Linked List

The MyList class is a linked list of Person objects (String name, int age).

1. addLast(String xName, int xAge) – check if xName has the first letter ‘B’ or xAge < 17 then do nothing, otherwise add new person to the end of the list.
2. Delete the first node having age = 20.
3. Display the first 5 person having age > 22
4. Find the second max age. Display the first person having that age.
5. Sort the list descendingly by age.
6. Delete the last node having age = 20
7. Display the last 5 person having age > 22
8. Find the third max age.
9. add(String xName, int xAge, int index) – insert the new person at the given index
10. sort(int startIndex, int endIndex) – sort the linked list ascendingly by name from startIndex to endIndex

Binary Search Tree

Write a program in Java using Binary Search Tree data structure to manage information about persons. Variables used to store information about a person are:

- name - the name of a person (character String) , which is **the key of the tree**.
- age - the age of a person (integer value).

1. void insert(String xName, int xAge) - check if the first letter of xName is 'B' (i.e. xName.charAt(0) == 'B') then do nothing, otherwise insert new person with name=xName, age=xAge to the tree.
2. Save all elements having age < the average age of the tree in format (name, age) to the file "q2.txt" by post-order traverse.
3. Calculate the height of the tree.
4. Calculate the number of nodes of the tree.
5. Delete the root of the tree by copying.
6. Perform breadth-first traverse from the root and delete by copying the second node having age >= the average age.
7. Check if the root having non-empty left-son then rotate it to right about its left-son.
8. Perform pre-order traverse from the root, rotate the third node having non-empty right-son then rotate it to left about its right-son and display the tree to the output screen.
9. Calculate balance factor of all nodes. Display all node with balance factor by breadth-first traverse.
10. Check whether a given binary search tree is height balanced (AVL tree) or not.
11. Calculate level of all nodes. Display all node with level by breadth-first traverse.
12. Balance a binary search tree by simple balancing algorithm.

13. Perform pre-order traverse from the root, find the first node p having age >= 10, if node p has parent f and p is the right child of f then rotate node f to left about its right-son and display the tree to the output screen; otherwise, do nothing.
14. Perform pre-order traverse from the root, find the first node having age >= 10, if that node has parent f then delete by copying node f; otherwise, do nothing.

Graph:

1. Perform depth-first traversal (to the file f1.txt) from the vertex i = 1 but display 4 vertices from the 2nd vertex to the 5th vertex only.
2. Apply the Dijktra's shortest path algorithm to find (1) the shortest path from vertext 1 to vertex 7, then (2) from vertex 0 to vertex 6. Write 3 lines to the file f2.txt:
   - line 1 contains vertices in shortest path (1)
   - line 2 contains the last 4 vertices selected into the set S with their lables in (2)
   - line 3 contains vertices in shortest path (2)

Euleur/Hamilton: cho pseudocode, chúng ta cần viết lại và chạy từ đỉnh i nào đó.

MST: tương tự như Dijktra