Report for Programming Problem 2

Team: 2019216646_2019216792

Student ID: 2019216646 Name: António Pedro Correia

Student ID: 2019216792 Name: Mário Guilherme de A. Martins Sequeira Lemos

1. Algorithm description

-We start by storing the input in a map as explained in point 2.

-We then start the algorithm by recursively iterating through all the nodes until we reach one without any children (members that didn't recruit anyone); its then calculated the cost of choosing and not choosing the node in question and stored in the "output" map. When all the recruited members of a certain node have gone through this recursive step, we can calculate the cost of choosing all the recruited nodes or their "father" (this decision must be made because of the rules of the problem in question). By continuously storing all the "costs of choosing nodes" (number of nodes chosen and money paid by the recruited members chosen, we can continue to make this decision until the recursion ends.

-Lastly, we just need to analyze the contents of the *mapped value* corresponding to the first node in the tree and check which option is of greater value (the choose or the don't choose).

2. Data structures

-We have two main data structures. The first is a map where we store the input data; every *key value* represents a node and in the corresponding mapped value we store all the recruited members and respective amount of money paid. The second is also a map that as the key value representing a node but we store in the mapped value two arrays "choose" and "dontChoose" representing the cost of choosing and not choosing the corresponding node.

3. Correctness

We managed to achieve a score of 200 in Mooshak mainly because:

-Having a small usage of memory overall, using only two maps and two auxiliary vectors;

-After storing the input data, we only need to iterate through all the nodes once reducing the computation time.

Assumption: Choosing this node is the best solution.

Negation: Not choosing this node is the best solution.

Consequence: The best solution does not choose this node.

Contradiction: The best solution is not using the node in question.

4. Algorithm Analysis

-The space complexity of our algorithm is 6n - 1 -> O(n), 2n - 1 from the first map (input map) and 4N from the second one;

-Since we only iterate trough the tree once, our time complexity is O(n); we start on the root node and go trough all its children calling our method recursively. This causes the methos to be called n times.

5. References

-C++ Cheat Sheets: https://hackingcpp.com/cpp/cheat_sheets.html