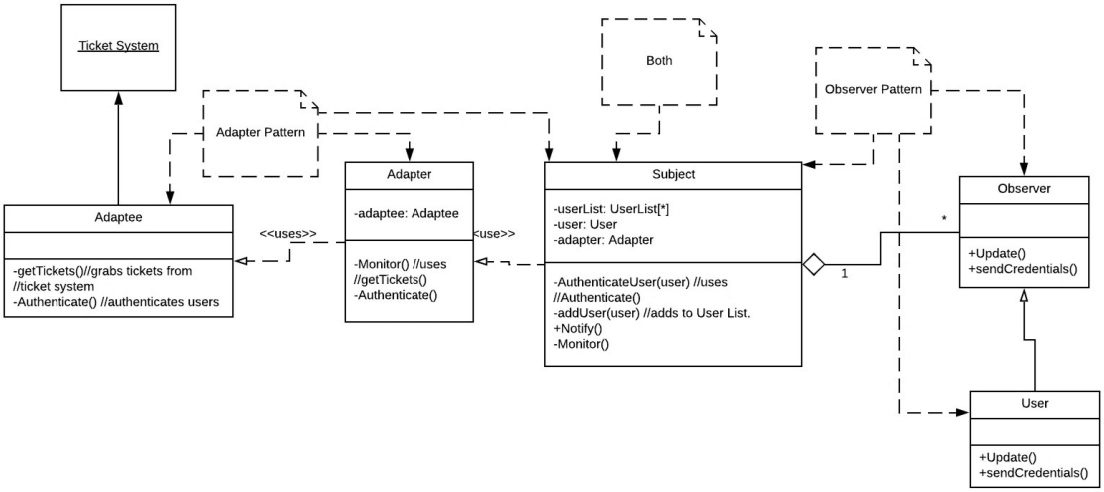
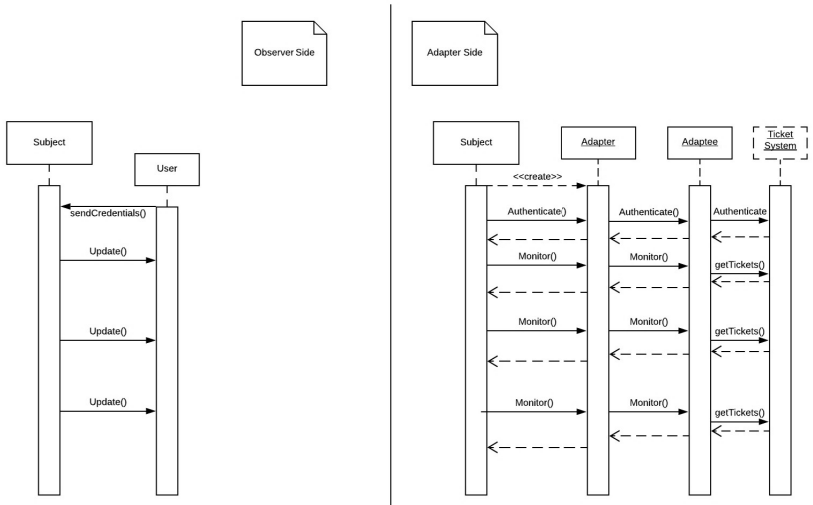

Jesse Arstein & Jay Van Alstyne
HW 3


Ticket System
Both
Observer Pattern
Adapter Pattern
Adapter
Subject
Observer
Adaptee
-adaptee: Adaptee
-userList: UserList[*]
-user: User
-adapter: Adapter
+Update()
+sendCredentials()
<<uses>>
<use>>
*
1
-getTickets()//grabs tickets from
//ticket system
-Authenticate() //authenticates users
-Monitor() //uses
//getTickets()
-Authenticate()
-AuthenticateUser(user) //uses
//Authenticate()
-addUser(user) //adds to User List.
+Notify()
-Monitor()
User
+Update()
+sendCredentials()


