# CS 473 ✧ Fall 2022

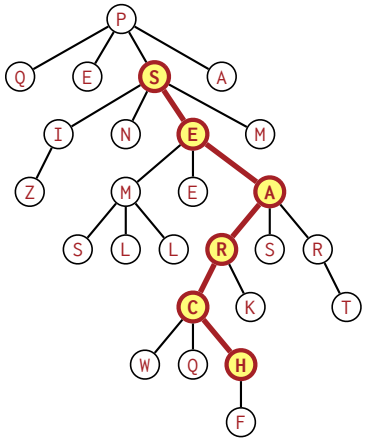
# Homework 6

Due Tuesday, October 18, 2022 at 9pm

---

1. Describe and analyze an efficient algorithm to find strings in labeled rooted trees. Your input consists of a *pattern string* $P[1..m]$ and a rooted *text tree* $T$ with $n$ nodes, each labeled with a single character. Nodes in $T$ can have any number of children. A path in $T$ is called a *downward path* if every node on the path is a child (in $T$) of the previous node in the path. Your goal is to determine whether there is a downward path in $T$ whose sequence of labels matches the string $P$.

   For example, the string SEARCH is the label of a downward path in the tree shown below, but the strings HCRAES and SMEAR is not.




2. A *fugue* (pronounced "fyoog") is a highly structured style of musical composition that was popular in the 17th and 18th centuries. A fugue begins with an initial melody, called the *subject*, that is repeated several times throughout the piece.

   Suppose we want to design an algorithm to detect the subject of a fugue. We will assume a *very* simple representation as an array $F[1..n]$ of integers, each representing a note in the fugue as the number of half-steps above or below middle C. (We are deliberately ignoring all other musical aspects of real-life fugues, like multiple voices, timing, rests, volume, and timbre.)

   (a) Describe an algorithm to find the length of the longest prefix of $F$ that reappears later as a substring of $F$. The prefix and its later repetition must not overlap.

   (b) In many fugues, later occurrences of the subject are ***transposed***, meaning they are all shifted up or down by a common value. For example, the subject $(3, 1, 4, 1, 5, 9, 2)$ might be transposed transposed down two half-steps to $(1, -1, 2, -1, 3, 7, 0)$.

   Describe an algorithm to find the length of the longest prefix of $F$ that reappears later, *possibly transposed*, as a substring of $F$. Again, the prefix and its later repetition must not overlap.

For example, if the input array is

$$\overbrace{3,1,4,1}^{},5,9,\underbrace{2,6,5,}_{}\overbrace{3,1,4,}^{}\underbrace{1,-1,2,-1,3,7,0,}_{}1,4,2$$

then your first algorithm should return 4, and your second algorithm should return 7.

3. There is no question 3!