Data Structures - Quiz - 6

Question: Consider the following class:

```
public class sillyList extends DoublyLinkedList<Integer>{
    public boolean equals(sillyList<E> list2){
        if (size != list2.size()) return false;
        boolean equal_first_half = true;
        for (int i = 0; i < list2.size()/2; i++){
            equal_first_half = equal_first_half && (getAtIndex(i) == list2.getAtIndex(i));
        }
        Integer sum1 = 0;
        Integer sum2 = 0;
        for (int i = 0; i < list2.size(); i++){
            sum1 = sum1 + i * getAtIndex(i);
            sum2 = sum2 + i * list2.getAtIndex(i);
        }
        return equal_first_half && (sum1 == sum2);
    }
}
```

(a) (2 points) What does `equals` check? Write two `sillyList` that are equal according to this method but are not equal as `DoublyLinkedLists`.

(b) (8 points) Based on this implementation of `equals`, implement a sensible version of `hashCode()`.

Solution:

```
public int hashcode() {
int sum = 0;
int hash = 0;
int multiplier = 1;
for(int i = 0; i < size()/2; i++){
    hash += multiplier * getAtIndex(i);
    multiplier *= 512;
}
for(int i = 0; i < size(); i++){
    sum += i*getAtIndex(i);
}
return multiplier * sum + hash;
}
```