10-1

	Unsorted single linked	sorted single linked	unsorted double linked	sorted double linked
Search(L,K)	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$
Insert(L,X)	$\Theta(1)$	$\Theta(\|L\|)$	$\Theta(1)$	$\Theta(\|L\|)$
Delete(L,X)	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$
Successor(L,X)	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$
Predecessor(L,X)	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$
Minimum(L)	$\Theta(\|L\|)$	$\Theta(1)$	$\Theta(\|L\|)$	$\Theta(1)$
Maximum(L)	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$	$\Theta(\|L\|)$

* Assuming the linked list accepts duplicated elements.

12.2-2

```
Tree-Minimum(X)
    if left[X] = NIL
        return X
    else return Tree-Minimum(left[X])
```

```
Tree-Maximum(X)
    if right[X] = NIL
        return X
    else return Tree-Maximum(right[X])
```

12.2-3

```
Tree-Predecessor(X)
    if left[X] ≠ NIL
        then return Tree-Maximum(left[X])
    Y ← p[X]
    while Y ≠ NIL and X = left[Y]
        do X ← Y
           Y ← p[Y]
    return Y
```

2. (a) Open addressing with linear probing:

K	$h(k) = k^2 \text{ mod } m$	$h(k, i) = (h'(k) + i) \text{ mod } m$	Final Position
3	3^2 mod 5 = 4	(4 + 0) mod 5 = 4	4
4	4^2 mod 5 = 1	(1 + 0) mod 5 = 1	1
2	2^2 mod 5 = 4	(4 + 0) mod 5 = 4, (4 + 1) mod 5 = 0	0
5	5^2 mod 5 = 0	(0 + 0) mod 5 = 0, (0 + 1) mod 5 = 1, (0 + 2) mod 5 = 2	2
1	1^2 mod 5 = 1	(1 + 0) mod 5 = 1, (1 + 1) mod 5 = 2, (1 + 2) mod 5 = 3	3

(b) Open addressing with quadratic probing, where $c_1 = 1$ and $c_2 = 2$:

K	$h'(k) = k^2 \text{ mod } m$	$h(k, i) = (h'(k) + i * c_1 + i^2 * c_2) \text{ mod } m$	Final Position
3	3^2 mod 5 = 4	$(4 + 0 * 1 + 0^2 * 2)$ mod 5 = 4	4
4	4^2 mod 5 = 1	$(1 + 0 * 1 + 0^2 * 2)$ mod 5 = 1	1
2	2^2 mod 5 = 4	$(4 + 0 * 1 + 0^2 * 2)$ mod 5 = 4, $(4 + 1 * 1 + 1^2 * 2)$ mod 5 = 2	2
5	5^2 mod 5 = 0	$(0 + 0 * 1 + 0^2 * 2)$ mod 5 = 0	0
1	1^2 mod 5 = 1	$(1 + 0 * 1 + 0^2 * 2)$ mod 5 = 1, (1 + 1*1 + 1*2) mod 5 = 4, (1 + 2*1 + 4*2) mod 5 = 1, (1 + 3*1 + 9*2) mod 5 = 2, (1 + 4*1 + 16*2) mod 5 = 2. No place for 1 in the hash table.	

(c)

K	$h(k) = k^2$ mod m, m = 5	Final Position
3	3^2 mod 5 = 4	4
4	4^2 mod 5 = 1	1
2	2^2 mod 5 = 4	4
5	5^2 mod 5 = 0	0
1	1^2 mod 5 = 1	1

The load factor is $\alpha = n/m = 5/5 = 1$.

(d)

K	0	1	2	3	4	5	6	7	8	9	10
$h(k)=k^2$ mod 11	0	1	4	9	5	3	3	5	9	4	1

For k≥11, we can represent k = 11*a + b, where a is the quotient of k divided by 11 and b is the remainder, 0≤b≤10. Then we have:

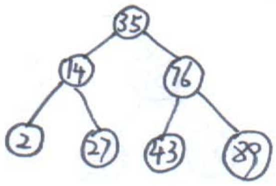
$$h(k) = k^2 mod\ 11 = (11a + b)^2\ mod\ 11 = (11^2 a^2 + b^2 + 2 * 11 * ab) mod\ 11$$
$$= b^2 mod\ 11, 0 \leq b \leq 10$$

This goes back to the above table. Thus, for any k≥0, all the possible values of $h(k) = k^2 mod\ 11$ are: 0, 1, 3, 4, 5, and 9

Solutions for HW#9.

1\. a) minimum height = 2.

k = <35, 14, 76, 2, 27, 43, 89>



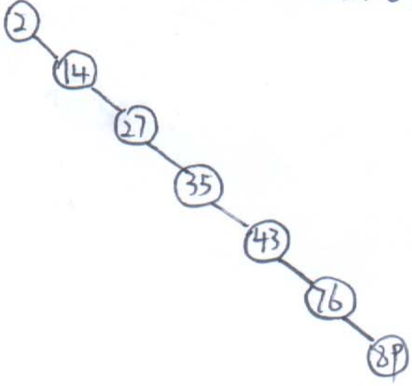
OR

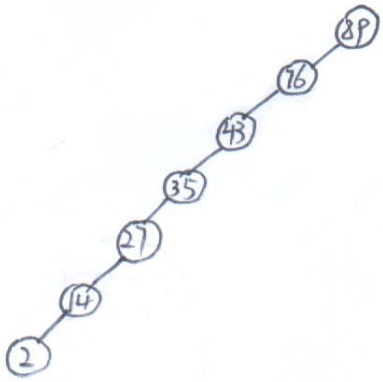
k = <35, 14, 2, 27, 76, 43, 89>

There are also other insertion orders for the keys in k that can generate the same Binary Search Tree of minimum height above.

(b) maximum height = 6.

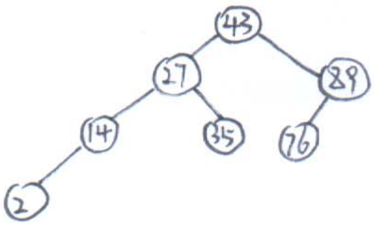
k = <2, 14, 27, 35, 43, 76, 89> OR k = <89, 76, 43, 35, 27, 14, 2>





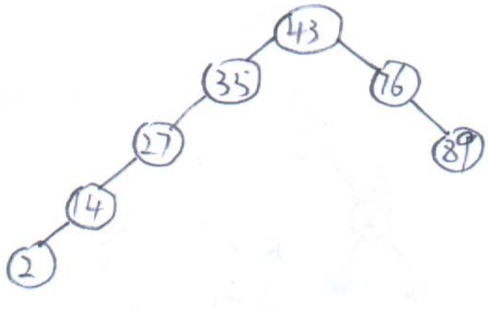
(c) height = 3

k = <43, 27, 14, 2, 35, 89, 76>



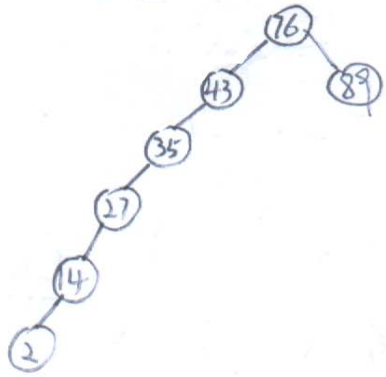
height = 4

k = <43, 35, 27, 14, 2, 76, 89>



height = 5

k = <76, 43, 35, 27, 14, 2, 89>



Note: There are also other solutions.