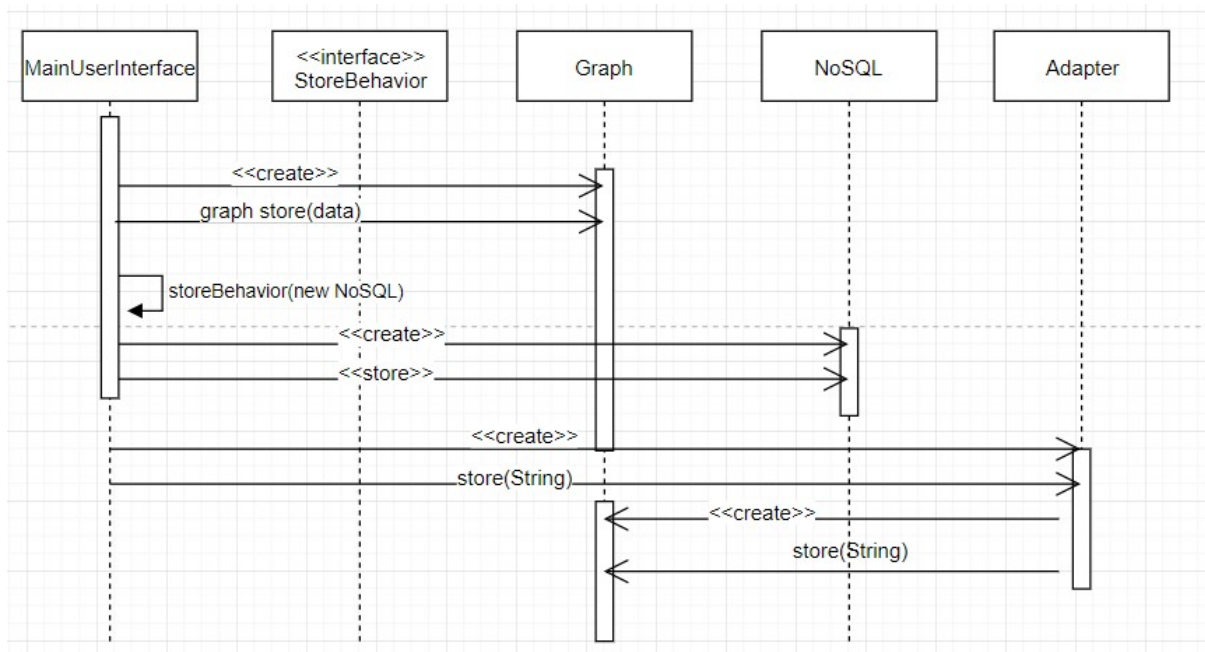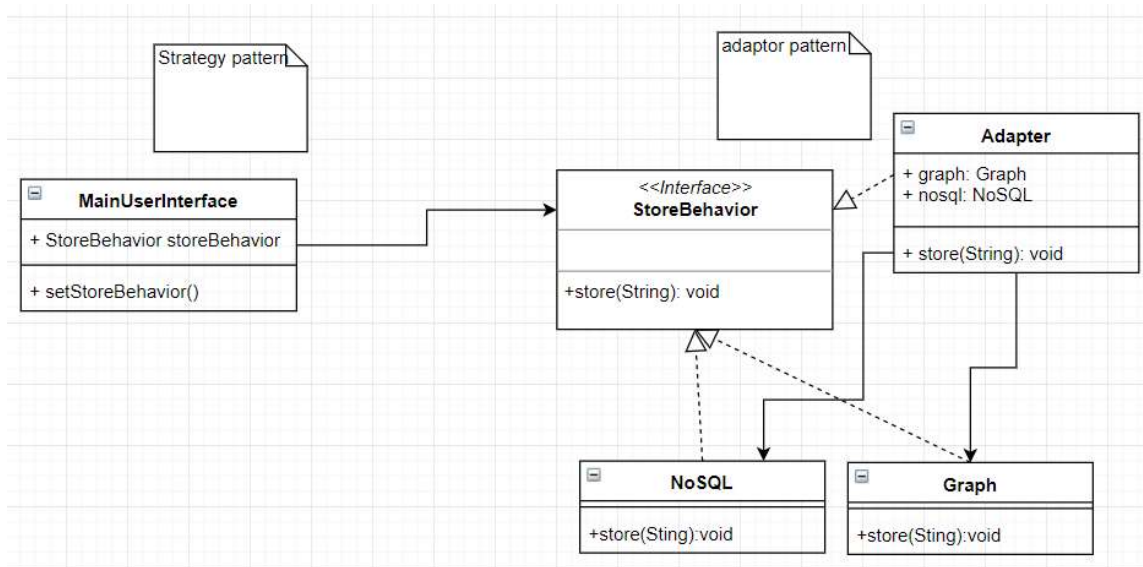Bruce Clark
9/24/2019
ESOF HW 3

