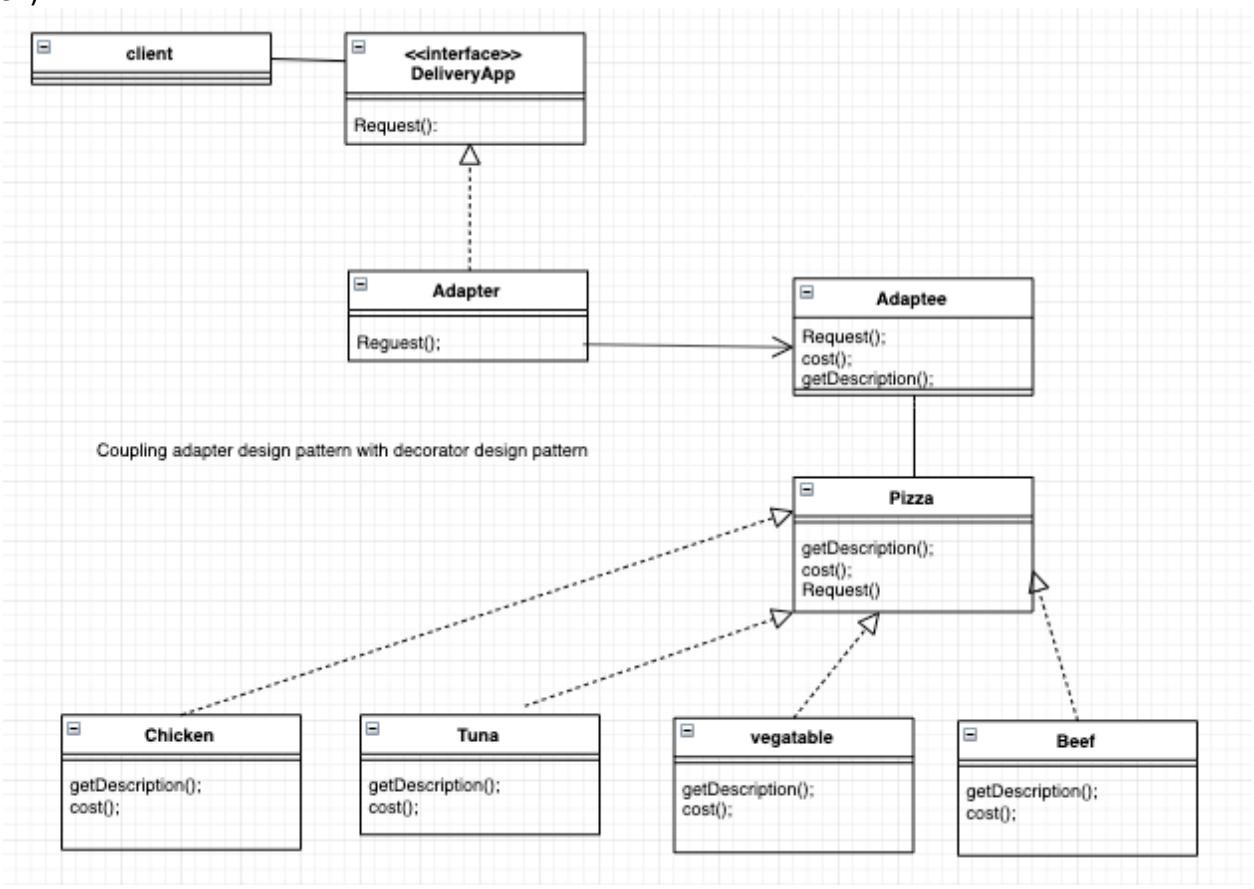
Abdulrahman Alhitmi
Ahmed Naji

