Algorithms Midterm Exam (Spring 2020)

總分：<u>117%</u>

Name: ______________________

Student ID #: ________________

Question	Score
1 (42%)	
2 (20%)	
3 (15%)	
4 (10%)	
5 (10%)	
6 (20%)	
Total	

1. **[Simple questions: 42%]** Complete the following questions with simple answers.
   (a) (5%) We have uploaded video streaming of many kinds, depending on the media that we use and the way we interact with students. Can you name them? Which video streaming style is your favorite?

   (b) (5%) Name a sorting method that is *stable* and *sorted-in-place*. A few explanation is needed.

   (c) (5%) Is it possible to have $O(n) + O(n^2) = O(n^3)$? Explain your answer.

(d) (5%) Given three functions $f(n)$, $g_1(n)$, and $g_2(n)$, if we have $g_1(n) = O(f(n))$ $g_2(n) = \Omega(f(n))$; also, we have $g_1(n) + g_2(n) = \Theta(f(n))$, what conclusion you can draw for the relationship between $f(n)$, $g_1(n)$, and $g_2(n)$?

(e) (5%) If we have a hash table with its size m equal to a prime number, explain the concept of *simple uniform hashing*.

(f) (5%) What is the length of the shortest path in the decision tree model for a quicksort? Why?

(g) (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.

(h) [6%] Suggest a data structure for (i) a group of college students and their grades, (ii) a set of bank customers. Some explanations are necessary.

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%) $T(n) = 9T(n/6) + n$

   (b) (5%) $T(n) = 4T(n/2) + n^3 \lg n\,(1 + \sin n)$

(c) (10%) Let us assume that we have found the solution for a recurrence equation $T(n) = aT(n/b) + f(n)$ such as $T(n) = \Theta(g(n))$.

Now for what choice of $f(n)$, we shall obtain the same solution for the new recurrence equation

$T(n) = a^2 T(n/(b^2)) + f(n)$?

You can make an additional assumption such as when either case 1, 2, or 3 has been applied to the original recurrence. Answer the questions for any two cases up to your decision. Note that the answer may not be unique. Whatever making sense can earn you credits.

3. **[Heap: 15%]** We have the following code 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)
```

(a) (10%) Is ⟨20, 10, 1, 2, 3, 4, 5, 6, 7, 8, 9⟩ a max-heap? Explain your answer if your answer is "yes", or make it a max-heap step-by-step by the above code if your answer is "no".

(b) (5%) What is the effect of calling MAX-HEAPIFY (A, i) for $i > \left(\frac{2}{3}\right)$heap-size[$A$]?

4. **[Quicksort: 10%]** We have a good approach to speed up the quicksort algorithm by substituting small subfiles with insertion sort such as the following implementation:

```
ISS-QUICKSORT(A, p, r, M)
    if   r – p + 1 ≥ M
         then  q  ←  PARTITION(A, p, r)
                     ISS-QUICKSORT(A, p, q – 1)
                     ISS-QUICKSORT(A, q + 1 ,r)
    else if   p < r
         then
                     INSERTION-SORT(A, p, r)
```

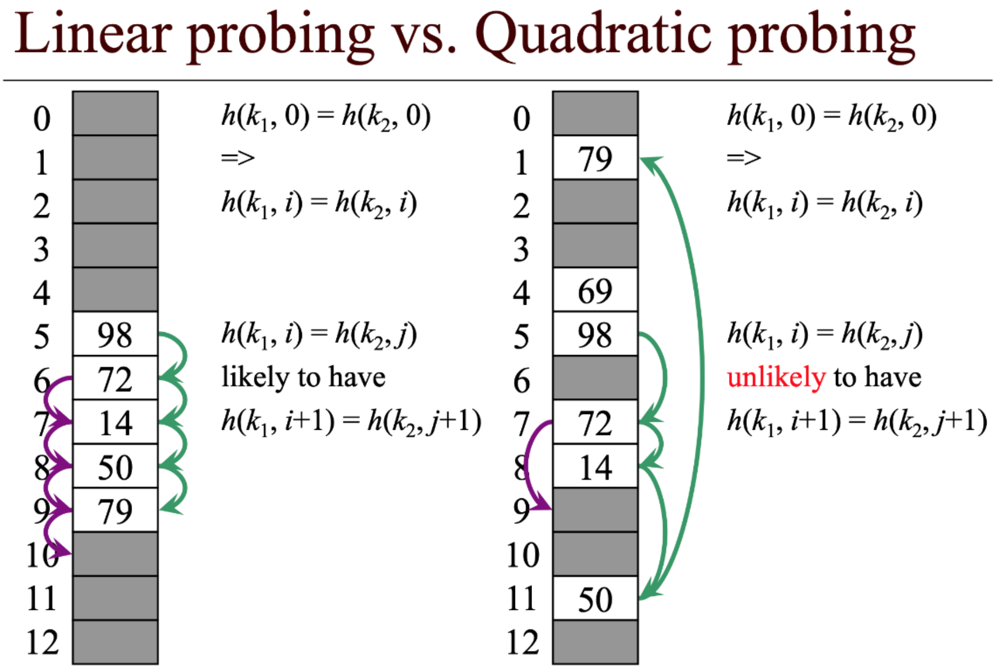
```
PARTITION(A, p, r)
    x ← A[r]
    i ← p − 1
    for j ← p to r − 1
        do if A[j] ≤ x
             then i ← i + 1
                  SWAP(A[i], A[j])
    SWAP(A[i + 1], A[r])
    return   i + 1
```

Compare the performance for the above implementation to the one for the quicksort without using the insertion sort for (i) a sorted input, (ii) an input of all identical elements.

5. **[Sorting in Linear Time: 10%]** Show how to sort $2n$ n-bits numbers efficiently by radix sort.

6. **[Hashing : 20%]** Answer the following questions related to hashing.

(a) (10%) In one of the slides in my lecture,



assume you are a teacher now and I am your student, explain the slide to me as clear as possible.

(b) (10%) Let us choose the *division method* to be the hash function such as $h(k) = k \mod m$. Someone claims that using a prime number for m is a good choice in this case. On the other hand, not using a prime number may introduce some bad performance in hashing. Now discuss two situations: using the division method *without* a prime number in either h_1 or h_2 for double hashing. You may explain your answer by discussing the cases if we try $m = 10$ for h_1 or h_2. What will happen?