B10815057 Algorithm Homework 3

3.1

Linear probing:

22	88			15	4	17	28	59	31	10

Quadratic probing c1 = 1　c2 = 3:

22		31	88	15	59	17		4	28	10

double hashing with $h2(k) = 1 + (k \bmod (m - 1))$

22		31	59	28	15	17	88		4	10

3.2

Best-case Time : T(n) = Θ(1) 第一次 hash 後就找到，因此

Worst-case Time : T(n) = Θ(n) 經過 hash n 次後才找到，舉例: hash table 長度 11，每次輸入的數字都是 11 的倍數，且使用 linear probing，因此碰撞不斷的堆積，此時如果要搜尋最新輸入的數字時，就需要搜尋 n 次才可找到，不管哪種 probing 都會有此極端情形發生 Worst-case

3.3

因為輸入的數字是被限制住的(U 集合)，因此碰撞的機率非常高，而 hash table 又儲存了大量的資料，所以平均來說，每插入一個資料(在 U 集合內)，就需要經過$\frac{n}{|U|(U\text{集合的數量})}$次碰撞，與理想的 hash 時間複雜度:O(1)相差甚遠，因此該 hashing 不 useful。

3.4

順序不同就可能會造成碰撞的資料或時機不同，因此 hash table 在不同輸入順序的狀況下結果也是不相同的

3.5

將刪除後的位置設為 DELETED。

```
HASH-DELETE(T,k)
    I ← 0
    repeat j ← h(k,i)
        if T[j] == k
            tmp ← T[j]
            T[j] ← DELETED
            return tmp
        i ← i+1
    until T[j] == NIL or i == m
    return NIL
```

插入時，若遇到 DELETED 或空的位置，就可將值插入取代，而不需要再繼續做 hash。

```
HASH-INSERT(T,k)
    I ← 0
    repeat j ← h(k,i)
        if T[j] == NIL or T[j] == DELETED
            T[j] ← k
            return j
        i ← i+1
    until i == m
    error  “hash table overflow”
    return NIL
```