Observer Side
Adapter Side
Subject
User
sendCredentials()
Update()
Update()
Update()
Subject
Adapter
Adaptee
Ticket System
<<create>>
Authenticate()
Authenticate()
Authenticate
Monitor()
Monitor()
getTickets()
Monitor()
Monitor()
getTickets()
Monitor()
Monitor()
getTickets()

Jesse Arstein

Jay Van Alstyne

Exercise 2

a) A three person team has 15 man days a person. It took 45 man days to complete 32 story points so we can say 32/45 = .7 story points a man day. We can assume the two engineers will keep on pace with the current team so we can assume they also will complete .7 story points a man day. Since one engineer can only work 80% of the time we can only add 15 + (15*.8) man days = 27 man days. 27 + 45 = 72 man days at .7 story points a day is 50.4 story points. Round that down since we have to complete a story to gain story points to 50 story points in a 3 week sprint.

b) A brand new team is typically estimated at the standard 70% focus factor.

c) Have signs showing how many story points you think that feature would take. Have the group split up into how many story points they think it will take. Have the groups argue until everyone is united under one. This is probably worse since it allows for mid splitting discussion and can single people out mid process.

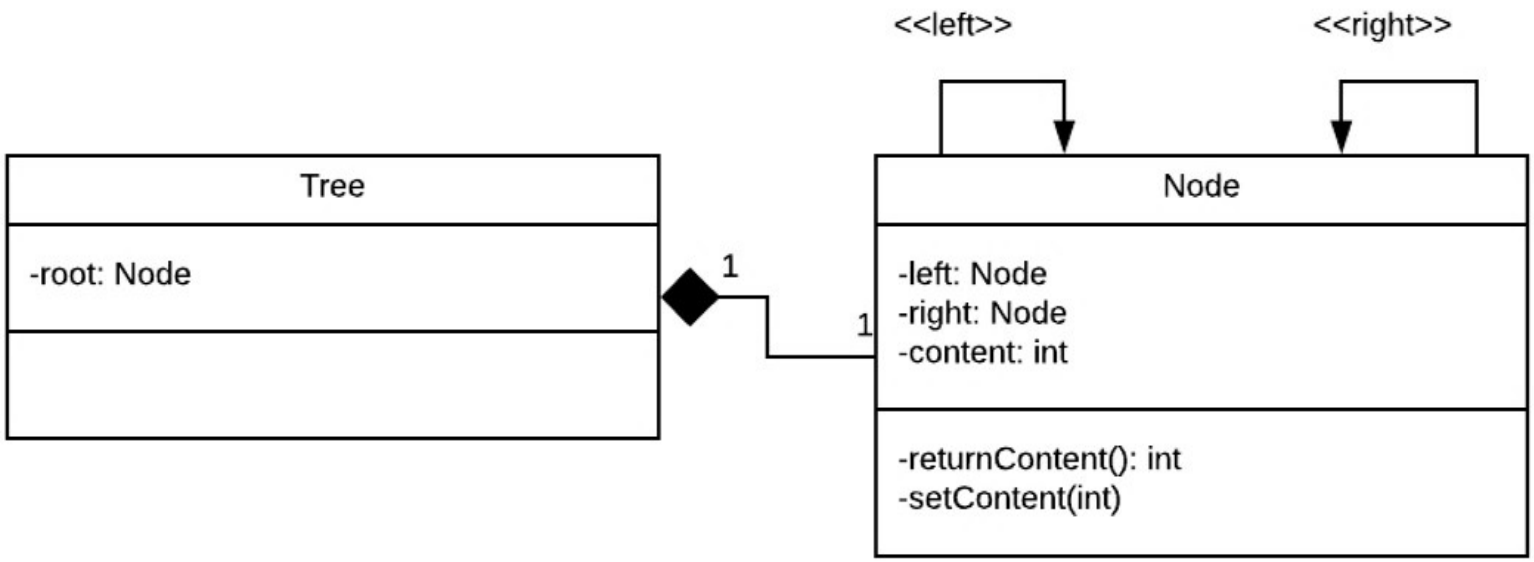
d) See attached

e) See attached

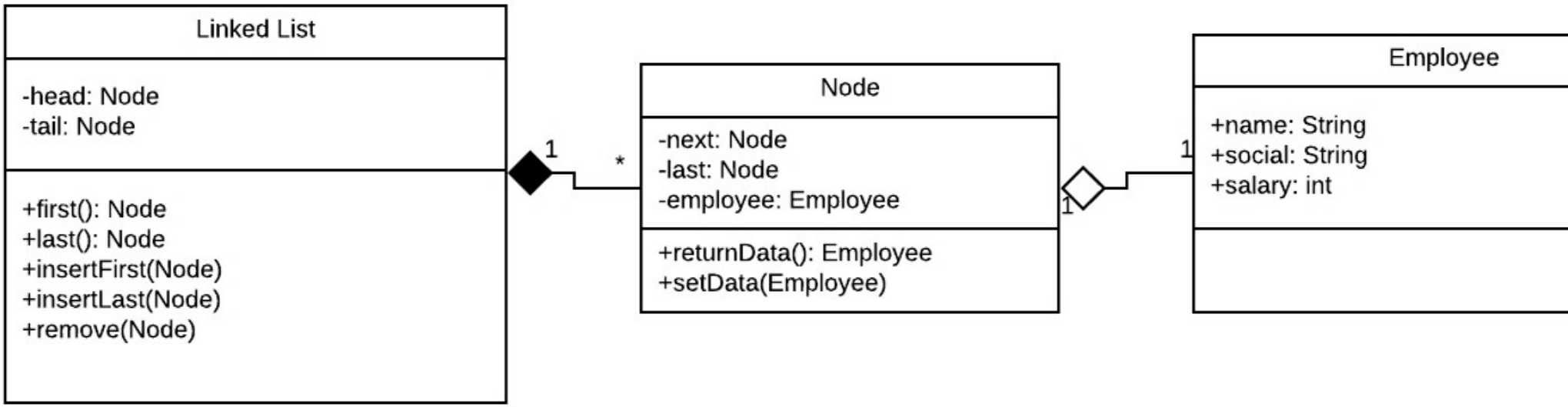
f) See attached

g) See attached

d)


<<left>>
<<right>>
Tree
-root: Node
1
1
Node
-left: Node
-right: Node
-content: int
-returnContent(): int
-setContent(int)

f)


Linked List
-head: Node
-tail: Node
+first(): Node
+last(): Node
+insertFirst(Node)
+insertLast(Node)
+remove(Node)
1
*
Node
-next: Node
-last: Node
-employee: Employee
