Alexander Pelaez
Ryan Pallman

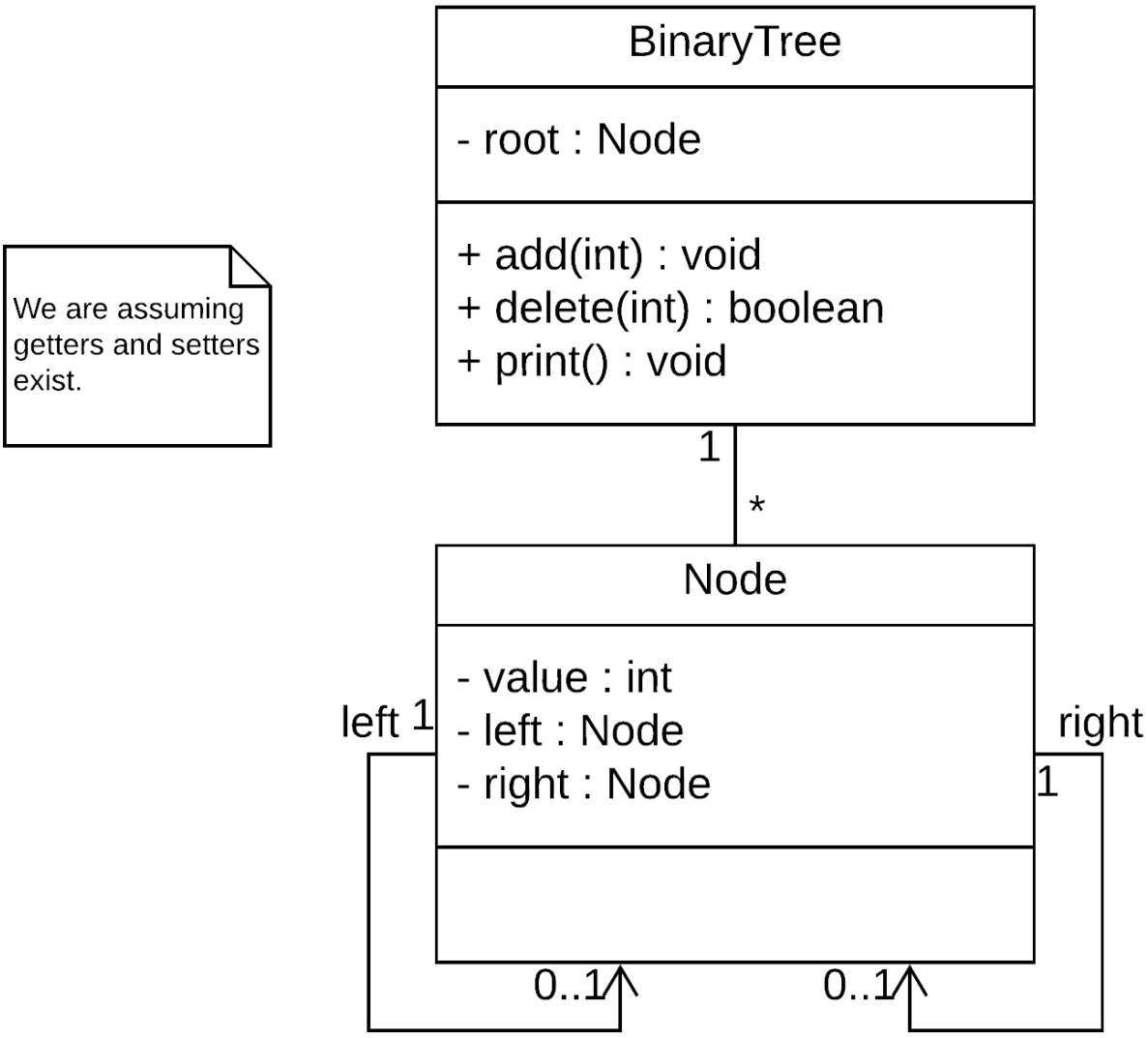
Exercise 2

A) $60 + 0.8(15) = 72$ man days $* \frac{32}{45} = 51.2$ story points

B) According to Prof Clem when deciding on a focus factor for a brand new team one should choose something around 70%.

C) Another way of estimating story points is going around the room and having everyone on the team say what point value they think the issue should be. We think that this way of estimating story points is worse than planning poker because team member's opinions can be influenced by what values other team members estimate.

D)