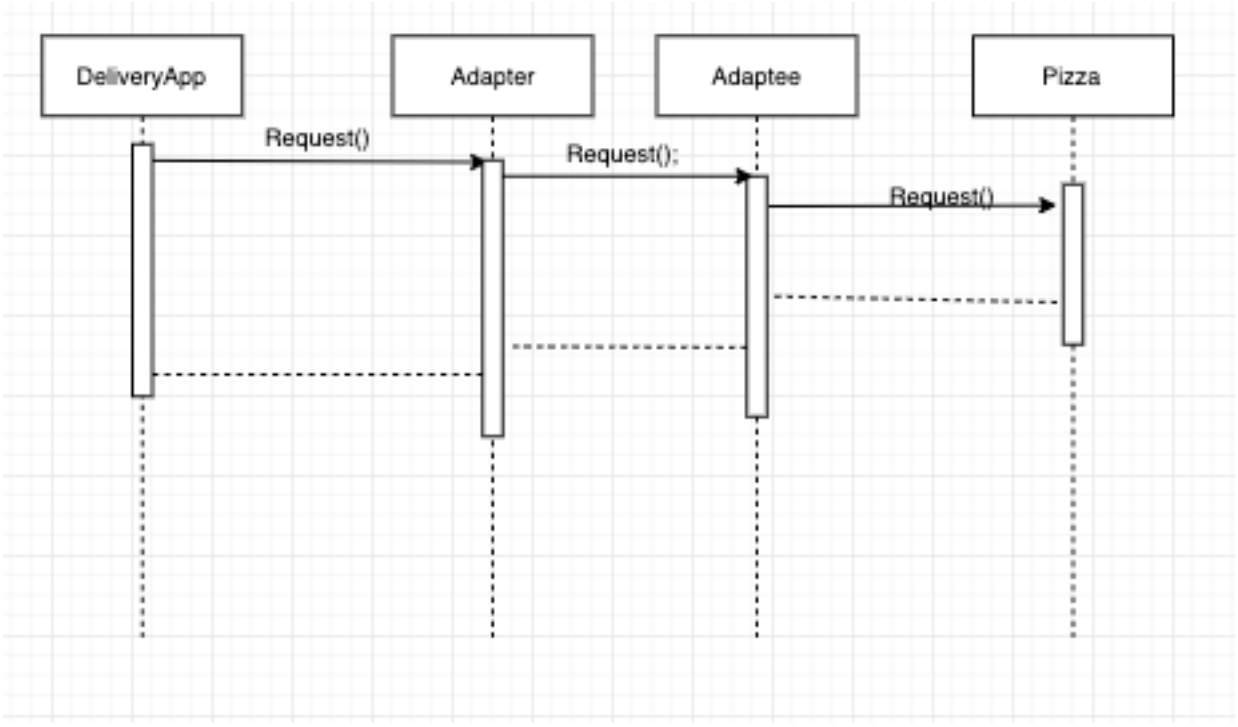
Homework #3

Exercise1)

