Algorithms Midterm Exam (Spring 2021)

總分：<u>117%</u>

Name: ______________________

Student ID #: ________________

Question	Score
1 (50%)	
2 (20%)	
3 (20%)	
4 (12%)	
5 (15%)	
Total	

1. **[Simple questions: 50%]** Complete the following questions with simple answers.
   (a) (6%) Name pros and cons when we compare between the quicksort and insertion sort.

   (b) (6%) Name pros and cons when we compare between the mergesort and quicksort.

   (c) (6%) What is your strategy when we need to use radix sort for a set of *signed* floating-point numbers with the format $\pm$xxx.xxxx? That is, the numbers could be positive or negative, with three digits in the integer part and four digits for the floating part below 1.

(d) (6%) Name two difficulties when we allow the DELETE operation in hashing, one for the linked-list case and one for the double hashing case.

(e) (6%) Name one topic that is difficult to you and you cannot handle it well. Name one topic that is difficult but you still can handle it well. Explain your answer briefly.

(f) (5%) Is $\Theta(n) + O(n^2 \log n) = O(n^3)$ correct? Explain your answer.

(g) (5%) Explain why hashing is not needed when we need to build a database for the NTUST students who are majored in CSIE.

(h) (5%) Name one defect if we choose the division method $h(k) = k \mod m$ as the hash function and m is not a prime number.

(i) (5%) Name one defect if we choose the division method $h(k) = k \mod m$ as the hash function and m is a prime number.

2. **[Recurrence equations: 20%]** Solve the recurrence equations in (a) and (b). You can assume the integer arguments for all cases. The answer should be an asymptotically tight solution. You are welcome to apply the master theorem when there is a need. After that, answer the question in (c)

   (a) (5%) $T(n) = 3T(n/\sqrt{3}) + n^2$

   (b) (5%) $T(n) = 9T(n/2) + n^3 \lg n$

(c) (10%) When solving the following two recurrence equations

$T(n) = aT\left(\frac{n}{b}\right) + f(n)$ and $T(n) = aT\left(\frac{n}{b}\right) + g(n)$,

suppose we can find out that the solution for the first equation (on the left) as

$$T(n) = \Theta\left(n^{\log_b a}\right) ,$$

based on the case 1 of the *master theorem* and the solution for the second equation (on the right) as

$$T(n) = \Theta\left(n^{\log_b a} \lg n\right) ,$$

based on the case 2 of the *master theorem*. Can you solve the following recurrence equation

$$T(n) = aT\left(\frac{n}{b}\right) + f(n) + g(n) ,$$

based on the aforementioned information?

You should clearly state how you use the master theorem.

3. **[Quicksort: 12%]** We have the following code for the quicksort algorithm implementation.

```
QUICKSORT(A, p, r)
1   if  p < r
2       then q ← PARTITION(A, p, r)
3           QUICKSORT(A, p, q – 1 )
4           QUICKSORT(A, q + 1, r)
```

```
PARTITION(A, p, r)
1   x ← A[r]
2   i ← p − 1
3   for j ← p to r − 1
4       do if A[j] ≤ x
5           then i ← i + 1
6               SWAP(A[i], A[j])
7   SWAP(A[i + 1], A[r])
8   return  i + 1
```

How many pairwise comparisons (line 4) in the *PARTITION* function for the best and the worst cases of quicksort? An asymptotic result is good enough.

4. **[Heap: 20%]** We have the following codes for heap manipulation. Answer the questions after that.

MAX-HEAPIFY(*A*, *i*)

```
l → LEFT(i)
r → RIGHT(i)
if   l ≤ heap-size[A] and A[l] > A[i]
then   largest  ← l
else   largest  ← i
if   r ≤ heap-size[A] and A[r] > A[largest]
then   largest ← r
if   largest ≠ i
    then   SWAP(A[i], A[largest])
           MAX-HEAPIFY(A, largest)
```

BUIDE-MAX-HEAP(*A*)

```
heap-size[A] ← length[A]
for i ← ⌊length[A]/2⌋ downto 1
    do MAX-HEAPIFY(A, i)
```

HEAP-INCREASE-KEY(*A, i,* key)

```
if key < A[i]
    then error "new key is smaller than current key"
A[i] ← key
while i > 1 and A[Parent(i)] < A[i]
    do SWAP(A[i], A[Parent(i)])
        i ← Parent(i)
```

(a) (5%) If $(16, a_2, 8, 4, 9, a_6, 1, 2, 3, 7\ )$ is a max-heap, do we know the range of a_2 and a_6? Suppose all elements are greater than zero.

(b) (5%) Use *MAX-HEAPIFY*(*A*, 2) to make the array to be a max-heap for $a_2 = 1$.

(c) (5%) Find the result for *HEAP-INCREASE-KEY*(*A*, 6, 18).

(d) (5%) Different from (a) to (c), what we can conclude if we know both the array $(a_1, a_2, \dots, a_n)$ and its reverse array $(a_n, a_{n-1}, \dots, a_2, a_1)$ are max-heap?

5. **[Hashing : 15%]** Answer the following questions related to hashing.

(a) (8%) Demonstrate what happens when we insert the keys 1, 2, 3, 15, 14, 13, 12, 11, 10 into a hash table of size 11 with collision resolved by double hashing. The hash functions are defined by $h_1(k) = k \bmod 11$ and $h_2(k) = 1 + (k \bmod 5)$.

(b) (7%) What may go wrong if we choose the second hash function to be $h_2(k) = 1 + (k \bmod 4)$?