Lect 9: Sep 23

1. Tree continued. The binary search tree. In in-order property of a search tree.
2. Representation of a tree:

```
tree = ^ treerecord ;
treerecord = record
     key : integer ;
     left, right : tree ;
     end;
```

3. Searching in a tree.
4. Insertion in a tree.
5. Traversing a tree. Pre, post and Inorder traversal. The code for a stack pre-order traversal.
6. Bredth and Depth first search regimes.

Lect 10: Sep 28

1. Finish material in Sedgewick: representation of forests.
2. Inorder traversal.
3. Recursion.

Lect 11: Sep 29

1. Recursion: more examples, tail recursion, while loop and repeats done recursively. Recursion replaced with iteration and a stack.
2. Algorithmic analysis. Fundamental notions. Problem size and problem families. Average and worst case. Instruction counting and the inner loop. Complexity families,

$$1, logn, n, nlogn, n^2, n^3, 2^n.$$

Lect 12: Oct 5

1. Big O notation. Rules and regulations.
2. Analysis of our homework run times using the big oh.

Lect 13: Oct 7

1. Sorting: Selction, Insert, Bubble, Shell, with some analysis.
2. Quick: worst, best and average case analysis.

Lect 14: Oct 12

1. Quick sort, last words.
   (a) Reduction in size of stack to $O(logn)$ in the worst case
   (b) Using insertion sort for the small recursive partitions
2. Merge sort, using a linked list.
3. Heap sort, idea, ADT def. of a heap, using function map notation, and categorization of efficiency of operations.
4. Implementation of a heap in two ways using arrays, leading to $O(n)$ in either insert or deletemin.

Oct 14, Midterm.

Lect 15: Oct 19

1. Went over Midterm
2. Went over solution to Homework 6 (as part of the solution to Homework 6) deletion in binary trees.

Lect 16: Oct 21

Heaps, continued ... Heap order in trees, array based implementation How to insert and delete min, use in a sorting algorithm.

Lect 17: Oct 26

1. Linear time build of a heap, Analysis ... see notes
2. Bucket sort. Screwed this one up, need to be certain about numbers always going small to large as one scans any pile bottom to top.
3. Balanced binary trees. 2-4 trees, and red-black trees.

Lect 18: Oct 28

1. 2-4 trees, red-black trees and splay trees. Discussion of homework 7.

2. Notes: waters were muddied w/ 2-4 trees about Sedgewick's notation for leaves (versus internal nodes). Screwed this one up to, not remembering that values go up into a parent, then forgetting which node to send up.

   The rule is:

   $$\text{middle value to parent}$$

   when splitting 4 nodes.

   This includes splitting the leaf, so there are no creations during insert.

   To note:

   (a) now 1-3 values, but no longer 1 child, 2-4 children.

   (b) no creations during insert

   (c) 4 children (3 values) is unstable.

Lect 19 : Nov 2

1. Hashing, Open Chaining, Linear Probing and Double Hashing. Dynamic Rehashing. Here is a sample text, not taken from the class, but from an explanation to a certain student afterwards.

   Take a able of size 17 and a skip value of 4. Here is the sequence of table locations visited:

   $$0, 4, 8, 12, 16, 3, 7, 11, 15, 2, 6, 10, 14, 1, 5, 9, 13$$

   Number 1 is the 13-th element on the list, and we subtracted 17 three times in the sequence of operations leading to 1. So:

   $$1 = 13 * 4 - 3 * 17.$$

   If $d|4$ and $d|17$ then it divides any linear comb, hence $d|1 \Rightarrow d = 1$.

Lect 20 : Nov 4 : off

Lect 21 : Nov 9

Radix Search, Tries and Patricia trees. Real-life example: i-nodes in Unix (complements discussion of FAT table in DOS.)

Lect 22 : Nov 11

Knuth-Morris-Pratt and Rabin-Karp String matching.

Lect 23 : Nov 16

NDFA and regular expression matching, Sedgwick's book chap 20 , Turing Machines, finite automata

Lect 25 : Nov 18

Turing machines, introduction of Homework 10, how to program a Turing machine.

1. Duplicating a string. Introduction of notation, $\Sigma^*, \Sigma^+, \alpha^n, \alpha \in \Sigma$. The method was to select a character outside the input language and to place two stars, *, one over the copy from character, one over the copy to position. And to chase back and forth from star to star.

2. Using $/xxxxxxx/$ to demark "records".

Lect 26 : Nov 23

Introduction to extra homework assignment and description of shaker sort. The proof of its run time. In particular to proof that: In a poset on n elements, there is either a chain of length $\sqrt{n}$ or an antichain of size $\sqrt{n}$. Including all the nec. def's (poset, incomparable, chain, antichain.)

Lect 27 : Nov 30

Parsing and Context-Free Grammers, Chap 21 in Sedgewick's

Lect 28 : Dec 2

File compression, Run-length codes and Huffman codes Chap 22 of Sedgewick's.

Dec 14, Final exam