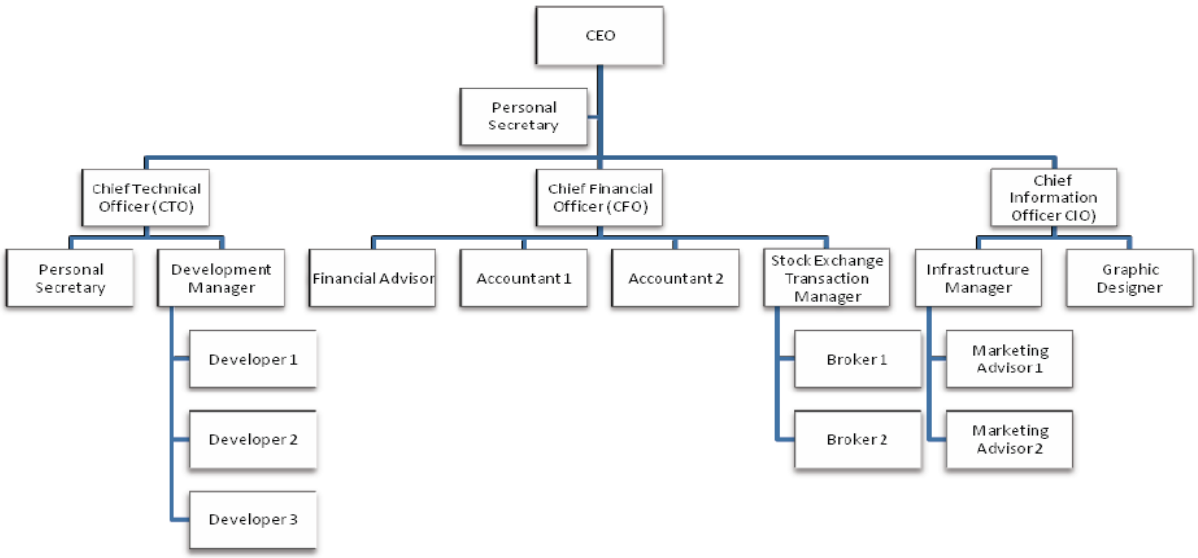
Practical Object oriented design (Composite, Visitor & Iterator)

Solve the following problems.

Problem1: Corporate Beaurocracy

A manager is an employee who leads engineers, technicians and support personnel, all of whom are also employees. A higher level manager can also lead other lower level managers. A typical hierarchy at the CyberCorp Inc will look similar to the one presented below:



a) Which design pattern you would use to implement our recursive corporate hierarchy? Provide the code that implements the above hierarchy and displays it as well.

b) Extend the solution from a) and show how the company average salary could be computed as well as the total cumulative income (the sum of all salaries) for all the employees at CyberCorp.

Problem2: Portfolio Manager

For this exercise you will implement a small portfolio application. A portfolio consists of accounts and other portfolios. An account is composed of securities classes: stocks, bonds, money market. The current value of a portfolio is the sum of the current values of the different securities it contains. You will need to write software to enable a user to build a portfolio of his choice and evaluate the total value of the portfolio. Do the following:

a) Write the interface for all components of a portfolio(both account and other portfolios). Operations to add a component, delete a component and return its current value, are part of the interface. You will also need an accept operation to allow the visitor to work.
b) Implement classes called Account and Portfolio which must implement the interface you created in part-a. Note that this may be recursive, in that a composite may contain both leaf objects as well as other composite objects.
c) You will also need a visitor called PricingVisitor which values the different securities and sums up their current value. The PricingVisitor defines operations (e.g. visitStock, visitBond,..) to evaluate each kind of component.
d) Implement an iterator called PortfolioIterator that escorts the visitor around the composite portfolio. The PortfolioIterator defines operations like hasNext(), next().
e) Finally you need to define a class called PortfolioManager that is a Singleton class. This would contain operations to initialize (build) a portfolio, and evaluate it. To evaluate the portfolio, the PortfolioManager would use the iterator to navigate the portfolio and the visitor to value each component. The structure and creation of the composite structure in memory can be hardcoded in the PortfolioManager. No client GUI is necessary.
f) Test your software by building a couple of portfolios and evaluating them.

