Course: DD2325 - Exercise Set 6

Exercise 1: *Trees Intro*

- Will review together some of the terminology associated with trees.
- A binary tree is a tree in which every element has at most 2 children. A binary tree is *full* if every node bar the leaves has 2 children. How many nodes does a full binary tree of height `h` have?
- A binary tree with `n` nodes has height at least $\lceil \log_2(n+1) \rceil - 1$.

Definitions for Tree Traversal

Preorder traversal a node is declared visited before its descendents are visited

Postorder traversal a node is declared visited after its descendents have been visited

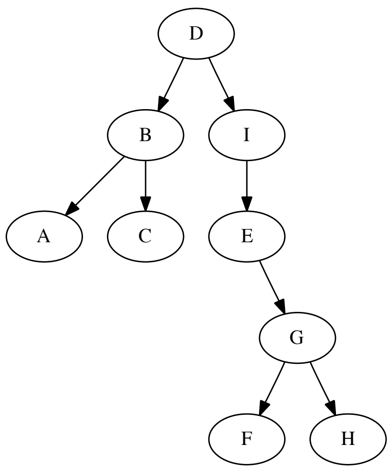
Inorder traversal a node is declared visited after its left subtree has been visited but before its right subtree has been visited

Exercise 2: *Euler Tree Traversal*

Generic traversal of a binary tree. Demonstration on the whiteboard.

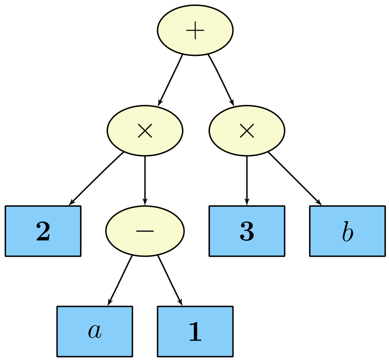
Exercise 3: *Binary Tree Traversal*

For this binary tree, write out the order in which the nodes are traversed for i) *preorder*, ii) *inorder* and *postorder*.



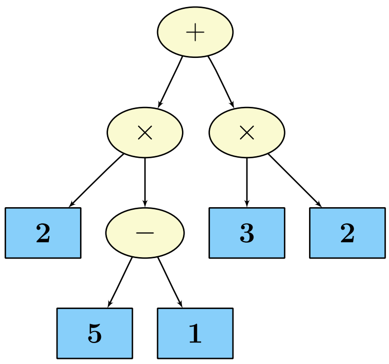
Exercise 4: *Print Arithmetic Expression*

Use *inorder* to traverse the tree in this question and in turn to print out the arthmetic expression it encodes. You can do this by following these instructions



- Print operand or operator when visiting node
- Print "(" before traversing left subtree
- Print ")" after traversing right subtree.

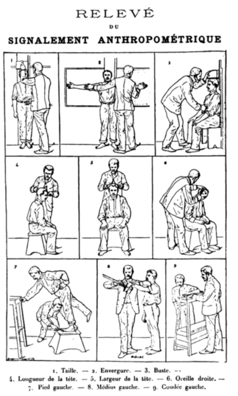
Exercise 5: *Evaluate Arithmetic Expression*



How can *postorder* traversal be used to evaluate the expression encoded in this tree ?

Exercise 6: *Hashing / Binary Search*

In the mid to late 19th century in European cities, the identification of suspects who had been previously arrested (ie habitual offenders) was a very hit and miss process. Thus a criminal could exist under several aliases with his movements and actions being very hard to trace. Photography did exist but there was no clear way of how to search a large database (tens of thousands) of pictures quickly and accurately. Fingerprints had not yet been adopted as the most reliable method for person identification. Then a French detective Alphonse Bertillon began to exploit the idea of **anthropometry** - an identification system based on physical measurements. A human can be uniquely identified by a set of measurements which remain constant over a life time. Bertillon used 11 such measurements:



height, length of torso, length from left shoulder to right middle finger when right arm is outstretched, width of head, length of head, length of torso, length of right ear, length of left foot, length of left elbow to tip of left middle finger, length of left little finger, width of cheeks

Thus as soon as an arrested person entered the person station he had his measurements taken. And then they were checked against those in the police's records to try and identify the criminal. Thus the two important questions are:

- If the police kept one page for each person in their database containing his measurement details, an attached photo, his name and aliases and these pages were then stored in a small number of filing cabinets, how should these pages have been arranged (stored) within the cabinets? Why?
- How should the search have been performed?

Exercise 7: *Huffman Coding Prelims I*

Three different encodings of the symbols `a, b, c, d` are shown in the table. For each of the symbol codes how is `acdbac` encoded? One of the codes is not very useful. Which one and why?

a_i	$c_1(a_i)$	$c_2(a_i)$	$c_3(a_i)$
a	1000	0	0
b	0100	10	1
c	0010	110	00
d	0001	111	11

Exercise 8: *Huffman Coding Prelims II*

We have two different codes for the symbols `a, b, c, d`. In the table p_i denotes the frequency of the symbol strings made from the symbols. What is the average length of the encoding per symbol for each code?

a_i	$c_1(a_i)$	$c_2(a_i)$	p_i
a	1000	0	$\frac{1}{2}$
b	0100	10	$\frac{1}{4}$
c	0010	110	$\frac{1}{8}$
d	0001	111	$\frac{1}{8}$

Exercise 9: *Huffman Coding Prelims III*

What are the properties of a good code?

The Huffman Coding Algorithm

1. Take the two least probable symbols in the alphabet. These two symbols will be given the longest codewords, which will have equal length, and differ only in the last digit.
2. Combine these two symbols into a single symbol, and repeat.

Note: There is no better symbol code for a source than the Huffman code.

Exercise 10: *Huffman Coding*

Use the Huffman coding algorithm to find the optimal binary coding for the symbols and their frequencies shown in the table. What is the expected length of the code

a_i	p_i
a	.25
b	.25
c	.2
d	.15
e	.15

Exercise 11: *Huffman Coding*

Find the optimal binary symbol code using the Huffman coding algorithm

$$\begin{array}{lllllllll} a_i & : & \{\,\texttt{a}, & \texttt{b}, & \texttt{c} & \texttt{d} & \texttt{e} & \texttt{f} & \texttt{g}\,\} \\ p_i & : & \{.01, & .24 & .05 & .20 & .47 & .01 & .02\} \end{array}$$

Exercise 12: *Huffman Coding*

Are there any disadvantages to a Huffman coding for compression purposes?