1.

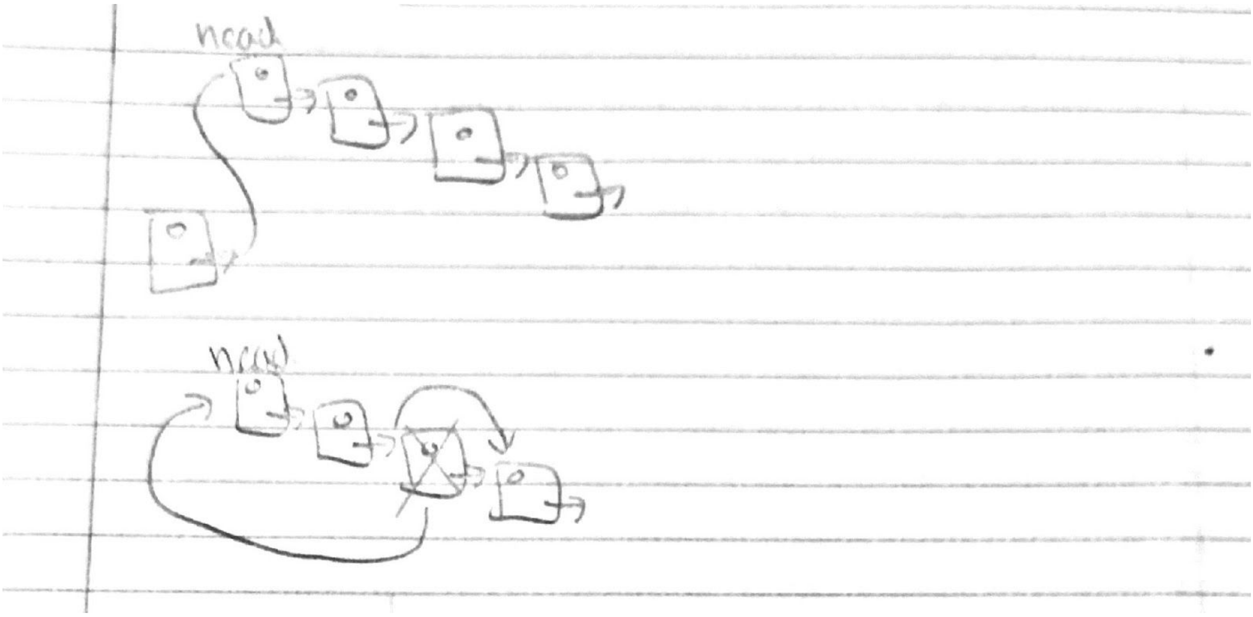
```
void bf_insert(Bloomfilter *bf, char *key){
        setbit(bf->filter, hash(primary(key)));
        setbit(bf->filter, hash(secondary(key)));
        setbit(bf->filter, hash(tertiary(key)));
}
```

2.

The more bits there are, the more bits the search function would have to look through the bloom filter. However, if we add more hash functions, that means that more bits of the same key will be set, so the chances of finding the set bit sooner is more probable.

Reading more into hash tables, insertion and search will take O(k) time.

1.



2.

```
ListNode *ll_node_create(GoodSpeak *gs){
        ListNode *n = (ListNode*)malloc(size of ListNode);
        If (n){
                n->gs = gs;
                n->next = NIL;
        }
        Return n;
}

Void ll_node_delete(ListNode *n){
    If (n){
        If (n->gs){
            free(n->gs);
            free(n);
        }
    }
}

Void ll_delete(ListNode *head){
    If (head){
        Listnode *p = head;
        While (p != NIL){
            Listnode *r = p;
            P = p->next;
            free(r->gs);
            free(r);
        }
    }
}

GoodSpeak *ll_node_gs(ListNode *n){
    if(n){
        Return n->gs;
    }
    Return null;
}

listNode *ll_insert(head, gs){
    Node = ll_node_create(gs);
    node ->next = *head;
    *head = node;
    Return node;
}
```

HashTable

Hash table is an array of heads of linked lists. Adding linked lists to the head will allow me to access each head with hashtable[index]. Adding a linked list to a head will be done by accessing the hashtable[index], and using that as the head to add to.

Linked List

Linked lists are linked up to the hash table. To add a linked list, we need to create a linked node with a given goodspeak structure and add it to a head, which will probably be the hashtable[index].

GoodSpeak

Goodspeak structure consists of oldspeak and newspeak. Forbidden words have no newspeak translation.