EXAMINATION BLUE BOOK

NAME Joshua Griffiths

SUBJECT Algo

INSTRUCTOR Chen

EXAM SEAT NO. SECTION

DATE 10/14/17 GRADE 88/100

University of Colorado Boulder

Honor Code Pledge

On my honor, as a University of Colorado Boulder student, I have neither given nor received unauthorized assistance on this work.

Signature Date 10/18/17

14/20

1a) $f(g(n)) = 10\, n^{4/3} \lg n^{4/3}$

$\Omega(n^{4/3} \log n^{4/3})$ $\Theta(n^{4/3} \lg n^{4/3})$ $O(n^{4/3} \lg n^{4/3})$ 0

1b) $a = 3$ $b = 3$ 5 $f(n) = cn$

$\log_3 3 = 1$ $\Theta(n^{\log_3 3}) = \Theta(n^1) = f(n) = n$

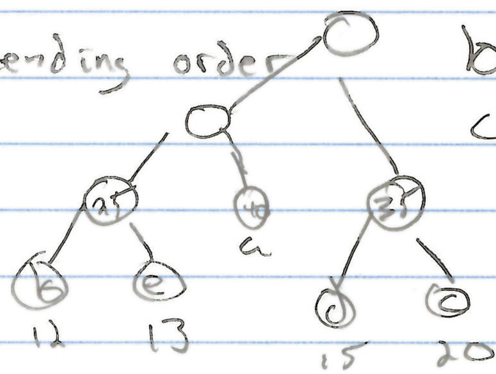
$\therefore T(n)\ \Theta(n \lg n)$ 2

1c) Second one 2

1d) False 2

1e) An array in ascending order 2

1f) a 40
~~b 12~~
~~c 20~~
~~d 15~~ be = 25
~~e 13~~
15 dc = 35 0

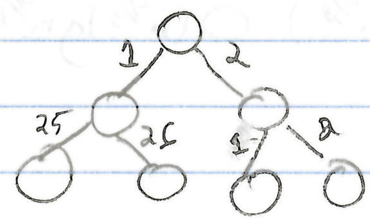


b: 000
c: 11

1g) Pr(2 keys same slot) = 1 0

1h) 11, 22, 33, 44, 55, 66, 77, 88 2

1i) ~~when the shortest~~ di·



2

(j) A-B

2

$m = 2 \quad n = 4$

2a)
```
computeProduct(m, n){
    prod = 0;
        for( i= 1 ; i≤n; i++){
            prod = prod + m
        }
    return prod;
```

$\frac{10}{10}$

i = 1 1st iter: $m = 0 + 2 = 2$

i = 2 2nd iter: $m = 2 + 2 = 4$

i = 3 3rd iter: $m = 4 + 2 = 6$

i = 4 4th iter: $m = 6 + 2 = 8$ ✓

2b) $\Theta(n)$

3) $A = [1, 2, 3]$ $B = [4, 5, 6]$

$\frac{5}{5}$

4) $T(n) = 2(\frac{n}{2}) + n$

$a = 2$ $b = 2$ $f(n) = n$

$\log_2 2 = 1$ $\Theta(n^{\log_2 2}) = \Theta(n) = f(n) = n$

$\therefore T(n) = \Theta(n \lg n)$

$\frac{10}{10}$

21 =

4b) 1, 2, 3, 4, 5, 6,

6 15

$\boxed{i}$, $\boxed{ii}$

1 2 3 4 5 6

3, 3 9, 6

5a) 1st line $x = A[r]$ changes to $x = A[floor((p+r)/2)]$

7th line the final swap changes from $x = A[r]$

to swap$(A[i+1], A[floor((p+r)/2)])$

$\frac{6}{5}$

4th line if $A[j] \le x$ changes to

if $A[j] < x$ && $j \ne floor((p+r)/2)$

last line changes return $i+1$ to return i

5b) A = [2, 6, 4, 1, 5, 3] $q_a = 4$, $q_B = 6$

x = A[floor((5+0)/2)] = A[2] = 4

i = -1

j = 0 : i = 0   A = [2, 6, 4, 1, 5, 3]

j = 1 : x

~~j = 2 : i = 1 swap(A[1], A[2]) A = [2, 4, 6, 1, 5, 3]~~

j = 3 : i = 1 swap(A[1], A[3]) A = [2, 1, 4, 6, 5, 3]

j = 4 : x

j ~~= 5 : i = 2 swap(A[2], A[5]) A = [2, 1, 3, 6, 5, 4]~~

swap(A[i+1]), A[floor((5+0)/2)] A[2])

A = [2, 1, 6, 3, 5, 4]

B = [2, 6, 1]

x = 6

i = -1

j = 0 : i = 0

j = 1 : x

j = 2 : i = 1   swap(B[1], B[2]) B = [2, 1, 6]

swap(B[2], B[1]) B = [2, 6, 1]

return i+1 = 2   null

$\frac{20}{20}$

6a) $\Omega(n^2)$ $\Theta(n^2)$ $O(n^2)$ 5

6b) @ the start of each iteration of the outside for loop the subaray A' [1 ... j-1] contains elements in ascending sorted order 5

Initialization:

6c) when length(A) = 0 the first 0 elements of Array A are trivialy sorted

Maitenence:

when inner loop terminates either

1) A[i ... n] > A[imin] in which case imin = j thus A[i ... n] > A[j] and loop invariant holds

10

2) or A[i ... n] < A[j] & i ≠ j swap(A[i], A[j]) and thus the ith element of A is now greater than the jth element of A thus A[i] must be the ith smallest element of A and our loop invariant holds

Termination: When j = n-2 and i = n-1 array A[1 ... n] contains n elements of A in ascending order

(a) $\Omega(n^2)$ $\Omega(n^2)$ $O(n^2)$

(b) (i) At the start of each iteration of the outside for loop the subarray $A'[1 \ldots i-1]$ contains elements in ascending sorted order

Initialization:

(ii) when $length(A) = 0$ the first 0 elements of array A are trivially sorted

Maintenance:

when inner loop terminates either

1) $A[1 \ldots i-1] \geq A[i+1]$ which case

swap(...) then $A[1 \ldots n] \geq A[j]$ and loop invariant holds

10

2) $A[1 \ldots n] < A[j]$

swap($A[j]$, $A[j+1]$) and thus the ith element of A is now greater than the jth element of A thus $A[i]$ must be the ith smallest element of A and thus loop invariant holds

Termination: when $j = n+2$ and the array $A[1 \ldots n]$ contains n elements of A in ascending order

7) // create binary search tree for two array
// at each node of A search B for
$a_i + b_i = 0$

10

```
findZero(A, B) {

    createTree(A);
    createTree(B);
  for i = a_0 to a_n
      zero = binarySearch(B, -a_i) //search array B
                                     for -a_i
  if (zero =! null) {
      return zero

  else return null
```

algorithm iterate through A n times and on each iteration does a binarySearch which takes lgn time that total time complexity O(nlgn)

8a) $T(n) = 2T(\frac{n}{2}) + n$

$T(1) = 1$

n because 2 for loops one running from $\frac{n}{2} \to 0$ and another running from $\frac{n}{2}+1 \to n$)

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

$\frac{15}{15}$

8b) $\log_2 2 = 1$ $\Theta(n^{\log_2 2}) = \Theta(n^1) = f(n) = n$

$\therefore T(n) = \Theta(n \lg n)$