1. (25 points) List the following 8 functions according to their growth rate:

\[(n/3)^{n^2}, \ (n \log n)^5, \ n!, \ n^{\log(n)}, \ (n/2)^{10}, \ n^{\log(\log(n))}, \ (n/2)^{n/3}, \ (\log n)^n\]

2. (25 points) We insert the following numbers in the given order into an empty AVL tree. For each rotation in an insertion, please display the tree before and after rotation. If no rotation happens during an insertion, simply draw the tree after the insertion:

\[1, 4, 9, 5, 6, 7.\]

3. (25 points) In-place heapsort is the only known in-place comparison-based sorting algorithm whose worst case time complexity is \(O(n \log n)\). Suppose the array \(A\) contains four numbers \(1, 4, 9, 5\), and the only operation to change the array \(A\) is by swap(i, j), which swaps the elements of \(A\) at positions \(i\) and \(j\). Please illustrate how the heap is constructed and changed by every swap operation (show the content of the array \(A\) after each swap operation).

4. (25 points) Given two sorted list \(A\) and \(B\), merge(A, B) will merge \(A\) and \(B\) into a single sorted list. The complexity of merge(A, B) is \(O(|A| + |B|)\), where \(|X|\) denotes the length of list \(X\). Please design an efficient algorithm (as fast as you can) which merges \(n\) sorted lists, each of length \(m\), into a single sorted list by calling merge(A, B). Please analyze the complexity of your algorithm in terms of \(n\) and \(m\).