case studies and lab exercises will be implemented using a high level programming language.

**Learning Objectives**

**At the end of this course, the successful student will be able to:**

1. Develop knowledge of designing and analyzing algorithms that can be used to solve various computational problems in the domain of engineering. (1a)
2. Apply mathematical principles to determine the run-time complexity (best, worst and average cases) of various algorithms. Learn about various asymptotic notations used for bounding the algorithm running times and develop knowledge on how to solve recurrence equations. (1b)
3. Learn about various data structures (e.g. stacks, queues, linked lists, binary search trees, red black trees, priority queues, heaps, hash tables etc.) that can be applied to construct efficient algorithms for a variety of engineering problems (1c)
4. Compare different approaches for designing algorithms such as incremental approach versus divide-and-conquer approach. Learn how to select the best design alternative for a given input size of a problem. (4g)
5. Analyze the efficiency of various Graph algorithms that are used in solving network related problems. The analysis will help to rank/rate alternative algorithms for a given problem. (4f)

*Note: Numbers in parentheses refer to the graduate attributes required by the Canadian Engineering Accreditation Board. For more information, see:*
http://www.feas.ryerson.ca/quality_assurance/accreditation.pdf

**Course Organization**

3 hours of lecture per week for 13 weeks.
2 hours of lab per week for 12 weeks.

**Course Evaluation**

| | |
|---|---|
| Labs | 30% |
| Mid-term examination | 25% |
| Final examination | 45% |
| Total | 100% |

**IMPORTANT:**

- In order to achieve a passing grade in this course, the student must achieve an average of at least 50% in both theoretical and laboratory components.
- All the Labs have to be done individually.
- Two week labs carry double weight than one week labs.

**Examinations**

Mid-term exam in Week 7, one hour, closed-book (covers Weeks 1-6).
Final exam, during exam period, three hours, closed-book (covers Weeks 1-13).

**Course Content**
NOTE: This is a preliminary schedule and is subject to change and modifications.

| Week | Topic | Topic descirption | Hours |
|---|---|---|---|
| 1 | Introduction: Course overview. Introduction to algorithms | Role of algorithms in solving various computational problems. | 3 |
| 2 | Analyzing and designing algorithms | Incremental approach, divide-and-conquer approach, insertion sort algorithm, merge sort algorithm, introduction to recursion, comparison of the two sorting algorithms. | 3 |
| 3 | Complexity analysis | Growth rate of functions, asymptotic notations. | 3 |
| 4 | Recurrence equations | The substitution method, the recursion tree. | 3 |
| 5 | Elementary Data Structures | Stacks, queues, linked lists, max heaps, min heaps, basic operations on these data structures, maintaining the maxheap property, building a max heap, heapsort algorithm. | 3 |
| 6 | Hash Tables | Direct address table, Hash table, basic operations on these data structures and the complexity analysis of those operations, hash functions, collision resolution using chaining | 3 |
| 7-9 | Trees and Priority Queues | Binary search trees, insertion, deletion; Red Black trees, rotation, insertion, deletion; Priority queues. | 9 |
| 10 | Graphs | Undirected graph, directed graph, weighted graphs, representations of graphs, adjacency-list representation, adjacency-matrix representation | 3 |
| 11-12 | Elementary Graph Algorithms | Breadth-first search, depth-first search, minimum spanning trees, Kruskal’s algorithm, shortest paths, Dijkstra's algorithm, Bellman-Ford algorithm. | 6 |
| 13 | Review | | 3 |

**Laboratory**

| Lab# | Week | Topic | Detail | Hours |
|---|---|---|---|---|
| 1 | 2 | Introduction | Reviews C programming | 2 |
| 2 | 3 | Recursion | Understanding of recursion using Tower of Hanoi problem. | 2 |
| 3 | 4, 5 | Sorting | Implement and analyze insertion sort and merge sort algorithms. | 4 |
| 4 | 6, 7 | State machines | Implement the flow of control from state to state. | 4 |
| 5 | 8, 9 | XML Balancing, stacks, direct-mapped tables, and hash table | Develop an algorithm to determine whether the start and end-tags balance by using a Stack data structure that keeps track of previously read start-tags. | 4 |
| 6 | 10, 11 | Heaps | Design a heap data structure and print it as an XML expression. | 4 |

**Important Notes**

1. All of the required course-specific written reports will be assessed not only on their technical/academic merit, but also on the communication skills exhibited through these reports.
2. Should a student miss a mid-term test or equivalent (e.g. studio or presentation), with appropriate documentation, a make-up will be scheduled as soon as possible in the same semester. Make-ups should cover the same material as the original assessment but need not be of an identical format. Only if it is not possible to schedule such a make-up may the weight of the missed work be placed on the final exam, or another single assessment. This may not cause that exam or assessment to be worth more than 70% of the student's final grade. If a student misses a scheduled make-up test or exam, the grade may be distributed over other course assessments even if that makes the grade on the final exam worth more than 70% of the final grade in the course.
3. Students who miss a final exam for a verifiable reason and who cannot be given a make-up exam prior to the submission of final course grades, must be given a grade of INC (as outlined in the *Grading Promotion and Academic Standing Policy)* and a make-up exam (normally within 2 weeks of the beginning of the next semester) that carries the same weight and measures the same knowledge, must be scheduled.
4. Medical or Compassionate documents for the missing of an exam must be submitted within 3 working days of the exam. Students are responsible for notifying the instructor that they will be missing an exam as soon as possible.
5. Requests for accommodation of specific religious or spiritual observance must be presented to the instructor no later than two weeks prior to the conflict in question (in the case of final examinations within two weeks of the release of the examination schedule). In extenuating circumstances this deadline may be extended. If the dates are not known well in advance because they are linked to other conditions, requests should be submitted as soon as possible in advance of the required observance. Given that timely requests will prevent difficulties with arranging constructive accommodations, students are strongly encouraged to notify the instructor of an observance accommodation issue within the first two weeks of classes.
6. The results of the first test or mid-term exam will be returned to students before the deadline to drop an undergraduate course in good Academic Standing.
7. Students are required to adhere to all relevant University policies including:
   - Undergraduate Grading, Promotion and Academic Standing, http://www.ryerson.ca/senate/policies/pol46.pdf
   - Student Code of Academic Conduct, http://www.ryerson.ca/senate/policies/pol60.pdf
   - Student Code of Non-Academic Conduct, http://www.ryerson.ca/senate/policies/pol61.pdf
   - Undergraduate Academic Consideration and Appeals, http://www.ryerson.ca/senate/policies/pol134.pdf
   - Examination Policy, http://www.ryerson.ca/senate/policies/pol135.pdf
   - Accom.of Student Relig., Abor. and Spir. Observance, http://www.ryerson.ca/senate/policies/pol150.pdf
   - Est.of Stud. Email Accts for Official Univ. Commun., http://www.ryerson.ca/senate/policies/pol157.pdf
8. Students are required to obtain and maintain a Ryerson Matrix e-mail account for timely communications between the instructor and the students.
9. Any changes in the course outline, test dates, marking or evaluation will be discussed in class prior to being implemented.
10. In-class use of cellular telephones is not permitted. Please turn off your cell phone prior to class. Quiet use of laptops, text-messengers and similar non-audible devices is permitted only in the rear two rows of class. This restriction allows use of such devices by their users while limiting audible and visual distractions to other students. This policy may change without notice.