Note: Please use your IDE to solve the following problem

Problem Statement

Assume that you have been given the task of writing a program to automate certain aspects of Human Resources for a large corporation. This company consists of teams of employees and each team has a manager. Managers in turns belong to higher level teams that have their own managers and so on. There are many layers of management across company. An entry level employee can have as many as ten levels of management above him, but this number is not fixed and can change in future. Managers have all the attributes of an employee, plus a few extra. Managers have a ***bonus*** which is added to their annual salary. Managers have an annual ***teamBudget*** that they can spend on anything they choose (from equipment purchase to parties, etc.).

Here are the use cases that your design needs to cover:

1) Employee class must implement a method called ***getManagers()*** which should return the list of managers in order (manager, manager of manager, etc.). The list should have at least one Manager in it (in case of the CEO of the company return an empty list!).
2) Manager class must implement a method called ***getEmployees()*** which should return the list of employees that work for this manager (including the employees of employees, etc.)
3) Manager class must implement a method called ***getTotalSalary()*** which should return a Double representing the total salary of all employees under this manager plus the manager himself/herself.
4) Manager class must implement a method called ***getTotalAnnualSalary()*** which should return a Double representing the total ***annual*** salary of all employees under this manager plus the manager himself/herself (remember that managers annual salary is ***12 * salary + bonus***).
5) Manager class must implement a method called ***getTotalAnnualBudget()*** which should return a Double representing the total ***annual*** salary of all employees under this manager plus the manager himself/herself plus ***teamBudget***.

As you can see in numbers 3 – 5 there is a high demand for operations to be added to Manager class over time. Do you know of a better way of solving the problem so that operations can be added from outside rather than modifying the Manager class?

In general, how do you make this more extensible so that in future any operation can be applied to all members of the aggregate?