

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.

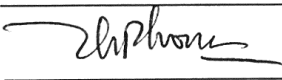
Your name on any exercise is regarded as assurance and certification that what you are submitting for that exercise is the result of your own thoughts and study. Where collaboration is authorized, you should state very clearly which parts of any assignment were performed with collaboration and name your 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: 5/27/2019

Name (please print): PHONG VO

Signature: 

Due Date: May 23, 2019 (Th), BEFORE the lecture starts

This assignment covers textbook Chapter1~2. A paper version must be submitted. Please keep a copy of your solution for yourself.

1. **Compare Functions:** (20 points) What is the smallest integer value of $n > 3$ such that an algorithm whose running time is $10n$ runs *slower than* an algorithm whose running time is $9(\log_2 n)^4$ on the same machine? Justify your answer. (Hint: You may write a program, draw a plot, or/and proof)

2. **Pseudocode and Loop Invariant:** (20 points) textbook, Exercise2.2-2, p29, Selection Sort.

When calculating the running time, you don't need to list the cost for each line and the number of execution times for each line for this question. Please use big-theta notation (θ) to represent the running time $T(n)$. Also, specify which line of your code contribute most to the total running of time; and then calculate the total number of times executing that particular line. Write a summation for it and calculate the summation.

3. **Sorting Algorithms:** (20 points) Using textbook Figure 2.2 and Figure 2.4 as models to illustrate the operations of Insertion_Sort and Merge_Sort on the array $A = \langle 38, 10, 83, 65 \rangle$

4. **Analysis:** (20 points) There is a mystery function called $\text{Mystery}(n)$ and the pseudocode of the algorithm is shown as below. Please analyze the worst-case asymptotic execution time of this algorithm using the method we learn in the class. Express the execution time as a function of the input value n . Assume that $n = 3^k$ for some positive integer $k \geq 1$. Justify your answer.

Hint:

- (a) Draw a recursion tree to help with your analysis.
- (b) Appendix A may help with your calculation

```

Mystery(n)
1  if  $n \leq 1$ 
2      return 1
3  for  $i=1$  to  $n$ 
4      for  $j = 1$  to 5
5          print "this is a recursive call."
6  Mystery ( $n/3$ )
7  Mystery ( $n/3$ )
8  Mystery ( $n/3$ )

```

5. **Algorithm Design** (20 points)

Input: array A contains n distinct numbers from 1 to n , in arbitrary order.

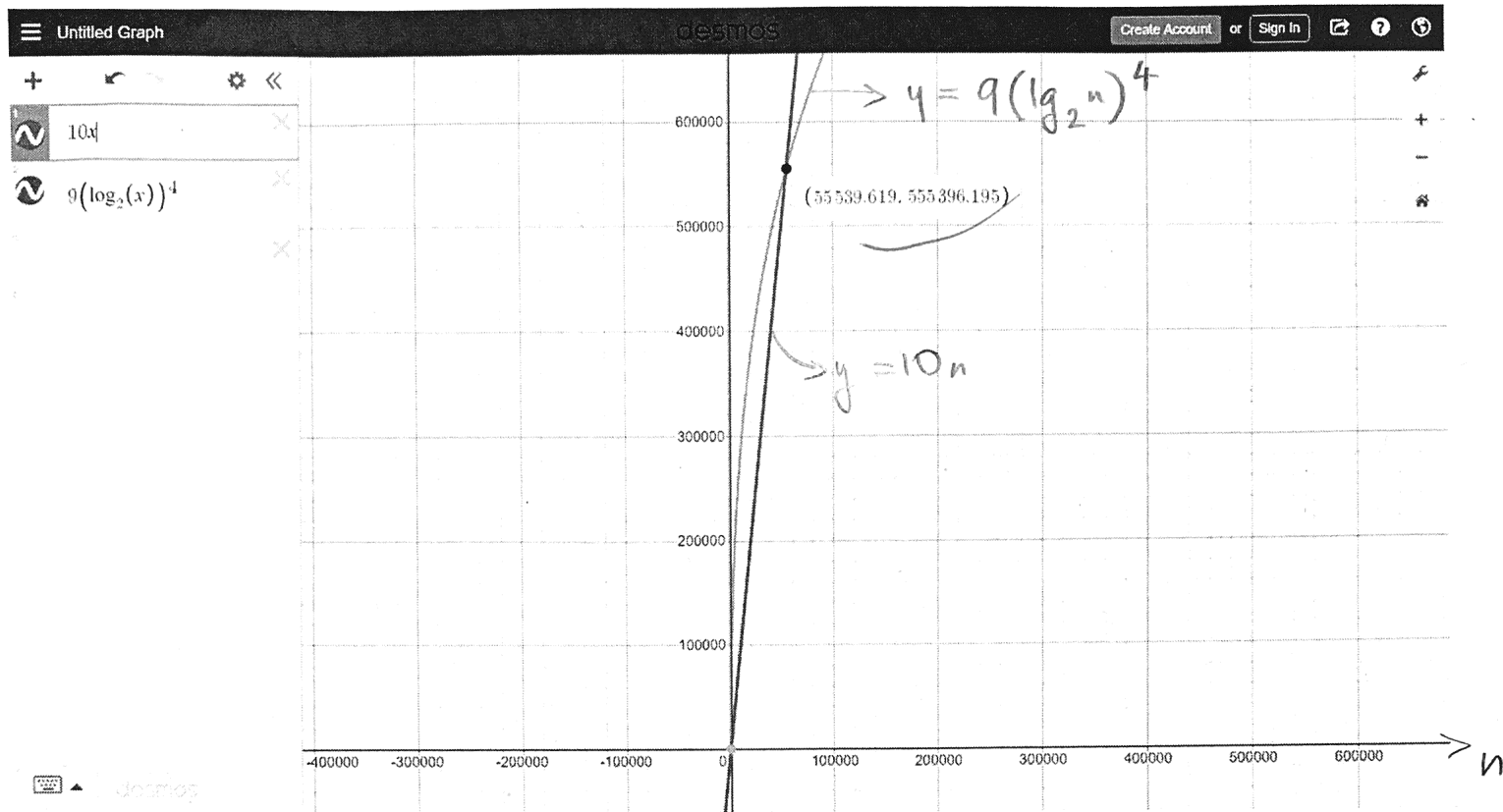
Output: *number of inversions* (defined as the number of pair (i, j) of array indices with $i < j$ and $A[i] > A[j]$).

