Algorithms -- COMP.4040 Honor Statement
(Courtesy of Prof. Tom Costello and Karen Daniels with modifications)

Must be attached to each submission

Academic achievement is ordinarily evaluated on the basis of work that a student produces independently. Infringement of this Code of Honor entails penalties ranging from reprimand to suspension, dismissal or expulsion from the University.

Your name on any exercise is regarded as assurance and certification that what you are submitting for that exercise is the result of your own thoughts and study. Where collaboration is authorized, you should state very clearly which parts of any assignment were performed with collaboration and name your collaborators.

In writing examinations and quizzes, you are expected and required to respond entirely on the basis of your own memory and capacity, without any assistance whatsoever except such as what is specifically authorized by the instructor.

I certify that the work submitted with this assignment is mine and was generated in a manner consistent with this document, the course academic policy on the course website on Blackboard, and the UMass Lowell academic code.

Date: 6 / 19 /2019

Name (please print): PHONG VO

Signature: ____________

COMP.4040 **HW7** Su19

Due Date: 06-20-2019 (Th), <u>BEFORE</u> the class begins

This assignment covers textbook Chapter 8 and Chapter1~7.

1. Sorting Algorithm Property (25 points)

Exercise 8.3-2, textbook p200

2. Counting Sort, Radix Sort, Bucket Sort (30 points)

Illustrate the execution of each of the following sorting methods on the given input:

(1) COUNTING-SORT (show as in Figure 8.2): A = <6, 0, 2, 6, 0, 8>

(2) RADIX-SORT (show as in Figure 8.3): English words: PAT, CAT, CART (*right-justify this word* so that T is compared first), FAT, FIX. Padded with white space when needed.

(3) BUCKET-SORT (show as in Figure 8.4): A = < 0.67. 0.82, 0.12, 0.46, 0.88, 0.61>

3. Counting Sort, Radix Sort (25 points)

(1) (20 points) Exercise 8.3-4, textbook, p200

(2) (5 points) What is the running time if we use Counting Sort? Justify your answer.

4. Sorting (20 points)

Exercise 8.4-2, textbook p204

Su19

HW7

1/ Sorting Algorithm Property

Exercise 8.3-2

- The sorting algorithms are stable: Insertion Sort, Merge Sort ✓

  The sorting algorithms are not stable: Heap Sort, Quick Sort ✓
- Simple Scheme that makes sorting algorithm stable:
  - Store the original index of each element ✓
  - Use that index as a secondary way of sorting elements with equal primary value.
- To implement this, the comparison function or operator (for example <) would be implemented so A < B returns TRUE if A.*originalIndex* is less than or equal to B.*originalIndex*. The otherwise induction returns FALSE.
- This requires one additional *originalIndex* to be stored per element. There are n elements, hence Θ(n*l*gn) extra space is required.

✓

Additional time? -1

HW7

2/ Counting Sort, Radix Sort, Bucket Sort

(a) Counting Sort: A = <6, 0, 2, 6, 0, 8>

	1	2	3	4	5	6
A =	6	0	2	6	0	8

A' = compare every element of A vs. C'

C =	2	0	1	0	0	0	2	0	1
	0 +	1	+ 2	3	4	5	6	7	8

C' =	2	2	3	3	3	3	5_{-1}	5	6
	0	1	2	3	4	5	6	7	8

A'

