CSI 33 LECTURE NOTES (Ojakian)

Topic 8: Trees

OUTLINE
(References: Ch. 6, 7)

1. Trees
2. Preorder, postorder, inorder
3. Binary Search Tree

1. Trees: Basic Terminology

   (a) Vertices, Nodes, Edges

   (b) Rooted vs. non-Rooted Tree

   (c) Root, Parent, Child, Sibling

   (d) Binary Tree

   (e) Example

   PROBLEM 1. *Discuss a basic decision tree for guessing what animal someone is thinking of (more to come on decision trees later ...).*

2. Programming Binary Trees

   (a)

   PROBLEM 2. *Create just a TreeNode class and build the above decision tree from scratch.*

   (b) Typical "addressing" in binary trees.

   PROBLEM 3. *Create a BinaryTree class that uses this addressing method to: 1) add nodes and 2) get node data.*

3. Application: Data Compression

   PROBLEM 4. *Consider Problem 8 (pages 249 - 251), and discuss how to do* **without** *data compression.*

   PROBLEM 5. *Reconsider the last problem, but now* **with** *data compression.*

   PROBLEM 6. *Use the BinaryTree class to program the data compression.*

4. Prefix and Postfix Expressions

   General Intuition: Reading left to right, for postfix, the operation comes after its two operands, while for prefix, the operation comes before its two operands.

   (a) Recall postfix (philosophy: wait for an operator, then look back at two operands - i.e. operator after operands). We can call this an "Operator Triggered Stack".
      i. Read left to right.
      ii. If next item is an operand, push onto stack.
      iii. If next item is an operator, pop last two items, evaluate, then push the result on the stack.
      iv. The single number on the stack at the end is the answer
   (b) Prefix (philosophy: wait for two operands, then look back at the operator - i.e. operator before operands). We can call this an "Operand Triggered Stack".
      i. Read left to right.
      ii. Push the next item onto the stack.
      iii. If the top two items on the stack are operands, then pop the top two operands and apply the next popped operator, evaluate, then push the result on the stack.
      iv. The single number on the stack at the end is the answer
   (c) Another approach to Prefix: Do the postfix approach (i.e. Operator Triggered Stack), but from right to left, instead of from left to right.
   (d) Another approach to Postfix: Do the prefix approach (i.e. Operand Triggered Stack), but from left to right, instead of from right to left.
   (e)

      PROBLEM 7. *Calculate the following postfix expression:* $5\ 2\ +\ 8\ 3\ -*$
   (f)

      PROBLEM 8. *Calculate the following prefix expressions*

      i. $*\ 9\ +\ 2\ 6$

      ii. $+\ 7\ *\ 45\ +\ 2\ \ 0$

5. Recursion

   (a) Wrapper function and Recursive Helper function

      PROBLEM 9. *See recursion examples: Fibonnaci and list sums*
   (b) Check the recursion works using idea of Proof-By-Induction

      PROBLEM 10. *Verify the last recursion examples (Fibonnaci and list sums) using Proof-By-Induction.*
   (c) Recursion on Linked Structures:
      i. Pass a node reference to the recursive function
      ii. Function returns a node reference to the properly modified linked structure
      iii. Notice pattern: link passed-in and the link modified are the same
      iv.

         PROBLEM 11. *Add a recursive append and delete to linked list, thinking inductively.*
         Then verify the operation of the functions using Proof-By-Induction.

6. Expression Tree and Traversals

   (a) **PROBLEM 12.** *For example from book (p. 277, Figure 7.4) do three kinds of traversal.*

   (b) Inorder - corresponds infix expression

   (c) Postorder - corresponds postfix expression

   (d) Preorder - corresponds prefix expression

   (e) **PROBLEM 13.** *Program in-order. Leave other two as group work.*

7. Binary Search Tree

   (a) Its defining property with examples.

   PROBLEM 14. *From diagrams, which are and which are not BSTs?*

   (b) Recursive and non-recursive insertion: Do examples, code, and prove

   (c) Recursive deletion:

      i. First just code and discuss the case where the deleted node has **at most ONE child**

      ii. Discuss theory for case: deleted node has TWO children

      iii. Code nust a deleteMax method

      iv. Then do full delete method

8. Application: Machine Learning and Decision Trees

   For all this, see the coding example.

   (a) Introduction to Data Frames.

   (b) The goal of Machine Learning Classification via the example Data Frame: From features to target.

   (c) Majority Vote and Node Purity (understood probabilistically)

   (d) How to split node on a feature: Maximize purity (understood probabilistically