EECS 233 Written Assignment #4

Due April, 28 11:59pm
21 points
NO LATE SUBMISSIONS

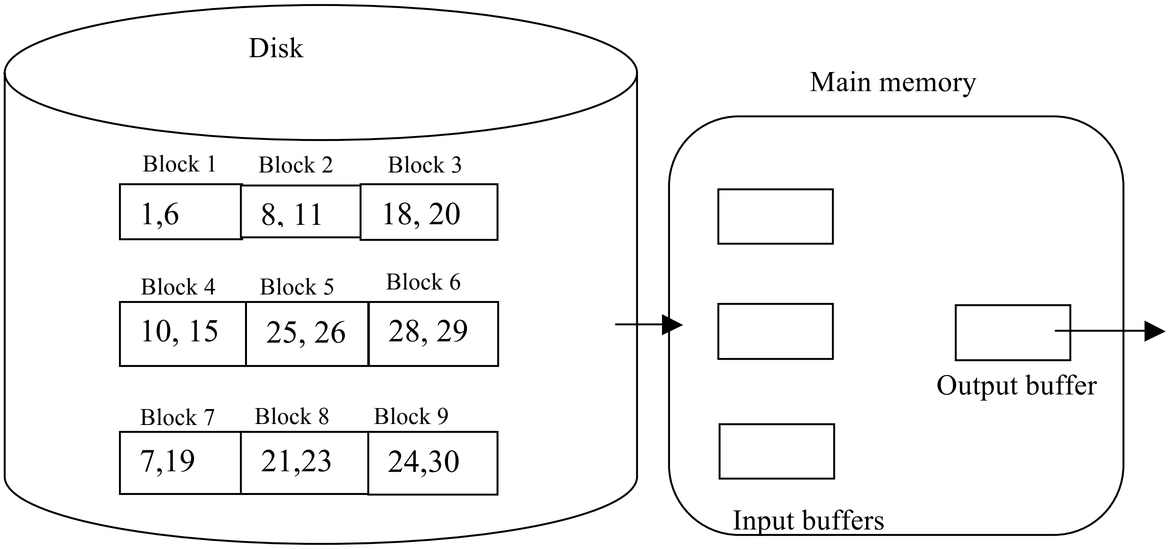
1. Consider a connected graph with N nodes.
   a) What is the smallest possible number of edges in a connected graph with N nodes? Prove that such a graph has to be acyclic. **(2 points)**
   b) What is the largest possible number of edges in a graph of N nodes? Prove that such a graph has to be complete. **(2 points)**

2. Consider QuickSort algoriothm, with the middle element of the input array always used as the pivot,
   (a) What is the asymptotic running time (expressed as Big-O) for sorted input **(2 points)**
   (b) What is the asymptotic running time (expressed as Big-O) for reverse-ordered input **(2 points)**
   (c) Which of the above arrays will be sorted faster? **(2 points)**

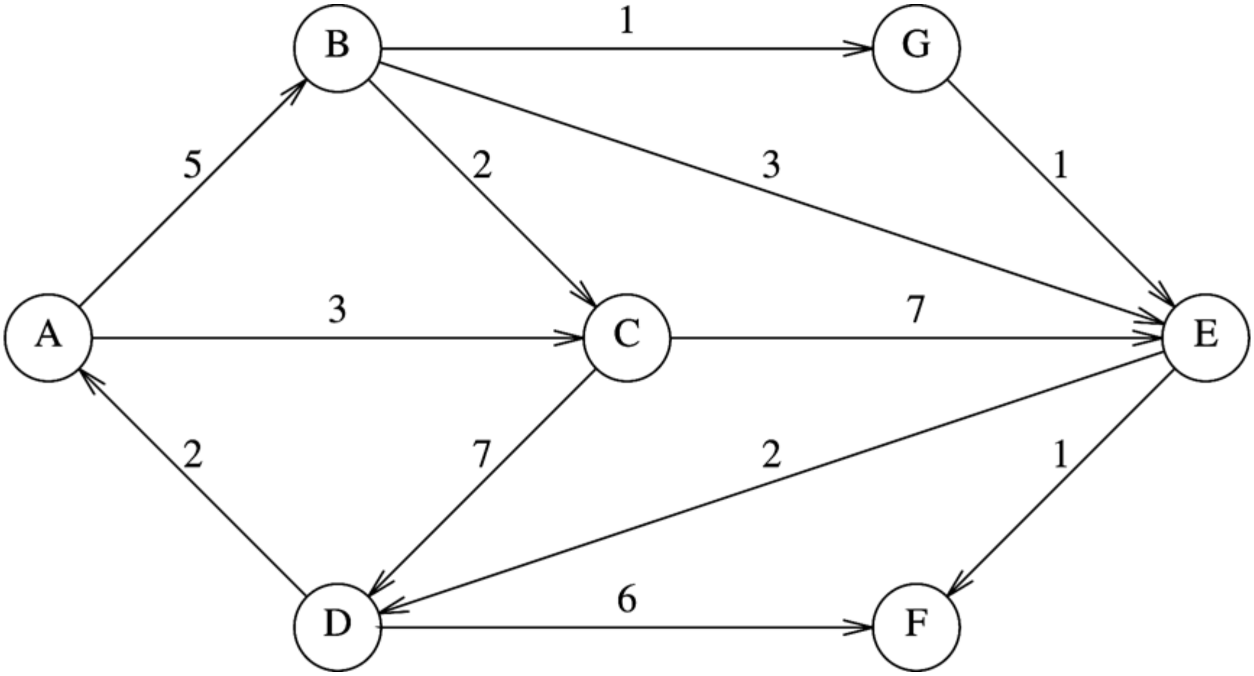
3. Suppose you have an array of N objects (as a reminder, in Java, the array would actually contain references to the objects). Each object has a key field and some other data fields. The key field is a string that can have only two distinct values, "red" and "blue". Give an O(N) algorithm to rearrange the array so that all "red" objects precede the "blue" objects. (Or rather that the references to "red" objects precede the references to "blue" objects.) You may uses only constant extra space. **(3 points)**

   Note: a non-solution would be to just count the number of reds and then rewrite the key field of the first k elements with the string "red" and the rest with "blue". This would change the nature of the objects. Imagine your objects are flowers, and one attribute is "color" and another is "species". Now if you just over-write the color attribute you will end up with corrupted data.

4. Consider a 3-way merge-sort with disk blocks holding only two numbers each. Assume that after the initial zero pass and the first pass, the result is shown in the figure below, with three input sorted runs of 3 blocks each. Please list the block numbers in the order in which they are read from the disk into input buffers during the second pass. **(2 points)**



5. Consider the following directed graph:



We can trivially generalize steps in Dijksra's and Prim's algorithms to a directed graph, by only considering neighbors of a newly added node that can be reached by following edges in the direction from the newly added node to the neighbors (you would get this automatically if the adjacency lists of a node only include outgoing edges). Find:

a. The Dijkstra's spanning tree from node A (using Dijkstra's algorithm). Show the order in which you add nodes into the tree as it is being built, and draw the resulting tree. **(2 points)**

b. The Prim's spanning tree from node A (using Prim's algorithm). Show the order in which you add nodes into the tree as it is being built, and draw the resulting tree. **(2 points)**

c. Does the Dijkstra's spanning tree represent a shortest-path spanning tree? Does the Prim's spanning tree of the previous step represent a minimum spanning tree? **(2 points)**