10

Question 3: 11 points (4 + 3 + 4)

a. You would like to determine which of your Facebook friends are early adopters. So, you have decided to sort them using Facebook account ids which are 64-bit numbers. Which sorting algorithm will be most appropriate – **Insertion sort, Merge sort, Quicksort, Counting sort or Radix sort? Explain why.**

Ans. Radix sort would be most appropriate. Because our data, account ids, is consisting of 64-bit numbers it would be good to sort each digit starting from right most digit first. The 64-bit number consists 4 section as: 63–48 | 47–32 | 31–16 | 15–

We place each digit in every second and using Radixsort we sort it starting from right most to left.

sorting direc

-1

b. Use RadixSort, using LSB to MSB (IBM method) for the following array:

A = [455, 61, 63, 45, 67, 135, 74, 49, 15, 5]

Ans.

455		061		005		005		5
061		063		015		015		15
063		074		135		045		45
045		455		045		049		49
067	=>	045	=>	049	=>	061	=	61
135		135		455		063		63
074		015		061		067		67
049		005		063		074		74
015		067		067		135		135
005		049		074		455		455

2 ✓

8 +1 = 9 ⇒ 9.5

Question 1: 11 points (3 + 4 + 4)

a. Show the running time of the following code using big O notation (*show your work*):

```
for (int j = 4; j < n; j=j+2) {          ≈ n/2     → (n/2 + 1)
val = 0;                                  1
    for (int i = 0; i < j; ++i)           n-1       = (n-1) + 3(n-1)
    {                                               = n-1 + 3n-3 = (4n-4)
        val = val + i * j;                3(n-1)
        for (int k = 0; k < n; ++k)       n-1       = (n-1) + 2(n-1)
            {                                       = n-1 + 2n-2
            val++;                        2(n-1)    = (3n-3)
            }
    }
}
```

Ans. Hence it a nested for loop, will multiply

$= (n/2+1)(4n-4)(3n-3)$

$= (2n^2-2n+4n-4)(3n-3)$

$= (2n^2+2n-4)(3n-3)$

$= 6n^3 - \cancel{6n^2} + \cancel{6n^2} - 6n - 12n + 12 \quad = O(6n^3 - 18n + 12) = \underline{O(n^3)}$

$=$

b. Determine whether f is O(g), Theta (g) or Omega (g). Mention all that applies. Show your work.

i) Show that 2^{2n} != $O(2^n)$ (!= means not equal to).

Ans. $\lim_{n \to \infty} \frac{2^{2n}}{2^n} = \lim_{n\to\infty} 2^n = \infty$, hence $2^{2n} != O(2^n)$?

O ~~than~~ ~~case~~

-1 ⇒ 0

ii) f = n ^ k and g = c ^ n; c > 1 and k >= 1.

Ans. $f = n^k$, $g = c^n$

$f(n) < c\,g(n)$

$0.5n^k < c \cdot c^n \quad \forall n \geq n_0$

take $n_0 = 1$

hence f is O(g)

-1 ⇒

f is o(g)

small

Maharishi International University (MIU)

MIDTERM

Course Title and Code: *CS 435 - Design and Analysis of Algorithms*
Instructor: *Dr. Emdad Khan*
Date: *Friday 07/29/2022*
Duration: *10am – 12:30 pm*

Student Name:

Amanuel Hadgu Tareke

Student ID:

614752

Total Mark
38 → 39 +1 = 40 → 40.5
46

1. This is a closed book exam. Do not use any notes or books!
2. Show your work. Partial credit will be given. Grading will be based on correctness, clarity and neatness.
3. We suggest you to read the whole exam before beginning to work on any problem.
4. There are 4 questions worth a total of 46 points, on 8 pages (including this one)
5. **You can use all back pages for scratch paper (may use for answers if you have used up the designated space for answer).**
6. **You can use a basic calculator (No smart phone unless network is disabled).**

Question 2 (contd): 12 points (4 + 4 + 4)

c. **Use Induction to prove that**

$$\sum_{r=1}^{n} r(r+1) = \frac{1}{3}n(n+1)(n+2)$$

Ans.

