http://cs.txstate.edu/~lk04/3358/BannerLine1.gif

**Exam 2 Study Guide**

|  |
| --- |
|  |

|  |  |
| --- | --- |
| http://cs.txstate.edu/~lk04/3358/BulletL1_7x15.gif | General |

|  |  |
| --- | --- |
|  |  |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | Date:  **11/02/2016** (Wednesday, 2nd half of class meeting for Sec. 001). |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | Duration: 80 minutes. |

|  |  |
| --- | --- |
|  | Questions will be based on the topics covered since Exam 1, ending with the **10/26/16 lecture** for **Sec. 001**, and the **10/31/16 lecture** for **Sec. 003**): |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Templating* (deeper coverage than in Exam 1, including implementation aspects) |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Introduction to algorithm analysis* |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Linked list* |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Stacks*/*queues* (and applications). |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Recursion* |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Tree Fundamentals* |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Tree Traversals* (and run-of-the-mill processings) |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Binary Search Tree* |

|  |  |
| --- | --- |
|  | Test will be *closed books* and *closed notes*. |

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| http://cs.txstate.edu/~lk04/3358/BulletL1_7x15.gif | Relevant Material |

|  |  |
| --- | --- |
|  |  |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | Most relevant *Lecture Notes*: ***309OrderAnalysisOfAlgorithms01*** through ***320BinarySearchTrees***. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | (Focus on topics/material we spent much time on - diligence in attending classes is significant here.) |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Handouts* (distributed in class) and *examples* (significant number of them posted under **Examples**) covering topics indicated above. |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | *Assignments* 4, 5 and 6 (Part 1). |

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| http://cs.txstate.edu/~lk04/3358/BulletL1_7x15.gif | Other Resources |

|  |  |
| --- | --- |
|  |  |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | You may want to check out ***sample past test/exam questions*** already posted on the class homepage. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | You should ***not*** however, expect the questions to be identical in number, kind, topic coverage, *etc.* |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | *You should not have to worry about questions being written on topics we have not yet covered; some such questions may appear as sample past questions because the associated topics were appropriate at that time.* |

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| http://cs.txstate.edu/~lk04/3358/BulletL1_7x15.gif | Checklist (exhaustiveness ***not*** guaranteed) of some things you are expected to know and/or know how to do/apply: |

