1. [30 points] You’ve decided to accept a job as senior algorithm designer at the well-known Millisoft Corporation. One day, the President of Millisoft, Gill Bates, comes to you with the following problem. “I’m throwing a company party,” Gill says excitedly, “And I need your help! As you know, Millisoft has a hierarchical structure. You can think of it as a tree. The president, (that’s me!), is at the root of the tree.” You take a sip of your diet coke (which Millisoft provides for free - what a perk!) and listen patiently as Gill continues. “Below the root are supervisors, below them are managers, below them are team leaders, etc., etc., until you get down to the leaves - the summer interns. Anyhow, to make the party fun, I thought it best that we dont invite an employee along with their immediate boss (their parent in the tree).”

In other words, your task is to take as input a tree representing the company hierarchy. The tree need not be binary. Your objective is to find an algorithm that computes the largest number of employees (nodes) that can be selected such that no two adjacent nodes (i.e. a node and its child) are chosen. Design a greedy algorithm for this problem.

2. [30 points] The next day you cheerfully show Gill your algorithm, but he has also been busy while you were hard at work. Gill says: “I’ve personally assigned every employee a real number (actually its a double precision floating point, but nevermind that!) called their coefficient of fun. My objective is to invite employees so as to maximize the total sum of the coefficients of fun of all invited guests, while not inviting an employee with his immediate boss.

(a) Seeing the greedy algorithm you designed in part a, Gill says “That’s perfect - the weights won’t change a thing, we can just use this!”. Prove that Gill is wrong by giving a counterexample where your greedy algorithm does not find a set of maximum total fun.

(b) Gill then proposes the following “heaviest-first” greedy algorithm:

```
sort the employees in descending coefficient of fun
for i = 1 to n
    if employee i’s boss and subordinates have not been invited
       invite employee i
```

Show that this algorithm also fails to work.

(c) Gill sighs and decides this is just too hard. “Let’s just enumerate all possible subsets of employees, throw out those subsets that include an employee and his or her boss, find the score for each remaining subset, and finally choose the best one. To make it faster, we need only consider employees with positive fun coefficients!” There are 1000 employees at Millisoft with fun coefficients greater than zero. Millisoft has also just purchased a Cray XT5 Jaguar supercomputer that can process a quadrillion (10^{15}) subsets per second. Tell Gill when to plan on sending invitations (i.e. how long it will take to find the optimal solution using this brute force approach on the Crayfish).
3. [40 points] The two of you decide to sleep on it, and you wake up with a brilliant idea!

(a) Design an efficient dynamic programming algorithm for this problem. (Hint: Normally, we build a dynamic programming table or array. In this case, the structure in which you do your dynamic programming will be a tree! There’s nothing wrong with that!) 

(b) Show Gill how to modify your algorithm to find the optimal solution in which he gets invited to his own party.