Problem3: XML processing

Design an application that applies a generic function (process) to XML elements with specific attributes. The elements are part of an XML document. Consider that you have a parser available with the following partially given interface:

```
class XMLElement {
   ...
   XMLElements getChildren();
   XMLAttributes getAttributes();
}
class XMLParser {
   XMLElement parseXMLDocument(URL url);
}
```

For those of you who know too little about XML we give a short explanation here. An XML document has a tree structure. The XML document has one root element which contains all other elements. The elements can have any number of children and attributes (which are specified as pairs: attributeNameX=”attributeValueX”). An example of an XML document is given here:

```
<root>
   <child1 attributeName1=”attributeValue1” attributeName2=”attributeValue2”>
```

Practical Object oriented design (Composite, Visitor & Iterator)

```
        <child1_1 ...>
          ...
        </child1_1>
    </child1>
    <child2 ...>
        ...
    </child2>
       ...
    <childN ...>...</childN>
</root>
```

Answer the following questions:

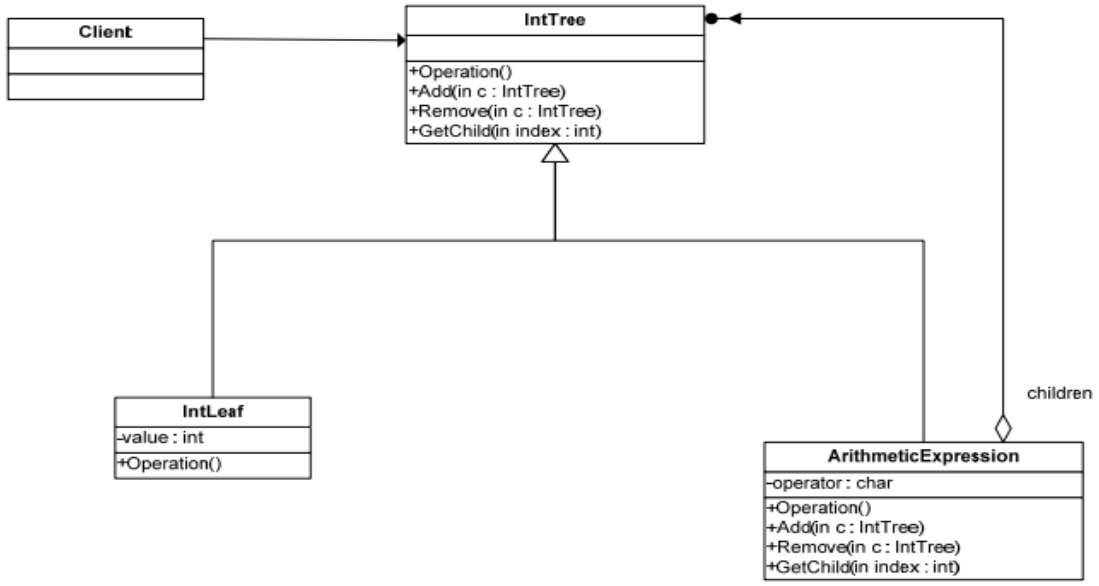
a) What design pattern could be used (or is already used) in the implementation of the XMLElement? Provide implementation

b) What changes should be done to the XMLElement class and how could the given classes be extended in order to apply a generic function to the XML element with specific attributes? What design pattern(s) could be applied here?

c) How would you extend/modify the given XMLElement in order to traverse the structure using visitors or iterators? Draw an UML sketch.

Problem4: Evaluation of Integer Trees

Consider the following UML diagram:



In the `ArithmeticExpression` the `operator` is of type `char` and can have one of the following values: `'+'` and `'-'`.

Practical Object oriented design (Composite, Visitor & Iterator)

Answer the following:

a) What pattern is represented by the diagram?

b) Change the class diagram to implement an interpreter over arithmetic expressions. Sketch the methods added to the diagram.

c) Changes the class diagram obtained in section (b) and add new classes (and methods to the old ones) to implement a visitor over the integer trees. Provide the required methods for visitor.

d) Implement the following types of visitors over arithmetic expressions:

- Print the arithmetic expression in infix form
- Print the arithmetic expression in postfix form