Problem 3.1. (5 pts) Solve the following recurrence equations by the master theorem or any other methods that we discussed in class.

(a) $T(n) = 32T(n/4) + n^2\sqrt{n}$
(b) $T(n) = 3T(n/9) + n^4 \lg n$
(c) $T(n) = 8T(n/4) + n\sqrt{n}$
(d) $T(n) = T(n-1) + \lg n$
(e) $T(n) = 2T(n/2 - 1)$

(a) $a=32,\ b=4$

$\log_{2^2} 2^5 \quad 5\log_{2^2} 2^1 = \frac{5}{2}$

$$\frac{n^2 \cdot n^{\frac{1}{2}}}{n^{\log_4 32}} = \frac{n^{\frac{5}{2}-\log_4 32}}{n} = \frac{n^0}{n} = \frac{1}{n}$$

Case 1 $= \Theta(n^{\log_4 32})$

(b) $a=3,\ b=9$

$$\frac{n^4 \lg n}{n^{\log_9 3}} = \frac{n^4 \cdot \lg n}{n^{\frac{1}{2}}} \qquad \text{case 3.}$$

$$3f\left(\frac{n}{9}\right) \le c \cdot f(n)$$

$$\Rightarrow 3\left(\frac{n}{9}\right)^4 \lg\frac{n}{9} \le c \cdot n^4 \lg n$$

$$\Rightarrow 3\left(\frac{n^4}{9^4}\right)(\lg n - \lg 9) \le c\, n^4 \lg n$$

$$\Rightarrow \left(\frac{3}{9^4}\right)(n^4 \lg n - n^4 \lg 9) \le c \cdot n^4 \lg n$$

$$\Rightarrow \frac{\left(\frac{3}{9^4}\right)(n^4 \lg n - n^4 \lg 9)}{n^4 \lg n} \le c, \quad T(n) = \Theta(f(n)) = \Theta(n^4 \lg n)$$

(c) $a=8, b=4$

$$\frac{n^{\lg_4 8}}{n\sqrt{n}} = \frac{n^{\frac{3}{2}}}{n^{\frac{3}{2}}} = 1 \quad \text{case 2}$$

$$T(n) = \Theta(n^{\lg_4 8} \lg n)$$

(d) $T(n) = T(n-1) + \lg n$

$$\log 1 + \cancel{\log} 2 + \dots + \lg n = \lg 1 \times 2 \times 3 \times \dots \times n = \lg n!$$

$$= \Theta(n \lg n)$$

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

$$= 2T\left(\frac{n-2}{2}\right) \quad u = n-2$$

$$= 2T\left(\frac{u}{2}\right)$$

$a=2 \quad b=2 \quad n^{\log_2 2 - \epsilon} = \frac{1}{1} \quad \epsilon = 0.5$

case 1 : $\Theta(n^{\log_2 2}) = \Theta(n)$

Problem 3.2. (3 pts) Consider inserting the keys 10, 22, 17, 28, 15, 4, 31, 88, 59 into a hash table of length $m = 11$ using open addressing with the hash function $h_1(k) = k \bmod m$. Illustrate the result of inserting these keys using either linear probing, quadratic probing with $c_1 = 1$ and $c_2 = 3$, or double hashing with $h_2(k) = 1 + (k \bmod (m-1))$.

$\boxed{\text{Linear}}$

$$h(k, i) = (h(k) + i) \bmod m$$

$$h_1(k) = k \bmod m$$

key : 10, 22, 17, 28, 15, 4, 31, 88, 59

0	22
1	88
2	
3	
4	15
5	4
6	17
7	28
8	59
9	31
10	10

key	h(k)	i	h(k)+i
10	10	0	
22	0	0	
17	6	0	
28	6	1	7
15	4	0	
4	4	1	5
31	9	0	
88	0	1	1
59	4	4	8

quadratic

$$h(k,i) = (h(k) + c_1 i + c_2 i^2) \bmod m$$

$$c_1 = 1, \quad c_2 = 3$$

0	22
1	
2	31
3	88
4	15
5	59
6	17
7	
8	4
9	28
10	10

key	$h(k)$	i	$h(k)+i+3i^2$
10	10	0	
22	0	0	
17	6	0	
28	6	2	9
15	4	0	
4	4	1	8
31	9	1	2
88	0	2	3
59	4		

double hashing

$$h(k,i) = (h_1(k) + i\,h_2(k)) \bmod m$$

$$h_1(k) = k \bmod m$$

$$h_2(k) = 1 + (k \bmod (m-1))$$

key : 10, 22, 17, 28, 15, 4, 31, 88, 59

0	22
1	
2	31
3	59
4	28
5	15
6	17
7	88
8	
9	4
10	10

key	$h_1(k)$	i	$h_2(k)$	$(h_1(k)+i\,h_2(k)) \bmod m$
10	10	0		
22	0	0		
17	6	0		
28	6	1	9	4
15	4	2	6	5
4	4	1	5	9
31	9	2	2	2
88	0	1	9	7
59	4	1	10	3

Problem 3.3. (1 pt) Suppose that we are storing a set of n keys into a hash table of size m, what is the best-case searching time, and what is the worst-case searching time? Write down your answer in the form of $T(n) = \Theta(g(n))$. Briefly discuss your answer.

Best-case : $T(n) = \Theta(1)$

Worst-case : 若每次 hash 出來都在同格，會是 $\Theta(n)$

Problem 3.4. (2 pts) Discuss when (1) we have a very small set of possible keys $|U|$; (2) we have a very large number of inserted keys n, relative to the number of all possible keys $|U|$, will hashing still be useful? Explain your answer.

如果所有的 key 都在一個有限的範圍內，
可能數字的重複性就很高，那碰撞的機率很高，
用 open-address 來看，不太可能第一次做就會成功，
自然離 best-case : $O(1)$ 差很多，hashing will not be useful.

Problem 3.5. (1 pt) If the order of the key insertion sequence is changed, will we obtain the same hashed result? Briefly discuss your answer.

keys 順序不同通常就不會有相同的結果，
因為可能 keys 中的數字 A 和 B 會發生碰撞，
原本順序是 A 先用了格子，調換會變成 B 用格子，
但如果都沒有碰撞就完成，調換順序就不影響
(不會發生佔格子的情形)

Problem 3.6. (3 pts) Write pseudocode for HASH-DELETE (using the pseudo-codes similar to slides no. 24 & 25), and modify HASH-INSERT accordingly to handle this situation. Let us assume the link to the element that we want to delete has been given.

在被刪除的位置作記號，這邊是用(*)star表示被刪除的位置

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

在做insert的時候，遇到(*)或是空位就可以直接差值，不用往下找

```
HASH-INSERT(T,k)
i ← 0
repeat j ← h(k,i)
    if T[j] = NIL or T[j] = (*)star
        then T[j] ← k
            return j
    else i ← i+1
until i = m
error "hash table overflow"
```