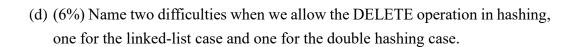
Algorithms Midterm Exam (Spring 2021) 總分: <u>117%</u>

Name:	
Student ID #.	

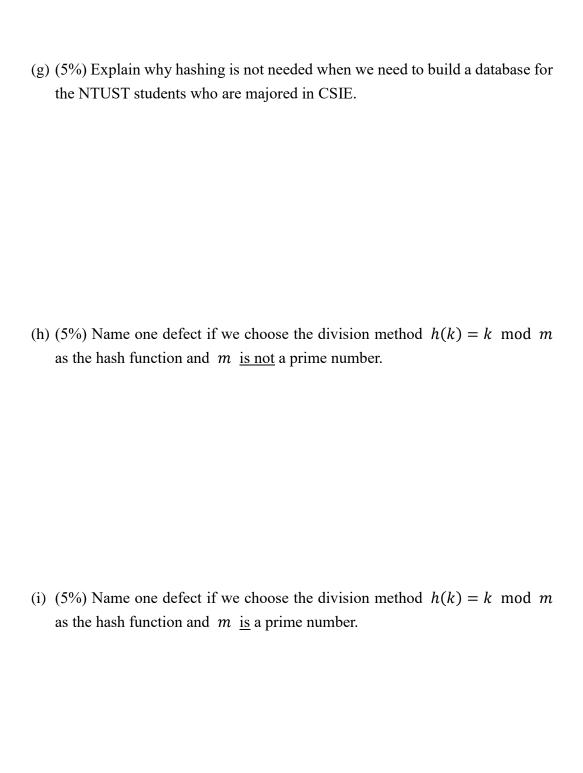
Question	Score
1 (50%)	
2 (20%)	
3 (20%)	
4 (12%)	
5 (15%)	
Total	

(a) (6%	questions: 50°) Name pros rtion sort.								
) Name pros ksort.	and co	ns whe	n we	compare	between	the 1	mergesort	and
floa) What is your ting-point nur d be positive o he floating pa	nbers wor negati	vith the ve, with	forma	at ±xxx.	xxxx? Th	at is	, the num	bers

1.



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



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

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

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

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

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)
     if p < r
2
         then q \leftarrow PARTITION(A, p, r)
3
            QUICKSORT(A, p, q - 1)
4
             QUICKSORT(A, q + 1, r)
PARTITION(A, p, r)
     x \leftarrow A[r]
    i \leftarrow p - 1
2
3
     for j \leftarrow p to r - 1
         do if A[j] \leq x
4
5
             then i \leftarrow 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)
 1 l \rightarrow LEFT(i)
 2 r \rightarrow RIGHT(i)
 3 if l \le \text{heap-size}[A] and A[l] > A[i]
 4 then largest \leftarrow l
 5 else largest \leftarrow i
 6 if r \le \text{heap-size}[A] and A[r] > A[\text{largest}]
     then largest \leftarrow r
 7
    if largest \neq i
 8
 9
        then SWAP(A[i], A[largest])
10
           MAX-HEAPIFY(A, largest)
BUIDE-MAX-HEAP(A)
    heap-size[A] \leftarrow length[A]
    for i \leftarrow \lfloor \operatorname{length}[A]/2 \rfloor downto 1
3
        do MAX-HEAPIFY(A, i)
HEAP-INCREASE-KEY(A, i, key)
    if key < A[i]
2
        then error "new key is smaller than current key"
3 A[i] \leftarrow \text{key}
    while i > 1 and A[Parent(i)] < A[i]
        do SWAP(A[i], A[Parent(i)])
5
6
            i \leftarrow 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 $(a_1, a_2, ..., a_n)$ and its reverse array $(a_n, a_{n-1}, ..., a_2, a_1)$ 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 $h_1(k) = k \mod 11$ and $h_2(k) = 1 + (k \mod 5)$.
 - (b) (7%) What may go wrong if we choose the second hash function to be $h_2(k) = 1 + (k \mod 4)$?