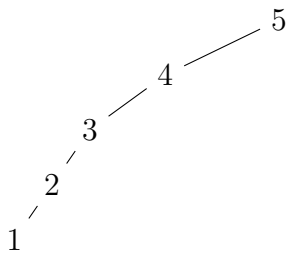
Today's topics: Splay trees; §§22.*.
Next class: polyphase merging; Reynolds's paper.

1. List the members of your group below. Underline your name.

2. In order to speed up access to frequently or recently accessed elements, the *rotate-to-root* method of maintaining a binary search tree (that is not explicitly balanced) requires each element that is *accessed* to be rotated with its parent, if any, until the accessed element is the root of the tree. Thus, if the same element is accessed again immediately it requires only one tree node to be visited. If a few other operations intervene, the previously accessed elements are still likely to be close to the root.

   Depict the transformations to the following binary search tree resulting from the above method applied in response to the access pattern $1, 2, 3, 4, 5$. Depict all rotations and highlight the tree after all rotations for each insertion have completed.



[additional space for answering the earlier question]

3. What is the total number of rotations in the transformation of Question 2?

   Generalize the example of that question to a sequence of accesses $1, 2, \ldots, n$ applied to an initial left-only tree with keys $n, n-1, \ldots, 1$ and express the number of rotations as a function of n.

4. Repeat Question 2 using the standard *bottom-up splay tree* operations of *zig*, *zig-zag*, and *zig-zig* instead of the rotate-to-root method of that question. Indicate the type of operation used in each step of the transformation.

[additional space for answering the earlier question]

5. Suppose we have a robot that operates on a physical stack of n widgets in a warehouse. The robot accepts a single parameter $k \in 1, 2, \ldots, n$ and responds by reversing the order of the topmost k items in the stack (by flipping them). Is it possible to sort an arbitrary stack of widgets in ascending order of dates of manufacture (assuming the dates are on labels visible to us)? If so, provide an algorithm; if not, explain why.