a-



b-

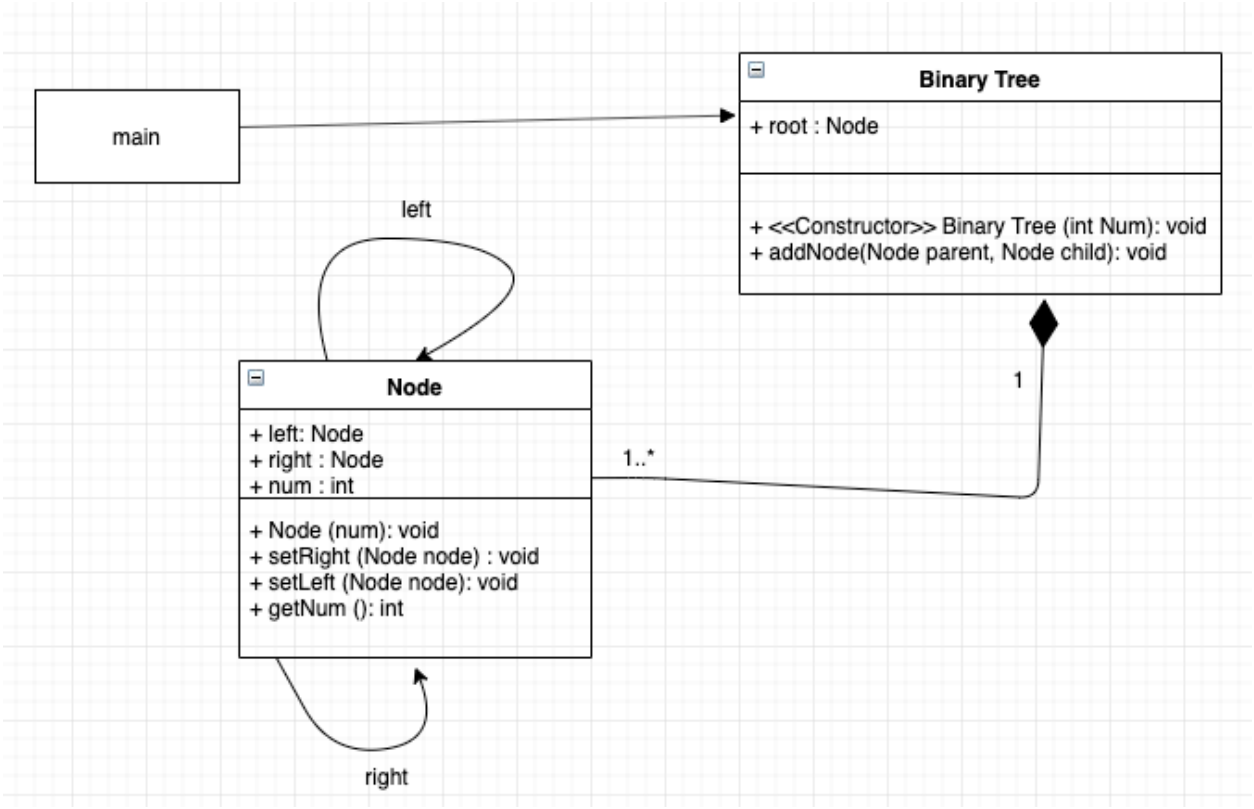


Exercise 2:

a- Last sprint: focus factor = actual velocity/ available man-days
Focus factor = 32/45 = 0.71.
This sprint: 5 person team but one is 80% working so, if the last one is 3 weeks with 3 people was 45 man-days, that means each had 15 man-days. So **4 person team is 60** man-days. The last person is **only working 80% so, (15*0.8 = 12) man-days**. We add that to the 60 man-days and we get **72 man-days total.**
So this sprint estimate is: **available man-days * focus factor = 0.71* 72 = 51.12 story points.**

b- We use the focus factor for the previous sprint and multiply it by the available man days in the new sprint.

c- Using past estimates. Many agile tools tracks story points, that can help the team understand and learn the difficulty and complexity of the current items. A good example would be bringing the team's past stories that were delivered, from there, the team compares each story to the current one in terms of similarity in effort. That could help the team determine precisely how complex the current story is and how much time it'll require. *(1)



d-

e- Code:

Node Class:

```
Public class Node{
        int num;
        Node left;
        Node right;

        Public Node(int num){
                Num = num;
                left = null;
                right = null;
        }
        Public void setRight(Node node){
                right = node;
        }
        Public void setLeft(Node node){
                left = node;
        }
        Public int getNum(){
                Return Num;
        }
}
```