j=6

				~~8~~	8
1	2	3	4	5	6

C' =	2_{-1}	2	3	3	3	3	4	5	6
	0	1	2	3	4	5	6	7	8

j=5

	0			~~6~~	8
1	2	3	4	5	6

−4

C' =	1	2	3_{-1}	3	3	3	4	5	6
	0	1	2	3	4	5	6	7	8

	0	~~2~~		6	8
1	2	3	4	5	6

C' =	1	2	2	3	3	3	4_{-1}	5	6
	0	1	2	3	4	5	6	7	8

	0	2	6	6	
1	2	3	4	5	6

C' =	1_{-1}	2	2	3	3	3	3	5	6_{-1}
	0	1	2	3	4	5	6	7	8

0	0	2	6	6	
1	2	3	4	5	6

C' =	0	2	2	3	3	3	3	5	6_{-1}
	0	1	2	3	4	5	6	7	8

0	0	2	6	6	8
1	2	3	4	5	6

⇑ Sorted A

C' =	0	2	2	3	3	3	3	5	5

(b) Radix Sort: PAT, CAT, CART, FAT, FIX

	P	A	T
	C	A	T
C	A	R	T
	F	A	T
	F	I	X

=>

	P	A	T
	C	A	T
	F	A	T
	F	I	X
C	A	R	T

=>

C	A	R	T
	C	A	T
	F	A	T
	F	I	X
	P	A	T

=>

	C	A	T
	F	A	T
	F	I	X
	P	A	T
C	A	R	T

(c) Bucket Sort: A = <0.67, 0.82, 0.12, 0.46, 0.88, 0.61>

A has 6 elements

$\lfloor 0.67 * 6 \rfloor = \lfloor 4.02 \rfloor = 4$

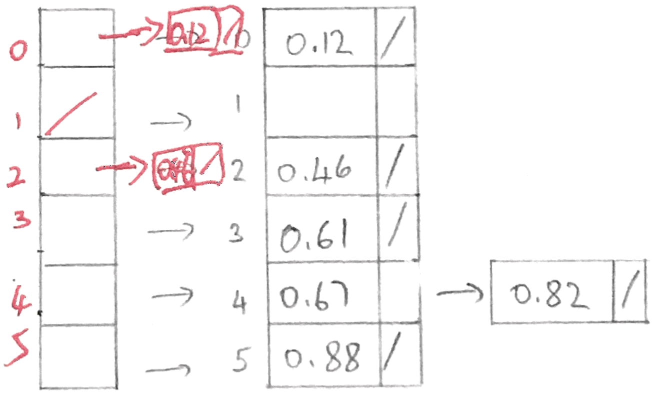
$\lfloor 0.82 * 6 \rfloor = \lfloor 4.92 \rfloor = 4$

$\lfloor 0.12 * 6 \rfloor = \lfloor 0.72 \rfloor = 0$

$\lfloor 0.46 * 6 \rfloor = \lfloor 2.76 \rfloor = 2$

$\lfloor 0.88 * 6 \rfloor = \lfloor 5.28 \rfloor = 5$

$\lfloor 0.61 * 6 \rfloor = \lfloor 3.66 \rfloor = 3$



⇨ Answer is <0.12, 0.46, 0.61, 0.67, 0.82, 0.88>

HW 7

3/ Counting Sort, Radix Sort   Exercice 8.3-4 p200

(1) Assume the integers are base n

For general case, k-base system

in range 0 to r-1

$d = [\log_k (r)]$ digits

Now, if we set $k = n$, $r = n^3$

$\Rightarrow d = \log_n n^3 = 3 \log_n n = 3$

Now, each integer can be represented in this n-base system by Solving for a, b, c in a format as

$$X = an^2 + bn^1 + cn^0$$

$$= an^2 + bn + c \qquad [0 \leq a, b, c \leq n-1]$$

$\Rightarrow$ The number in n-base system is (abc)

As provided, the longest number (n^3-1) can be:

$$(n-1)n^2 + (n-1)n^1 + (n-1)n^0$$

$$= (n-1)^2 + n^2 - n + n^2 - 1 = n^3 - 1$$ ✓

Thus, n-base system is represented as $(n-1 \;\; n-1 \;\; n-1)$

Radix-sort is applied $\Rightarrow \theta(d(n+k))$

$= \theta(3(n+n))$

$= \theta(n)$ time

(2). Countig Sort ? $O(n^3)$

-5

3. $n^3 - 1$

~~Treat the numbers as~~

Counting sort:

$$\theta(k+n) = \theta(n^3 + n) = \theta(n^3)$$

HW7

4/ Sorting: Exercise 8.4-2 p204

- Why the worst-case running time for bucket sort is $\Theta(n^2)$?
  - ⇨ Bucket Sort executes like Insertion Sort does, where all elements are inserted into the sorted array. This cause the array will be re-arranged n^2 times. So the worst-case running time for Bucket Sort is $\overset{O}{\Theta}(n^2)$.
- Solving:
  - ⇨ To fix the worst-case running time to be $\Theta(nlgn)$, we should call Merge Sort ✓ algorithm at each sorting step due to Merge Sort has worst-case running time is $\Theta(nlgn)$.

90/100