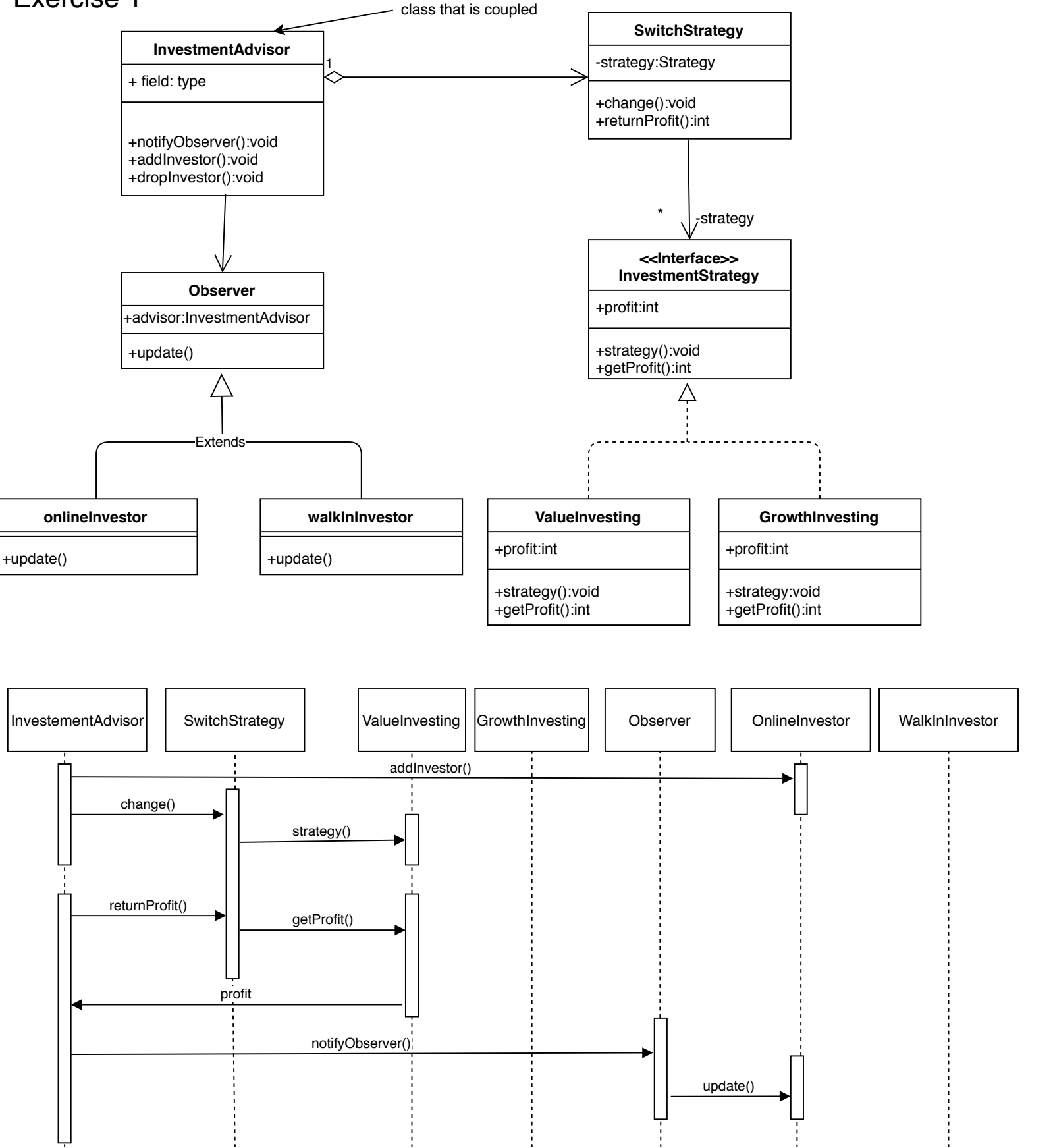


Exercise 1



## Exercise 2

- a.) 32 story points, 3 person team, working in sprints of 3 weeks = 45 total man days  
 Calculate estimated velocity for the next sprint based on another 3 week sprint,  
 but we have added 2 engineer's and 1 can only work 80% of the time.  

