**Due Date**: 04-19-2019 (F), BEFORE the class begins

BEFORE 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

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: 04/19/2019

Name (please print): DANGNHI NGO

Signature: [illegible]

100/100

25 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 (for example <) would be implemented so A < B returns true if A.originalIndex is less than or equal to B.originalIndex; otherwise, it returns false
* This requires one additional originalIndex value to be stored per element
  There are n elements hence $\Theta(n)$ extra space is required

30 2/ Counting Sort, Radix Sort, Bucket Sort
(1) COUNTING-SORT: A = <6, 0, 2, 6, 0, 8>

A

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

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 | 5 | 6 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

⇓

B

| | | | | | |
|---|---|---|---|---|---|
| | | | | | 8 |
| 1 | 2 | 3 | 4 | 5 | 6 |

C

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 2 | 2 | 3 | 3 | 3 | 3 | 5 | 5 | 5 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

↓

| | | | | | |
|---|---|---|---|---|---|
| | 0 | | | | 8 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 3 | 3 | 3 | 5 | 5 | 5 |

| | | | | | |
|---|---|---|---|---|---|
| | 0 | | | 6 | 8 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 3 | 3 | 3 | 4 | 5 | 5 |

↓

| | | | | | |
|---|---|---|---|---|---|
| | 0 | 2 | | 6 | 8 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 2 | 3 | 3 | 3 | 4 | 5 | 5 |

↓

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 2 | | 6 | 8 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0 | 2 | 2 | 3 | 3 | 3 | 4 | 5 | 5 |

↓

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

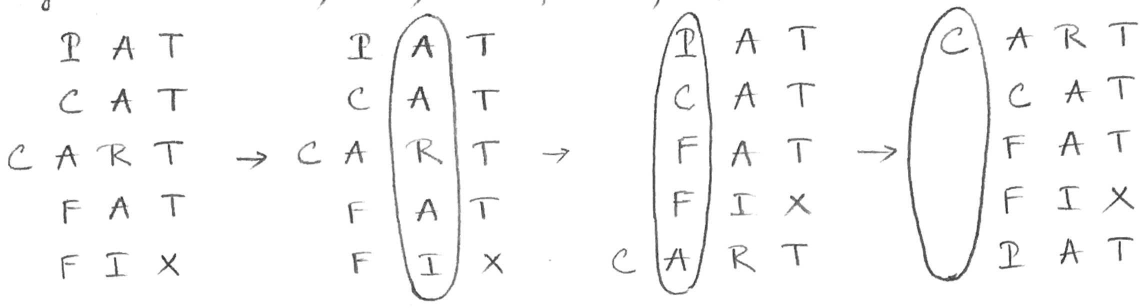
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0 | 2 | 2 | 3 | 3 | 3 | 3 | 5 | 5 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

Answer B

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 2 | 6 | 6 | 8 |

(2) RADIX - SORT

English words: PAT, CAT, CART, FAT, FIX



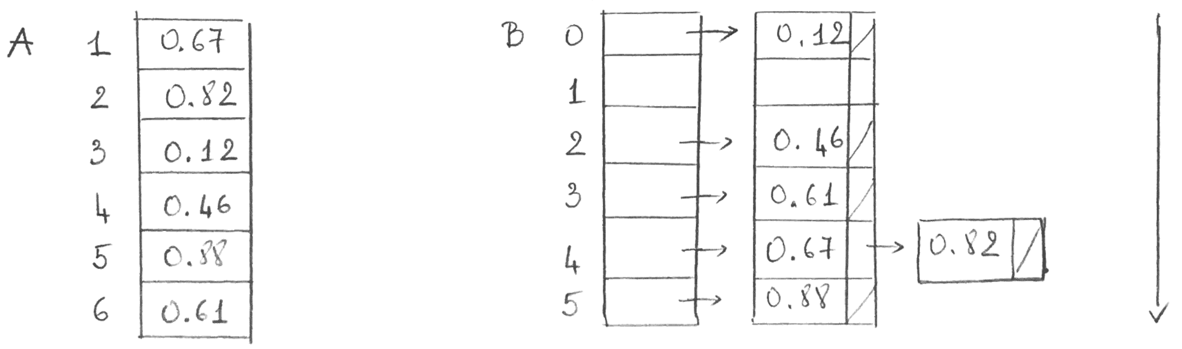
$\Rightarrow$

CAT
FAT
FIX
PAT
CART

Answer: CAT, FAT, FIX, PAT, CART

(3) BUCKET - SORT

A = < 0.67, 0.82, 0.12, 0.46, 0.88, 0.61 >



floor (0.67 × 6) = floor (4.02) = 4
floor (0.82 × 6) = floor (4.92) = 4
floor (0.12 × 6) = floor (0.72) = 0
floor (0.46 × 6) = floor (2.76) = 2
floor (0.88 × 6) = floor (5.28) = 5
floor (0.61 × 6) = floor (3.66) = 3

Answer: 0.12, 0.46, 0.61, 0.67, 0.82, 0.88

2J 3/ Counting Sort, Radix Sort

(1) Exercise 8.3-4

Show how to sort $n$ integers in the range 0 to $n^3 - 1$ in $O(n)$ time

Considering the number to be base $n$

For general case : $k$-base number system,

# in range 0 to $r-1$

Can be represented as, using $d = [\log_k(r)]$ digits

$\Rightarrow$ Now, if we set $k = n$

$r = n^3$

So, we can represent $n^3$ number using $d = \log_n(n^3) = 3$

Thus, 3 digits

$\Rightarrow$ So, now the number can be represented in this $n$ - base system by solving for $a$, $b$ & $c$ in

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

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

As ex: the longest number $(n^3 - 1)$ can be :

$$(n-1)n^2 + (n-1)n + (n-1)n^0$$
$$= (n-1)[n^2 + n + 1]$$
$$= (n-1)\frac{(n^3-1)}{(n-1)} = n^3 - 1$$

Thus, $n$ - base system the representation is $(n-1 \;\; n-1 \;\; n-1)$

Radix-sort to converted 3-digit base #,

$$\Rightarrow \theta(d(n+k))$$
$$= \theta(3(n+n))$$
$$= \theta(n) \text{ time}$$

(2) If we use Counting Sort :

From previous ex, we start with running through the list of integers and convert each one to base $n$, then radix sort them

Each number will have at most $\Rightarrow \log_n(n^3) = 3$ digits

So, there will only need to be 3 passes

For each pass, there are $n$ possible values, which can be taken on.

so now using counting sort to sort digit (3 digit) in $O(3n)$ time

20 4/ Sorting

Exercise 8.4-2

* Why the worst-case running time for bucket sort is $\theta(n^2)$
When all the inputs fall into a single bucket, and since we use Insertion-sort for sorting buckets which algorithm has a worst case of $\theta(n^2)$
Thus, at the end Bucket-sort worst-case running time become $\theta(n^2)$
* Solving : Simple change
We should use algorithm like Merge-Sort with worst-case, and now worst-case running time for Bucket-Sort becomes $O(n\lg n)$ (instead of using Insertion-Sort)
So now we can ensure that the worst-case of Bucket-Sort is $O(n\lg n)$ without affecting the average case running time