First lets prove base case

i) if $n=1$

$\Rightarrow \sum_{r=1}^{1} r(r+1) = 1(1+1) = 2$ //

$\Rightarrow \frac{1}{3}(1+1)(1+2) = \frac{6}{3} = 2$ //, hence proved

ii) Assume this is true for k and let's prove for $k+1$

$\sum_{r=1}^{k} r(r+1) = \frac{1}{3}k(k+1)(k+2)$ // Assume

$\sum_{r=1}^{k+1} r(r+1) = \sum_{r=1}^{k} r(r+1) + (k+1)((k+1)+1)$

$= \frac{1}{3}k(k+1)(k+2) + (k+1)(k+2)$

$= \frac{k(k+1)(k+2) + 3(k+1)(k+2)}{3}$

$= \frac{1}{3}(k+1)(k+2)(k+3)$

$= \frac{1}{3}(k+1)((k+1)+1)((k+1)+2)$ // hence proved

$\therefore \sum_{r=1}^{n} r(r+1) = \frac{1}{3}n(n+1)(n+2)$

Question 2: 12 points (4 + 4 + 4)

12

For each of the following recurrences, derive an expression for the running time using iterative, substitution or Master Theorem.

a. Consider the following recurrence algorithm **[Use Master Theorem – See LAST Page]**

```
Procedure (Array A, int n) {
If (n == 0) return True;

   for i = 1...n {
       A[i] = A[i] + 1;
   }
   if (n>1) Procedure(A, n/2);
} // end procedure
```

2

$n+1$

$4(n$

$1 + T(n/2)$

$= 2 + n+1 + 4n + 1 + T(n/2)$

$= T(n/2) + 5n + 4$

i. (2 points) Write a recurrence equation for T(n)

Ans. $T(n/2) + 5n + 4$

ii. (2 points) Solve recurrence equation using Master's method i.e. give an expression for the runtime T(n).

Ans. $a = 1$
$b = 2$
$c = 5$
$k = 1$

a	b^k
1	2^1

$\Rightarrow 1 < 2$, hence $T(n) = \Theta(n^k)$

b. **Use Iterative method to solve the following recirsion:**

T(n) = 3T(n-1) + 1, T(1) = 0

Ans. $T(n) = 3T(n-1) + 1$

$= 3[3T(n-2) + 1] + 1$

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

$= 3^2[3T(n-3) + 1] + 3 + 1$

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

$T(k) = 3^k T(n-k) + 3^{k-1} + \dots + 3^2 + 3 + 1$

if $k = n-1$, $T(n-(n-1)) = T(1) = 0$

$= 3^k T(n-k) + \sum_{i=0}^{k-1} 3^i$ // from geometric series $\sum_{i=0}^{k} r^i = \dots$

$= 0 + \frac{3^{k+1} - 1}{3-1}$ // $n = k+1$

$T(n) = 3^n - 1$

Maharishi International University (MIU)

MIDTERM

Course Title and Code: *CS 435 - Design and Analysis of Algorithms*
Instructor: *Dr. Emdad Khan*
Date: *Friday 07/29/2022*
Duration: *10am – 12:30 pm*

Student Name:

Amanuel Hadgu Tareke

Student ID:

614752

Total Mark
~~38~~ → 39 +1 = 40 → 40.5
46

1. This is a closed book exam. Do not use any notes or books!
2. Show your work. Partial credit will be given. Grading will be based on correctness, clarity and neatness.
3. We suggest you to read the whole exam before beginning to work on any problem.
4. There are 4 questions worth a total of 46 points, on 8 pages (including this one)
5. **You can use all back pages for scratch paper (may use for answers if you have used up the designated space for answer).**
6. **You can use a basic calculator (No smart phone unless network is disabled).**

8

Question 4: 12 points (3 + 5 + 4)

a. Is Mathematics Consistent? Explain your answer with an example.

Ans. Mathematics is not consistent.
Hence mathematics can not proof all problems as both true and false. eg:

$$2^0 = 1$$
$$2^1 = 2$$
$$2^2 = 4$$
$$\vdots$$
$$2^\infty = \text{undefined}$$

b.

(i) Why do we need to use "Expected Value" in computing running time in Bucket sort?