|  |  |
| --- | --- |
|  |  |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | Templating. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | How templating is put to good use in ***Assignment 4***. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | What particular situation (that makes templating advantageous) is illustrated in ***Assignment 4***. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | How to develop a template class. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Implementation aspects associated with ***Assignment 4***. |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | Introduction to algorithm analysis. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Nature-of-input-dependent scenarios: *worse case*, *average case* and *best case*. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Big-O notation/characterization. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Upper bound (order on resource requirement growth rate), asymptotic ("in-the-big", "settled-down"), order of magnitude ("broad-brushing"). |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | What it can (is intended to) capture (and what is not captured): implications and common misconceptions. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Common categories and their growth-rate behavior and relative ordering: O(1), O(log n), O(n), O(n log n), O(n2). |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Be able to quickly inspect a code segment (involving the basic flow-of-control constructs - sequence, selection, repetition) and characterize it (as "tightly" as can be determined). |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | Linked list. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Array versus linked list: strengths and weaknesses. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | *Random access*, *insertion*/*deletion anomaly*, *resizing woe*. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | When to use which to take advantage of the strength(s) and minimize/avoid the weakness(es). |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | (There's no one "panacean data structure" -> use the right tool for the right problem.) |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Be able to design/implement functions that manipulate linked list(s). |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Manipulate: add, delete, modify, search, inspect, ... (may be in combination within a function). |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Common coding idioms/"patterns": what they mean and when to use. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | **cursor = cursor->link;** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | **while (cursor != 0) { ... }** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | **while (cursor->link != 0) { ... }** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | ... |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | How head pointer(s) should be passed to a function: *by value* or *by reference*. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Be cognizant of and know how to avoid *null-pointer exception*. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | When writing code that dereferences a pointer (at some point), always check that the pointer will never contain the *null address* (at that point). |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | Be careful when writing relational expression involving *short-circuit evaluation*. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Know when to say no: |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | Don't use memory that's not allocated. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | Don't access memory that's already deallocated (if that memory access must be done, do it *before* deallocation). |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | Don't leak away memory. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Be able to read/understand C++ code that manipulates linked list(s) and *identify any bugs* (and associated problems that can arise from them). |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | Stacks/queues and applications. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Containers restricted in specific ways (especially wrt addition and removal of data items) to support commonly required operational characteristics. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | LIFO with stacks and FIFO with queues |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Fundamental operations (besides construction and destruction), STL-style. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Stacks:**push**,**pop**,**top**,**empty**,**size**. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | **top + pop**for *traditional pop*. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Queues:**push**,**pop**,**front**,**empty**,**size**. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | **push**for *traditional enqueue*. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | **front + pop**for *traditional dequeue*. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Error conditions: *underflow* (always possible) and *overflow* (implementation and system resource dependent). |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Keys reasons for their usefulness: LIFO and/or FIFO buffering, reversal and echoing capabilities. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Applications (all related to LIFO and/or FIFO buffering). |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Reversal and echoing. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | How to put these effects to use (expressing algorithms involved in *pseudocode*). |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | For examples on how to clearly write pseudocode involving stacks and queues, see "***STL\_Stack\_Queue\_Example***", "***Implementing Queue Using 2 Stacks***" and "***StackQueueAppEg02\_LevelTravOfLLofLL\_Pseudocode***" posted under **Examples**. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Invocation (function-calling) flow of control support/management: system/call/run-time stack. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | (Underlying LIFO rationale revealed here - more details to come when covering recursion.) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | *Level* (*breadth-first*) *traversal* of non-linear data structures. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Concept of *priority queue*. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | STL in perspective. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | How to use STL stack and queue templated containers (**chk\_pal**). |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Implementations using array and linked list. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Using either to implement *stack* is pretty straightforward and mundane - stack has "only 1 door". |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Using either to implement *queue* is more complicated and interesting - queue has "2 doors": |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | *Circular array*. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL6_6x12.gif | **size\_type next\_index(size\_type current\_index, size\_type capacity)**  **{ return (current\_index + 1) % capacity; }** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | *Circular linked list*. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL6_6x12.gif | Where should *front* and *rear* be and *why* (especially in regard to*push* and *pop*)? |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | Recursion. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Recursive thinking |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Dealing with the seemingly infinite in finite fashion |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | "Divide-and-conquer" + "results of division are identical and smaller versions of the  original (what gets divided)" |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | 4 criteria for successful application: recursively decomposible, base case(s), making progress, ultimate reachability of base case(s) |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | How function-calling (recursive functions included) is typically implemented with the help of system/call/run-time stack ... |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | *Activation records* (*stack frames*) |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | and how to use that to trace recursive functions |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Design/implement recursive algorithms for given problems (including problems that involve arrays, linked lists and trees) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Most important first hurdle - express/formulate problem in terms of smaller problems of the same type |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Tail recursion |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Indirect recursion |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Advantages/disadvantages |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Conceptual elegance/clarity and code compactness |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Function-call overhead and stack-overflow risk |

|  |  |  |
| --- | --- | --- |
|  | http://cs.txstate.edu/~lk04/3358/BulletL2_5x12.gif | Trees |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | General properties of trees and specific properties of special trees covered |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Tree versus graph |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | *n-ary tree* -> binary tree -> binary search tree and AVL tree, heap |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | AVL tree and heap only mentioned in passing (as heads up of what's to come). |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Most (if not all) "traits" related to a tree must be *recursively applied* (for certain terms such as *balanced*, *binary search tree*, *in-order traversal*, *etc.* to be true) |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Representation of binary tree |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Pointer-based representation |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Efficient representation of *complete* binary tree using array |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | *Traversal* (and processing) of trees |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Breadth-first (level): recall the usefulness of queue. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | How algorithm for "breadth-first traversal of linked list of linked lists" is adapted. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Depth-first: *pre-order*, *in-order* and *post-order*. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | (Relating to *prefix*, *infix* and *postfix* expressions can be helpful: what replaces "where, relative to the *operands*, a binary *operator* is placed".) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Most tree manipulations would be very difficult (at best) for us to track if we don't do it *recursively* |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | http://cs.txstate.edu/~lk04/3358/BulletL3_5x12.gif | Binary search trees (BST) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | What is and why. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | Binary tree with storage rule (invariant) to enable binary search and ordered access. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL6_6x12.gif | Which traversal algorithm leads to which kind of ordering? |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Insert, search and remove operations. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | Focus on the conceptual. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | (Simple algorithms to reach the "low-end" and "high-end" nodes.) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL4_5x12.gif | Performance aspects. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | Is searching always guaranteed to be logarithmic? If not, how can logarithmic behavior be attained? |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | http://cs.txstate.edu/~lk04/3358/BulletL5_6x12.gif | (What would a BST degenerate into in the worst case?) |

|  |
| --- |
|  |