Exercise 1

The class method of store in the Graph class could either be changed by the adaptor pattern or the strategy pattern:

Exercise 2

a) Answer: 50 completed story points in the 2nd sprint
- a. Sprint 1
  - i. 45 man days
  - ii. 32 completed story points
  - iii. 45 * focusfactor = 32
  - iv. Focusfactor = .71111
- b. Sprint 2
  - i. 45 man days * .71111 = 32 eng days
  - ii. (15 man days + (15 * .8)man days) * .7 = 18.9 eng days
  - iii. Estimated velocity = 32 + 18.9 = 50.9

b) You would have to pull a number out of your back pocket. Usually .7 is a good starting focus factor.

c) You could estimate story points by the number of story points by the number of words inside the story. This would not be as effective because it would lack the thoughtfulness of the engineers who would be developing from the story and would just be a simple calculation.
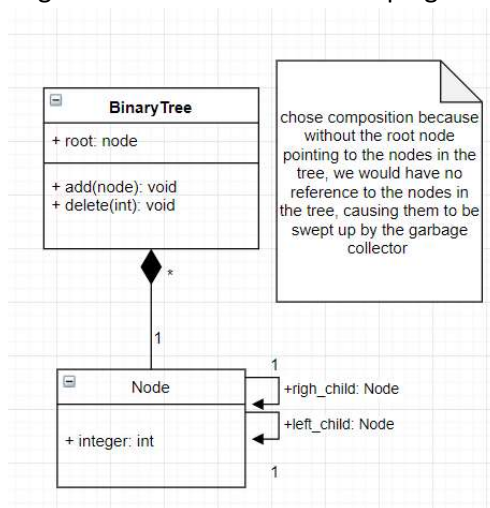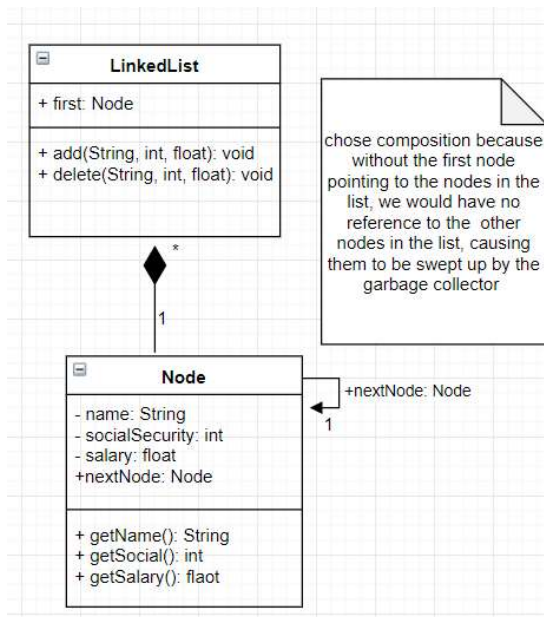
d)

e)

```
public class BinaryTree {
    Node root;

    public void add(int x){
        //I am assuming we do not need to specify how to add to a binary tree
    }

    public void delete(int x){
        //I am assuming we do not need to specify how to delete from a binary tree
    }

    public static void main(String[] args) {

    }
}
public class Node{
    Node right_child;
    Node left_child;
    int integer;

}
```

LinkedList

+ first: Node

+ add(String, int, float): void
+ delete(String, int, float): void

chose composition because without the first node pointing to the nodes in the list, we would have no reference to the other nodes in the list, causing them to be swept up by the garbage collector

*

1

Node

- name: String
- socialSecurity: int
- salary: float
+nextNode: Node

+ getName(): String
+ getSocial(): int
+ getSalary(): flaot

+nextNode: Node

1

f)

g)

```java
public class Node {
    private String name;
    private int socialSecurity;
    private  float salary;

    Node nextNode;

    public String getName(){
        return name;
    }
    public int getSocial(){
        return socialSecurity;
    }
    public float getSalary(){
        return salary;
    }
}
public class LinkedList {

    Node first;

    public void add(String name, int socialSecurity, float salary){
        //I am assuming we do not need to specify how to add to a linked list
    }
    public void delete(String name, int socialSecurity, float salary){
        //I am assuming we do not need to specify how to delete from a linked list
    }


    public static void main(String[] args){
    }
}
```