Ans. We need to use "Expected Value" in computing running time in Bucket Sort, becuase the number of Buckets depends on the distribution of our data. Take array with values [1, 999, 3, 10, 12, 7, 6, 502, 103, 409, 70, 303, 55]
= our Bucket number is $n^1 < 5 < n^k = 10^0 < 5\ 10^3$.
In the above array sorting we will need three buckets of size 10.

(ii) Why Quicksort, in general performs better than Mergesort. Explain with an example using solutions for running time T(n) for both methods.

Ans. Quicksort generaly performs better than Mergesort for the following reason:
i) Quicksort is in-place, but Mergesort is not in place. This mergesort's use of extra space adds overhead to the operation which in turn slows the sorting.
ii) The worst case of Quicksort does NOT occur often. This overcomes the weak side of Quicksort. Hence, the worst case run time of Quicksort $(O(n^2))$ is not of much effect.
iii) Mergesort $(T(n) = n \log n)$, Quicksort $(T(n) = n \log n$ // average most expected

-1

c. Give a Big O estimate for

$(2^n + n^2)(n^3 + 3^n)$

Clearly show all steps.

Ans. $(2^n + n^2)(n^3 + 3^n)$

$= 2^n \cdot n^3 + 2^n \cdot 3^n + n^2 \cdot n^3 + n^2 \cdot 3^n$

$= 2^n \cdot n^3 + 6^n + n^5 + n^2 \cdot 3^n$

$= O(2^n \cdot n^3 + 6^n + n^5 + n^2 \cdot 3^n)$

$= \max(O(2^n \cdot n^3), O(6^n), O(n^5), O(n^2 \cdot 3^n))$

$= \underline{O(6^n)}$

$2^2 \cdot 3^2 = 6^2$
$4 \times 9 = 36$
$= 36$
$2^3 \cdot 3^3$
$= 8 \cdot 27 = 216$

$2^{3+1} = 2^4$

c. **Use basic concepts of "Inversion Bound" and prove that all inversion bounds sorting algorithms cannot do better than O(n^2).**

Ans. All inversion bound sorting algorithms can not do better than O(n^2)

Proof : The key operation in these algorithms is comparison. In comparing we will always need nested loop whose running time is quadratic. So whenever we use these algorithms we will have O(n^2) running time.

Not a complete proof.

Need to use n_{C_2} based inversion.

-2

Masters Formulae

For recurrences that arise from Divide-And-Conquer algorithms (like Binary Search), there is a general formula that can be used.

Theorem. Suppose $T(n)$ satisfies

$$T(n) = \begin{cases} d & \text{if } n = 1 \\ aT(\lceil \frac{n}{b} \rceil) + cn^k & \text{otherwise} \end{cases}$$

where k is a nonnegative integer and a, b, c, d are constants with $a > 0, b > 1, c > 0, d \geq 0$. Then

$$T(n) = \begin{cases} \Theta(n^k) & \text{if } a < b^k \\ \Theta(n^k \log n) & \text{if } a = b^k \\ \Theta(n^{\log_b a}) & \text{if } a > b^k \end{cases}$$

0,2,4,5,8

Question 3 (contd): 11 points (4 + 3 + 4)

c. Use the QuickSelect algorithm to manually compute the 4-th smallest element of the array [2, 5, 23, 0, 8, 4, 33, 60, 30]. Assume that the rightmost element is used as the pivot in each case. Show what happens in each self-call, indicating the new input array and the current value of *k*.

Ans. $A = [2,5,23,0,8,4,33,60,\underline{30}]$

$k = 4$, $E = [30]$

E: [30]

L: [2,5,23,0,8,$\underline{4}$]   G: [33,60]

$|L| + |E| > k$, then take L

E: [4]

L: [2,0]   G: [5,23,$\underline{8}$]

$|L| + |E| < k$, then take G, $k' = k - |L| - |E|$
$= 4 - 2 - 1$
$= 1$

E: [8]

L: [$\underline{5}$]   G: [23]

$|L| + |E| < k$, then take L

E: [5]

L: [ ]   G: [ ]

hence, $|L| < k \le |L| + |E|$

return 5