1 Radix Sort

In radix sort, we iteratively run bucket sort, using a different part of the key in each pass. In the first pass, we sort by the least important part of the key; in the last pass, we sort by the most important part.

1.1 Doing it by Hand

Use Radix sort to sort the following four-letter words. Show the contents of the array after each iteration of Bucket Sort.

Make sure to check your work - if you sort it properly, then the last version of the array will be completely sorted!

- alms
- best
- afar
- burp
- ball
- anon
- bags
- cool
- bell
- coal
- curl
- adds
- barn
- bolt
- case
- bark

1.2 Give an Example of Broken Sorting

Why doesn’t Radix Sort start with the most important part of the key first? Doesn’t that make more sense?

Come up with your own example of four-letter words (you only need two of them!), and then sort them using the first character in the first pass (that is, the reverse of normal Radix Sort). Show that the result is not a sorted array.

2 Breadth-First Reconstruction

With a simple (non-balanced) BST or an AVL tree, it is pretty simple to replicate any valid tree simply by adding nodes in layers (this is called a “breadth-first” traversal). For instance, to build the following AVL tree simply by adding the nodes in the following order: K,D,Q,B,E,N,T,A.

However, this simple algorithm doesn’t work for all types of binary trees.
2.1 Splay Trees

Explain why this doesn’t work for a Splay tree. Explain why it is a problem using words; then give an example of a tree where this method fails.

2.2 Red-Black Trees

Explain why this doesn’t work for a Red-Black tree. Explain why it is a problem using words; then give an example of a tree where this method fails. (Assume that you are using the bottom-up method for all insertions.)

3 Going Back in Time

Let’s look at trees, and work backwards in time. For each of the following trees, come up with a previous tree, such that inserting a single value would result in the tree I gave you (also identify the value that should be inserted). For red-black trees, assume that we use the bottom-up method for insertions.

Some problems may have multiple possible solutions; only give one. Others may have no solution. For partial credit, simply state that they cannot be solved; for full credit, explain why.
For example, consider the following red-black tree:

```
    K
   / \
  G   T
 / \ / \
D  I J  
```

One possible solution is to insert I into the following tree:

```
    K
   / \
  G   T
 / \ / \
D  I J  
```

Give a solution for each of the trees below.

3.1 Tree 1 (Splay)
3.2 Tree 2 (Red-Black)

3.3 Tree 3 (Splay)