

**INSTRUCTOR:** Tom Naps

**OFFICE:** Halsey 229, phone 424-1388

**EMAIL:** naps@uwosh.edu

**OFFICE HOURS:** MonThurFri 4:00-5:00, Tue 3:30-4:30, Wed 2:00-3:00

**TESTS:**

- Exam 1: Wednesday, March 4
- Exam 2: Wednesday, April 15
- Exam 3: Friday, May 15

**REFERENCES:**

- Absolute C++, 3rd edition, by Walter Savitch. Used for the “C++ component” of the course.
- Daily class handouts. Used for the “Algorithms and Data Structures” component of the course. Organize them, take notes on and about them. Handouts that are not liberally saturated with your own explanatory notes will likely prove useless when you need them most
- Course Web page at <http://plonedev.uwosh.edu/jhave/uwo-data-structures-spring-2009/>.

### Topic Coverage

C++ Chapter and section numbers below are from Savitch.

- Introduction to C++ – 1.1-1.3, 2.1-2.3
- Functions and parameters – 3.1-3.3, 4.1
- Stream I/O – 12.1-12.2, 9.3
- Arrays – 5.1-5.4
- Pointers and arrays re-visited – 10.1-10.3
- Classes and structs – 6.1-6.2, 7.1-7.2
- Linked Data Structures – 17.1-17.3

### Algorithms and Data Structures

- Algorithm Analysis for non-recursive algorithms
- Recursion, algorithm analysis for recursive algorithms, backtracking, algorithm design
- Building effective applications using linear data structures
  - Arrays/vectors
  - Linked lists
  - Stacks
  - Queues
- Higher-powered sorting algorithms and lower bounds on how good they can become
- Trees, binary search trees, heaps, other applications of trees
- Hash tables
- Graphs and their applications

### Learning Outcomes

Given our coverage of these topics, you will be expected to . . .

1. Given an algorithmic specification of a process based on decision and iterative control structures, dynamic memory de/allocation, and/or text-based file and console I/O, the student will be able to edit, compile, debug and run in a UNIX/Linux development environment a C++ program that uses pointers, the new and delete operators, the iostream library and/or other predefined classes in the STL to correctly implements the algorithm.
2. Given a problem or task description, the student will be able to develop a C++ solution based on user-defined classes and structs, the reuse of standard library functions and classes, as well as programming language features not available in Java (e.g., selecting by-value or by-reference parameter passing, taking advantage of operator overloading).
3. 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.
4. 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.
5. 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.
6. 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.

7. 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.
8. 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.
9. 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.
10. 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).
11. 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 Policies

Your grade for the course will be based on the following weighted factors:

Factor	Weight
Class participation and preparation	10%
C++ Labs/Assignments	22.5%
Algorithms and Data Structures Labs/Assignments	22.5%
3 exams:	
Exam 1	15%
Exam 2	15%
Exam 3	15%

At the end of the term, your work in all of these areas will contribute to a numerical grade for the course based on a 100-point scale. Grade cutoff levels on this final scale are:

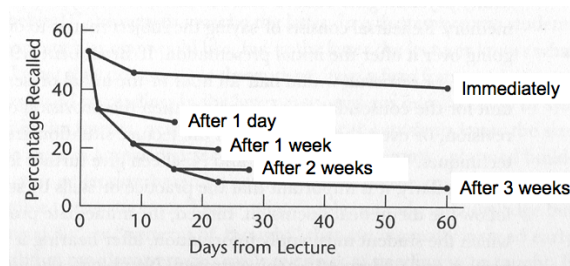
A $\geq$ 92	B $\geq$ 82	C $\geq$ 72	D $\geq$ 60
AB $\geq$ 89	BC $\geq$ 79	CD $\geq$ 69	F < 60

## FAQ

**Do I have to come to class?** You are expected to arrive prepared to ALL the course sessions. Furthermore you are expected to participate in the classroom discussions and activities to the best of your abilities. It is difficult to envision a student missing and/or arriving unprepared to a number of the class sessions and still succeeding in the course.

**How much time will this course take?** Figure about three hours outside of class for each hour in class. That heuristic makes being a full-time student pretty much equivalent to holding a full-time job, so this is really good preparation for the real world that awaits you after graduation.

**How can I best prepare for the exams?** We've known what the following graph illustrates since 1968:



Bassey M. (1968), in *Learning methods in tertiary education*

Consequently at the end of most of our class periods I will give you one or more exam review problems. The time to work on these review problems is immediately after the material is covered in class. If you have participated in class the day the review problem was distributed, have made a good faith effort to work on the review problem, and are “stuck” on it, I will be more than happy to help you with it if you come my office anytime within three days after you have received the review problem in class.

Stuckness shouldn't be avoided. It's the psychic predecessor of all real understanding. (Robert Pirsig – Zen and the Art of Motorcycle Maintenance)

After those three days (not counting weekends), *because you have made the choice to not learn effectively*, you are on your own in terms of grappling with these review problems.

**What if I'm late in submitting a lab/assignment for evaluation?** Each lab and assignment will carry with it a due date. If you are late in submitting it for evaluation, it will be accepted but will be penalized at the rate of 10% of point value the first day late, *an additional 20%* the second, *an additional 30%* the third ...

**Is there any way I can carelessly lose points in the course?** Yes ...

- Be late in submitting your work for evaluation on labs and assignments.
- Don't participate in and prepare for the class.

**What is this class participation/preparation stuff? How does it add up to 10% of my grade? ...**

- Be sure to do those review problems before the next class meeting. If you do that and get them right, you get full credit for them. If you do that but get them wrong, you get half credit. If you don't do them, you get no credit.
- Exhibit your knowledge when called on to explain your correct answer to a review problem.
- Be sure to do well in the online quizzes you will be taking in the JHAVÉ algorithm visualization system.
- Exhibit your knowledge when called on to respond to other questions in class

**Is there any way I can get some bonus points? ...**

- Successfully finish and have evaluated by myself or Josh Dean (the lab assistant) the lab activity for a Monday or Friday lab session.
- Do an outstanding job when called on to explain your correct answer to a review problem.

**Can I get an extension on work that is due on a specified date?** Only if you're ill enough to provide a signed note from the attending physician or have other reasons serious enough that the Dean of Students Office is willing to provide a written note justifying the extension.

**If I miss a test, can I make it up?** If you are unable to take a scheduled exam, it may be possible to take a make-up exam provided that you do BOTH of the following, which are then subject to my approval:

- Make arrangements prior to the scheduled exam (for last minute emergencies, telephone me at 424-1388 or leave a message at the computer science office, 424-2068). No after-the-fact notifications will be accepted ... *AND*
- Have a written medical excuse signed by the attending physician OR have a note of justification from the Dean of Students Office.

Only one make-up exam will be given. It will be a rigorous comprehensive exam given at an arranged time during the last week of the semester.

**Can I work with others on assignments?** No, not in the sense of two people working on the same program. However, it is acceptable to consult another student for help in debugging a program that you have authored yourself and that is not producing the result you expected. It is also acceptable to cut-and-paste code snippets from in-class demos or examples you find on the internet *provided* that you cite the sources of these code snippets in the introductory documentation block at the beginning of your program.