+returnData(): Employee
+setData(Employee)
1
1
Employee
+name: String
+social: String
+salary: int

```java
public class Main {

    public static void main(String[] args) {
        Tree tree = new Tree();

    }

}
```

```java
public class Tree {
    Node root;
    public Tree() {
        root = new Node();
    }

}
```

```java
public class Node {

    int content;

    public Node() {
        Node left = new Node();
        Node right = new Node();
    }

    public int returnContent() {

        return content;
    }

    public void setContent(int in) {
        content = in;
    }
}
```

```java
public class Employee {
    String name;
    String social;
    int salary;

    public Employee() {
        name = "something";
        social = "124-52-1562";
        salary = 40000;
    }
}
```

```java
public class Main {

    public static void main(String[] args) {
        LinkedList list = new LinkedList();

    }

}
```

```java
public class Node {
    Node next;
    Node last;
    Employee employee;

    public Node() {
        next = new Node();
        last = new Node();
        employee = new Employee();
    }

    public Employee returnData() {
        return employee;
    }
    public void setData(Employee inEmployee) {

    }
}
```

```java
public class LinkedList {
    Node head;
    Node tail;
    public LinkedList() {
        head = new Node();
        tail = new Node();
    }
    public Node first() {
        Node someNode = new Node();
        return someNode;
    }
    public Node last() {
        Node someNode = new Node();
        return someNode;
    }
    public void insertFirst(Node inNode) {
    }
    public void insertLast(Node inNode) {
    }
    public void remove(Node inNode) {
    }
}
```