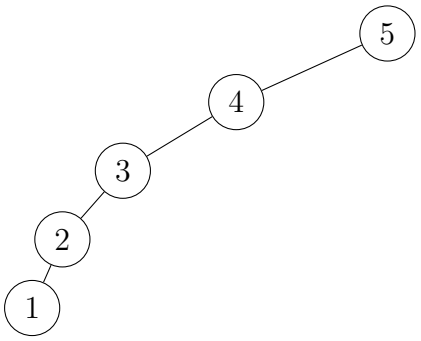
1. List the members of your group below:

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 $1, 2, 3, 4, 5$.



[additional space for answering the earlier question]

[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 $1, 2, \ldots, n$ 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*[1] instead of the rotate-to-root method of that question.

[1]Mark Allen Weiss, *Data Structures and Problem Solving Using Java*, 3rd edition (Addison-Wesley, 2006), §22.2.

[additional space for answering the earlier question]