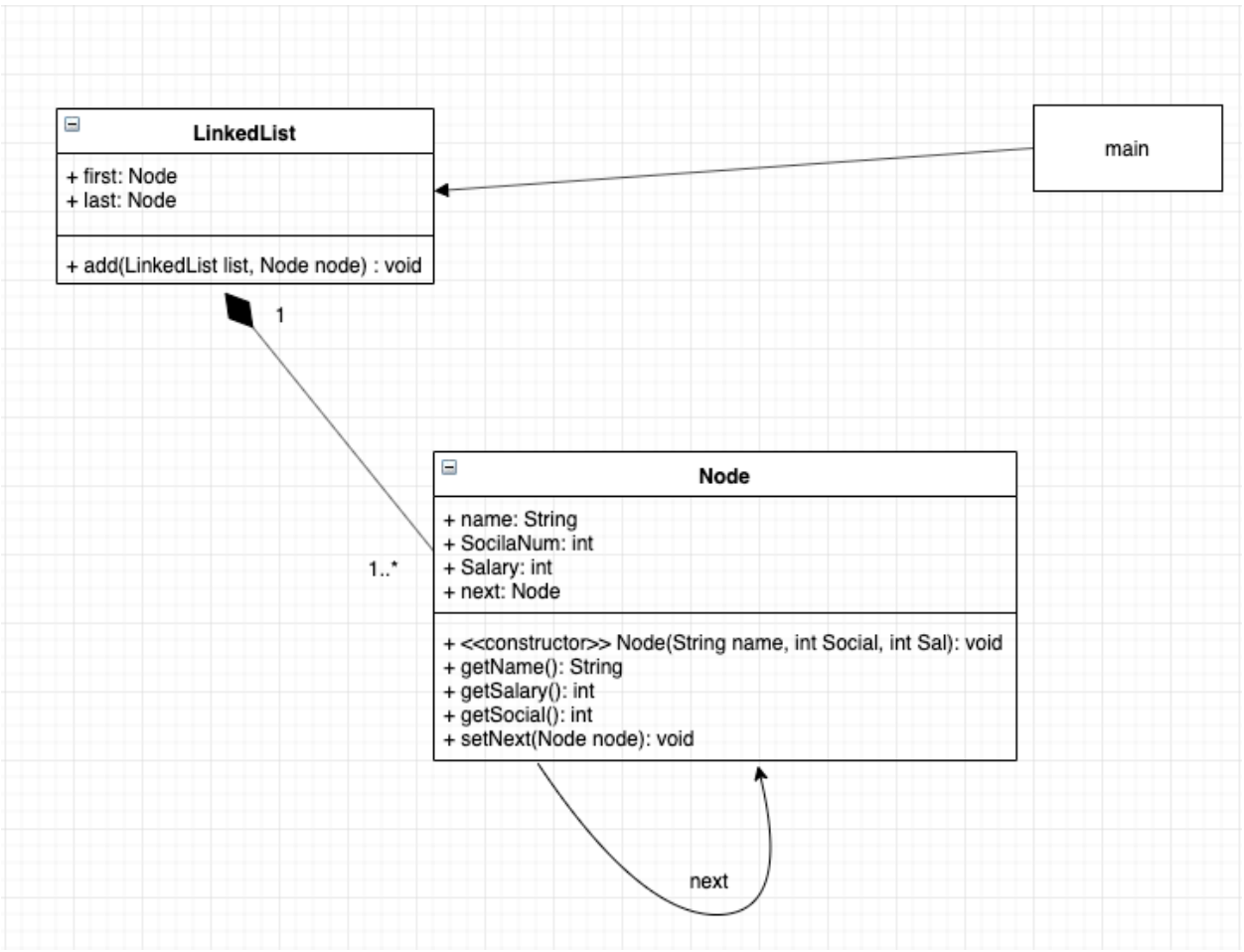
Binary Tree class:

```
Public class BinaryTree{
        Node root;

        Public BinaryTree(int num){
                Root = new Node(num);
        }
        Public void addNode(Node parent, Node child){
                If ((child.getNum%2)== 0){
                        parent.setRight(child);
                }
                else{
                        parent.setLeft(child);
                }
        }
}
```

Main class:

```
Public class main{
        Public static void main(String args[]){
                Node node1 = new Node(2);
                Node node2 = new Node(3);
                BinaryTree tree = new BinaryTree(1);
                tree.add(root, n1);
                tree.add(root,n2);
        }
}
```



f-

g- Code:

LinkedList class:

```
Public class linkedList{
        Node first;
        Node last;

        Public void add(linkedList list, Node node){
                If (first == null){
                        first = node;
                }
                Elseif (last == null){
                        last = node;
                }
                Else{
                        first.next = last;
                        last = node;
                }
        }
}
```

Node class:

```
Public class Node{
        String name;
        int SocialNum;
        int Salary;
        Node next;
        public Node(String name, int Social, int Sal){
                this.name = name;
                SocialNum = Social;
                Salary = Sal;
                next = null;
        }
        Public String getName(){
                Return name;
        }
        Public int getSalary(){
                Return Salary;
        }
        Public int getSocial(){
                Return SocialNum;
        }
        Public void setNext(Node node){
                next = node;
        }
}
```

Main class:

```
Public class main{
        Public static void main(String[]args){
                Node n1 = new Node(“Steve”, 999, 10);
                Node n2 = new Node(“April”, 100, 200);
                Node n3 = new Node(“Jeff”, 372, 32862);
                LinkedList list = new LinkedList();
                list.add(n1);
                list.add(n2);
                list.add(n3);
        }
}
```