- (a) (5 points) What array with elements from the set $\{1, 2, \dots, n\}$ has the most inversions? How many does it have?
- (b) (15 points) Create an algorithm using divide-and-conquer approach that determines the number of inversions in any permutation on n elements in $\Theta(n \lg n)$ worst-case time (Hint: modify the merge sort).

(The Honor Statement is in the next page. Please sign and attach it to your homework solution as the last page.)

Problem 1:

(1)



Function $y = 10n$ and $y = 9(\log_2(n))^4$ have 3 intersects:

$$I_1(0.5429, 5.429), \quad I_2(2.432, 24.32)$$

and $I_3(55539.619, 555396.195)$

I realize that when n is in the range $(2.432, 55539.619)$ then the function $y = 10n$ grows slower than the algorithm of the function $y = 9(\log_2(n))^4$.

* The best case: when all elements are sorted:

(2)

$$T(n) = c_1 \cdot n + c_2(n-1) + c_3 \left((n-1)n + (n-1)2 - \frac{(n-1)n}{2} \right)$$

$$+ c_4 \left((n-1)n + (n-1)(1) - (n-1) \frac{n}{2} \right) + c_6(n-1)$$

$$= n + n + n^2 - n + 2n - \frac{n^2}{2} + \frac{n}{2} + n^2 - n + n - \frac{n^2}{2} + \frac{n}{2} + n$$

$$= n^2 + 5n = \Theta(n^2)$$

* The worst case is same as the best case with line 5 and 7 are always executed.

$$T(n) = c_1 \cdot n + c_2(n-1) + c_3 \left((n-1)n + (n-1)2 - \frac{(n-1)n}{2} \right)$$

$$+ c_4 \left((n-1)n + (n-1)(1) - (n-1) \frac{n}{2} \right) + c_5 \left((n-1)n - (n-1) \frac{n}{2} \right)$$

$$+ c_6(n-1) + c_7(n-1)$$

$$= \underbrace{n^2 + 5n}_{\text{Best case}} + n^2 - n - \frac{n^2}{2} + \frac{n}{2} + n = \frac{3n^2}{2} + \frac{11n}{2} = \Theta(n^2)$$

⇒ Best case and worst case have the same growing rate n^2

* Line 3 runs in most of the running time because it is the main part of the code. The total number of times =

$$= c_3 \left((n-1)n + (n-1)2 - \frac{(n-1)n}{2} \right) = c_3 \left(n^2 - n + 2n - 2 - \frac{n^2}{2} + \frac{n}{2} \right)$$

$$= c_3 \left(\frac{n^2}{2} + \frac{3n}{2} - 2 \right)$$

loop invariant?

why does it need even for only first $n-1$ elements

-5

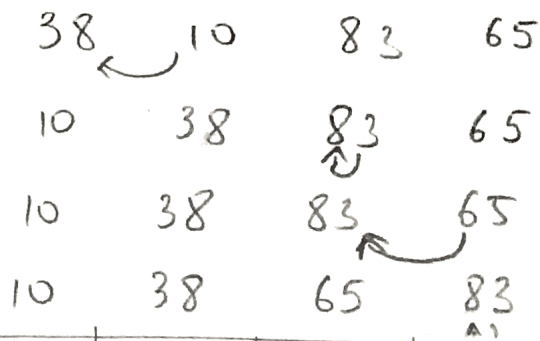
② Selection sort: pseudo code:

| | | | |
|---------------------------|-------|----|-------------------------------------|
| | n | 1. | for ($i=1; i < n; ++i$) |
| | $n-1$ | 2. | min_idx = i ; |
| $\sum_{i=1}^n n+2-i$ | | 3. | for ($j=i+1; j \leq n; ++j$) |
| $\sum_{i=1}^n n+1-i$ | | 4. | if ($A[j] < A[\text{min_idx}]$) |
| max of $\sum_{i=1}^n n-i$ | | 5. | min_idx = j ; |
| $n-1$ | | 6. | if min_idx $\neq i$ |
| max of $(n-1)$ | | 7. | Swap($A[\text{min_idx}], A[i]$); |

