1. by a previous student

1.) Hash Table 11.2-1

$X_{ij} = I\{\text{keys hash to same value}\}$

$= P(\text{keys hash to same value}) = \frac{1}{m}$

$X = \#$ of collisions

$$X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$$

$$E[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] \quad \text{(linearity of expectation)}$$

$$= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{m}$$

$$= \frac{1}{m} \sum_{i=1}^{n-1} (n-i)$$

$$= \frac{1}{m} \left(\sum_{i=1}^{n-1} n - \sum_{i=1}^{n-1} i \right)$$

$$= \frac{1}{m} \left((n-1)n - \frac{(n-1)n}{2} \right)$$

$$= \frac{n(n-1)}{2m}$$

1.

2. By Ben Albert

2) (1) $\langle 3, 4, 2, 5, 1 \rangle$

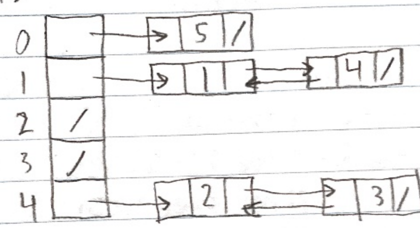
$h(3) = 4$

$h(4) = 1$

$h(2) = 4$

$h(5) = 0$

$h(1) = 1$



Load Factor $= \alpha = n/m = 5/5 = 1$

(2) $\langle 3, 4, 2, 5, 1 \rangle$ $h'(k, j) = (k^2 + j) \bmod M$

$h(3) = 4$
$h(4) = 1$
$h(2) = 4$ collision
$h'(2,1) = 0$
$h(5) = 0$ collision
$h'(5,1) = 1$ collision
$h'(5,2) = 2$
$h(1) = 1$ collision
$h'(1,1) = 2$ collision
$h'(1,2) = 3$

0	2
1	4
2	5
3	1
4	3

(3) $\langle 3, 4, 2, 5, 1 \rangle$ $h'(k, j) = (k^2 + c_1 j + c_2 j^2)$
$c_1 = 1, \ c_2 = 2$

$h(3) = 4$
$h(4) = 1$
$h(2) = 4$ collision
$h'(2,1) = 4 + 1 + 2 = 7 \bmod 5 = 2$
$h(5) = 0$
$h(1) = 1$ collision
$h'(1,1) = 1 + 1 + 2 = 4$ collision
$h'(1,2) = 1 + 2 + 8 = 11 \bmod 5 = 1$ collision
$h'(1,3) = 1 + 3 + 18 = 22 \bmod 5 = 2$ collision
$h'(1,4) = 1 + 4 + 32 = 37 \bmod 5 = 2$ collision
$h'(1,5) = 1 + 5 + 50 = 56 \bmod 5 = 1$ collision
$h'(1,6) = 1 + 6 + 72 = 79 \bmod 5 = 4$ collision
⋮

1 will never be inserted, gets stuck in loop

0	5
1	4
2	2
3	
4	3

1 never gets inserted

2(4)

The hash function can produce 6 different values: 0,1,3,4,5,9.
For any integer k, it can be written as $k = 11n + m$, $m = 0,1,2,...,10$.
Consider k^2, $k^2 = (11n)^2 + 22nm + m^2$. Obviously remainder can only come from m^2. Since $m = 0,1,2,...,10$, from m^2 mod 11, we can only get possible remainders 0,1,3,4,5,9.

3.

1)

```
CONTAIN-SUM(A, n, s)
    Create a hash table T of size s+1 with hash function
                                              h(k) = s - k
    for i = 1 to n
        if A[i] ≤ s
            CHAINED-HASH-INSERT(T, x)
    for i = 1 to n
        if A[i] ≤ s
            if CHAINED-HASH-SEARCH(T, s - A[i])
                return TRUE
    return FALSE
```

2). The running time of the algorithm is $\Theta(n)$, since the array must be traversed twice $(T(n) = \Theta(2n) = \Theta(n))$. All interactions with the hash table take constant time or time proportional to n, which does not affect the running time of $\Theta(n)$.

Space requirement: The hash table will have size $s+1$, and linked lists are only required to have a size of 1.