Problem 3

1. This solution achieves O(logN) time because it uses binary search to search the data structure. It divides it into halves and then keeps dividing it into halves until the middle element is the value we are searching for. This makes it O(logN).

Code for Linked List:

**int** binarySearch(Node a, **int** value){

getMiddle(a);

**if**(a.data == value){

**return** value;

}

**while**(a.data != value){

**if**(value < a.data){

getMiddle();

}

**if**(value > a[mid]){

low =mid +1;

}

mid = (low+high)/2;

}

**return** mid;

}

Node getMiddle(Node a){

Node one;

Node two;

one = first;

two = first.next;

**while**(two.next !=**null**){

two = two.next;

one = one.next;

}

**return** one;

}