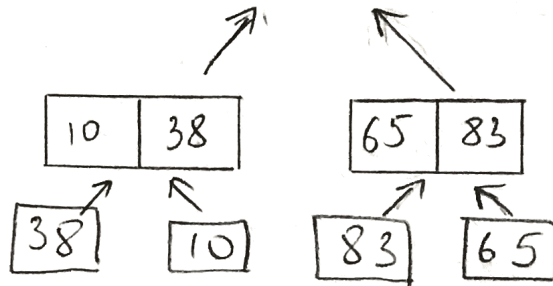
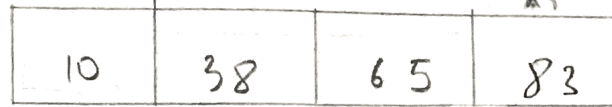
(3)

$$A = \langle 38, 10, 83, 65 \rangle$$

Insertion Sort:



Merge Sort:



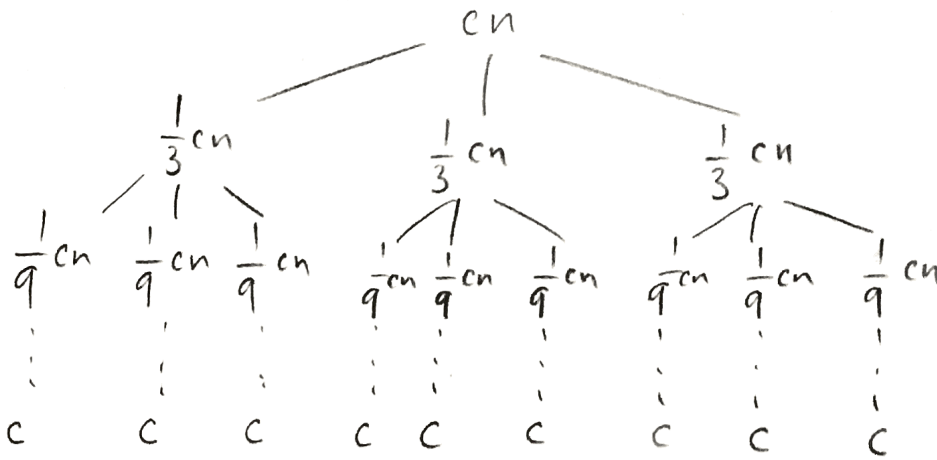
Merge

Merge

$$(4) \quad T(n) = 3T\left(\frac{n}{3}\right) + \underbrace{cn}_{\text{Nested for loops}}$$

Mystery

Recursive tree:



| level | leaves costs | Total cost |
|-------|------------------------|------------|
| 1 | cn | cn |
| 2 | $\frac{1}{3} cn$ | cn |
| 3 | $\frac{1}{9} cn$ | cn |
| i | $\frac{1}{3^{i-1}} cn$ | cn |

At level i : leaves cost = $\frac{1}{3^{i-1}} cn$ whereas total cost = cn

$$\frac{cn}{3^{i-1}} = c \Leftrightarrow \frac{n}{3^{i-1}} = 1 \Leftrightarrow 3^{i-1} = n \Rightarrow i-1 = \log_3 n$$

$$\Rightarrow \boxed{i = 1 + \log_3 n}$$

(4)

(5) The original array $[1, 2, 3, 4, \dots, n]$ (n-1) pairs to be inverted
 a) The inverted array $[n, n-1, n-2, \dots, 1]$
 \Rightarrow The number of inversions $= \sum_{i=1}^n A[n-i+1] = \frac{n(n-1)}{2}$

b) merge-sort(A, p, r)

invNum = 0

if (p < r)

$q = \lfloor \frac{p+r}{2} \rfloor$ // median value

invNum = count_inversions(A, p, q)

invNum += count_inversions(A, q+1, r)

invNum += merge(A, p, q, r)

return invNum

merge(A, p, q, r)

$n_1 = q - p + 1$

$n_2 = r - q$

let $L[1 \dots n_1+1]$ and $R[1 \dots n_2+1]$ be new arrays

for $i=1$ to n_1

$L[i] = A[p+i-1]$

for $j=1$ to n_2

$R[j] = A[q+j]$

$L[n_1+1] = \infty$

$R[n_2+1] = \infty$

$i=1$

$j=1$

invNum = 0

95/100

(see next page, pls) \Rightarrow

for $k = p$ to n

if $L[i] \leq R[j]$

$A[k] = L[i]$

$i++$

else

$A[k] = R[j]$

$j++$

$invNum += q - i$

// if elements of R are taken, then the
// remaining elements of L are all "inverted".

return invNum

Count_inversions(A, p, q)

let $temp[1..p+q]$ be a new Array // $p+q = n$: array's size

for $i = 1$ to $(p+q)$

$temp[i] = A[i]$

return merge_sort($temp, 1, p+q$)