Homework 10
EXTRA CREDIT 3
Due Friday May 1, 5 PM EST

Extra credit is worth one extra percent of your overall class grade. TAs will not answer Piazza questions or hours queries on any extra credit topics.

To submit your solutions, please make a directory ~/.course/cs0160/extraCredit3 containing your solution PDF and then run the script cs0160_handin extraCredit3.

1 Written Problems

Problem 10.1
Binary Search Tree Review

In a binary search tree, T, in-order traversal will visit the nodes in increasing-key order, thus providing a linear-time mechanism for extracting all keys in order. Suppose that you have some range of keys, say $k_1 \leq k \leq k_2$, and you’d like to extract all keys in T that lie within this range, and you’d like to get the results, as before, in increasing-key order. (Note that neither $k_1$ nor $k_2$ needs to actually be a key in the BST, and that the returned list of keys might well be empty!) Once again, inorder traversal with a test for in-range-ness will solve the problem in worst case $O(n)$ time, where $n$ is the number of keys in the tree. You are to describe how to solve the problem in time $O(h + s)$, where $h$ is the height of the tree $T$, and $s$ is the number of keys returned.

- Give a brief description of your approach to this problem.
- Write pseudocode for your algorithm.
- Explain briefly why your algorithm is $O(h + s)$.

Problem 10.2
Dictionary Segmentation

Given an input string and a dictionary of words, develop an efficient algorithm to determine if the input string can be segmented into a space-separated sequence of dictionary words. There is no limit to the number of times you may use each word in the dictionary.

For example, suppose we have the following dictionary of known words:

```
dictionary = {"i", "like", "ice", "cream", "icecream", "popsicle"}
```
Consider the following input/output pairs.

Input: "ilike"
Output: True
We can segment the string into "i like" using words from the dictionary.

Input: "ilikeicecream"
Output: True
We can represent the string as either "i like icecream" or "i like ice cream."

Input: "ieatpopsicles"
Output: False
We can’t segment the string into words contained in our dictionary.

Note that a greedy approach will not always guarantee the correct solution! For example, consider the following test case:

```
dictionary = {"a", "an", "ill", "pill", "pillow", "case"}
Input: "pillowcase"
Output: True
We can segment the string into "pillow case."
```

In this case, if we tried to find the first matching prefix, we would pick “pill.” However, that makes the rest of the string (“owcase”) unsegmentable given our dictionary.

Write pseudocode for your algorithm.