Today: DP on trees
1. Binary search tree
2. Maximum independent set
3. CYK

Exam 2: 7-9 pm on Nov 5
- Covers material up to and including next two lectures
- Conflicts? LMK by Tue

bst_search(root, key)
Θ(log n)

if root is None:
    return None
if root.value == key:
    return root
else if key < root.value:
    return bst_search(root.left, key)
else:
    return bst_search(root.right, key)

Worst-case search time is \( h \) for balanced tree \( h = \Theta(\log n) \) where \( n \) is the number of elements in BST.

Worst-case search time is \( \Theta(n) \) for unbalanced tree.
bst - ave - cost ( root, level = 1 );
    if root is None:
        return 0
    else:
        return \[
            \frac{\text{level of element } i}{n} \leq h(i) \leq f(i) \times (i)
        \]

Sorted list of names [1, 3, 5, 6, 9, 13, 20, 55, 70, 99]
list of frequencies \( f_1, \ldots, f_9 \)
find BST of least average search cost
output cost of least cost BST
\[
    f_4 \cdot 1
\]
\[
    [1, 3, 5, 6, 9, 13, 20, 55, 70, 99]
\]
BST ( freq - list , level
if list is empty
    return 0
for \( i = 0 \) to len(freq-list)-1:
    make i root
    level+1
\[
\text{calculate } cur = \text{BST}(0, n, -1, \text{level} + 1) \\
\text{if } cur < \text{best}, \text{best} = cur \\
\text{return } \text{best}
\]

\[
\text{BST}[i, j, l] = \text{cost of subtree containing } \varepsilon_i \ldots \varepsilon_j \text{ at level } l
\]

\[
\text{BST}[i, j, l] = 0 \text{ if } i > j
\]

\[
\text{BST}[i, j, l] = \min_{r \in \{i, \ldots, j\}} \text{BST}[i, r-1, l+1] + \text{BST}[r+1, j, l+1]
\]

\[O(n^3)\]

for \(i = 0 \text{ to } n-1\)
for \(j = i+1 \text{ to } \max(n-1, i+(n-\text{level}+1))\)
for \(l = 0 \text{ to } \text{level}\)

\[
\text{BST}[i, j, l] = \min...
\]

\[
\text{BST}[i, j, l+1] = \text{BST}[i, j, l] + \sum_{k=1}^{e_l} f(\varepsilon_k), (\varepsilon'_k, (1, l2)) \leq f(\varepsilon_k) (e'_k, (1, l))
\]

\[
\text{BST}[i, j, l+1] = \max \left( \text{BST}[i, j, l], \sum_{k=1}^{e_l} f(\varepsilon_k) \right)
\]