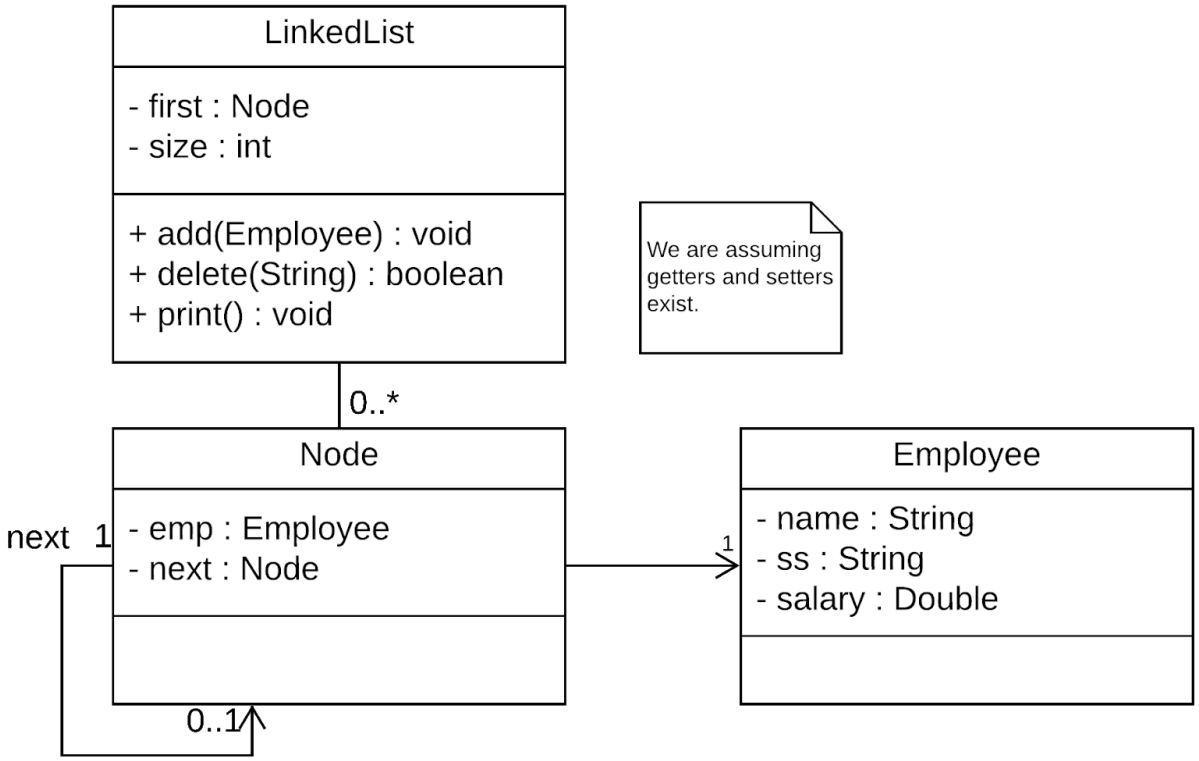
E)

```
import java.io.*;
import java.util.*;

public class BinaryTree
{
   private Node root;

   public BinaryTree()
   {
      root = null;
   }
```

```
    public void add(int value) {
        Node newNode = new Node(value);
        if(root == null) {
            root = newNode;
        } else {
            Node current = root;      // start at root
            Node parent;
            while (true) {           // exits internally
                parent = current;
                if (value < current.getValue()) {             // go left?
                    current = current.getLeft();
                    if (current == null) {             // if the end of the line
                        parent.setLeft(newNode);   // insert on left
                        return;
                    }
                } //end if go left
                else {                              // or go right?
                    current = current.getRight();
                    if (current == null) // if the end of the line
                    {                               // insert on right
                        parent.setRight(newNode);
                        return;
                    }
                }
            }
        }
    }

    public boolean delete(int value) {
        // Delete Node - did not implement since it seemed to be beyond the scope of the
assignment
        System.out.println("Deleted " + value);
        return true;
    }

    public void print() {
        // Print Tree - did not implement since it seemed to be beyond the scope of the assignment
        System.out.println("A tree.");
    }
}
```

```
public class Node
{
    private int value;
    private Node left, right;

    public Node(int value) {
        this.value = value;
        left = null;
        right = null;
    }

    public Node getLeft() {
        return this.left;
    }

    public Node getRight() {
        return this.right;
    }

    public void setLeft(Node node) {
        this.left = node;
    }

    public void setRight(Node node) {
        this.right = node;
    }

    public int getValue() {
        return this.value;
    }
}
```

F)



G)

```
public class Driver {

	public static void main(String[] args) {
		LinkedList list = new LinkedList();
		list.add(new Employee("alex", "123", 11.33));
		list.add(new Employee("Bob", "123", 11.33));
		list.add(new Employee("Jack", "126",15.33));
		list.print();

	}
}



public class LinkedList {
	private int size = 0;
	private Node first;
```

```
	public LinkedList() {//constructor
		first = null;
	}

	public void add(Employee emp) {
		Node temp = new Node(emp);
		if (first == null) {
			first = temp;
			first.setNext(null);
			size ++;
		}
		else {
			int tempSize = 0;
			Node iter = first;

			while (iter.getNext() != null)// while there is still data at iter
			{
				iter = iter.getNext();// get next node
				tempSize--;

			}
			iter.setNext(temp);
			temp.setNext(null);
			size ++;
		}
	}

	public boolean delete(String name) // delete method
	{	//Delete Node - did not implement since it seemed to be beyond the scope of the
assignment
		System.out.println("Deleted Employee "+ name);
		return true;
	}

	public void print() // print list out
	{
		Node iter = first;
		int temp = 0;
		while (iter != null)// while there is still data at iter
		{
			System.out.println(iter.getValue().getName());
			iter = iter.getNext();// get next node
			temp++;
```

```
                }
        }

}



public class Node {
        private Node next;//instance variables
        private Employee emp;

        public Node(Employee emp2) {//Constructor grab node value
                emp = emp2;
                next = null;
        }

        public void setNext(Node input) {//set next to input node
                next = input;
        }

        public Node getNext() {//get next node
                return next;

        }

        public Employee getValue() {//get the node value
                return emp;
        }
}



public class Employee {
        private String name;
        private String ss;
        private Double salary;

        public Employee(String n, String s, Double sal)
        {
                name = n;
                ss = s;
                salary = sal;
        }
```

```
	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getSs() {
		return ss;
	}

	public void setSs(String ss) {
		this.ss = ss;
	}

	public Double getSalary() {
		return salary;
	}

	public void setSalary(Double salary) {
		this.salary = salary;
	}


}
```