$$\text{Estimated velocity (based on last sprint)} = (\text{Total available man days}) (\text{Focus Factor})$$

$$\text{Focus factor of last sprint} = 32/45 = 71\%$$

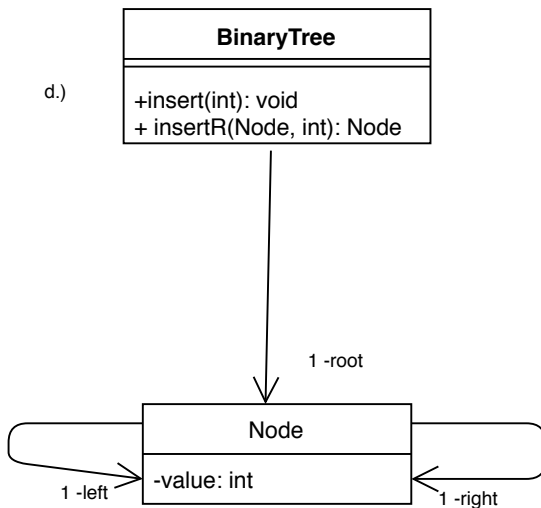
$$\text{Total man days} = (4 \text{ engineers} \times 15) + (\text{One engineer @ } 80\%) = 72 \text{ total man days}$$

$$\text{Estimated Velocity} = 72 \times .71 = 51.12 \text{ story points}$$

- b.) If it's a new team you would use the average of accepted story points over the past 3 to 5 iterations for velocity and taking the average over the development teams capacity.

- c.) The pseudo-fibonacci poker seems like a good idea but i think an easier and speedier Version would still discuss the reason why someone would have a large or small outlier but then just average out the numbers and if the numbers were to fall within the timeframe that "topic" would then fall into a priority category. Say our timeline was 3 weeks( 15 days x 3 Engineers) 45 days and our average from the poker was 30 days, that would assume the highest priority and anything lower would be in a lower priority.

d.)



```

e.)
public class BinaryTree {
    private Node root;

    public BinaryTree() {
        root = null;
    }

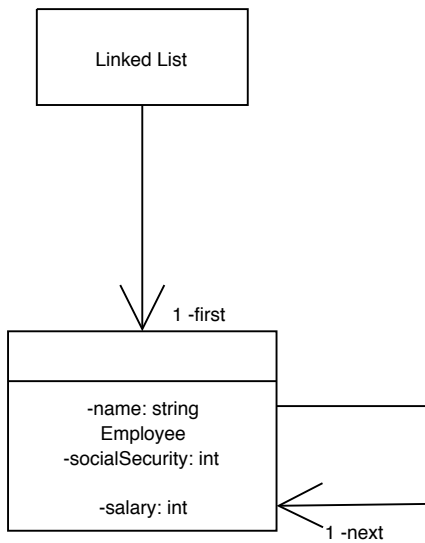
    public void insert(int val){
        insertR(root, val);
    }

    public Node insertR(Node newNode, int val) {
    }
}

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

    public Node(int v) {
        value = v;
        left = null;
        right = null;
    }
}
  
```

f.)



g.)

```

public class Employee {
    private String name;
    private int socialSecurity;
    private int salary;

    public Employee(String name, int socialSecurity, int salary){
        name = name;
        socialSecurity = socialSecurity;
        salary = salary;
    }

    public String getName() {
        return name;
    }

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

    public int getSocialSecurity() {
        return socialSecurity;
    }

    public void setSocialSecurity(int socialSecurity) {
        this. socialSecurity = socialSecurity;
    }

    public int getSalary() {
        return salary;
    }

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

import java.util.LinkedList;

public class Linked_list {
    private LinkedList<Employee> linked_list = new LinkedList<>();

    public void addTail(Employee newNode){
        linked_list.addLast(newNode);
    }
}
  
```