CIS 275 Discrete Math I Summer 2019

Correction of the Quiz 03

Name:

Time: Complete and submit to the instructor

Evaluation:

- As described in the syllabus, the Quiz is 20% of the overall grade.

Exercise 1: Find the complexity of the below program:

```
(A)

function(int n)
{
    if (n==1)
       return;
    for (int i=1; i<=n; i++)
    {
        for (int j=1; j<=n; j++)
        {
            printf("*");
            break;
        }
    }
```

Solution:

(A) Time Complexity of the above function O(n). Even though the inner loop is bounded by n, but due to break statement it is executing only once.

Exercise 2:

Are each of the following true or false?

a) (a) 3 n^2 + 10 n log n = O(n log n)

b) (b) 3 n^2 + 10 n log n = Omega(n^2)

c) (c) 3 n^2 + 10 n log n = Theta(n^2)

d) (d) n log n + n/2 = O(n)

e) (e) 10 SQRT(n) + log n = O(n)

f) (f) SQRT(n) + log n = O(log n)

g) (g) SQRT(n) + log n = Theta(log n)

b) **Solution:**

c) (a) **False**, since n^2 (the dominate term on the left) is asymptotically faster growing than n log n and hence not upperbounded by it.
d) (b,c) **True**, since n^2 (the dominate term on the left) asymptotically grows like n^2 and hence it is Omega(n^2) and also Theta(n^2). faster growing than n log n and hence not upperbounded by it.
e) (d) **False** since n log n (the dominate term on the left) is not asymptotically upperbounded by n.
f) (e) **True**, since the dominate term on the left, 10 SQRT(n), is asymptotically upperbounded by n.
g) (f,g) **False**, since the dominate term on the left, SQRT(n), is not asymptotically upperbounded by n.

Exercise 1 : Find a theta notation for each expression

a) $6n + 1 = \Theta(n)$
b) $3n^2 + 2n \lg n = \Theta(n^2)$
c) $2 + 4 + 6 + \ldots + 2n = \Theta(n^2)$
d) $2 + 4 + 8 + 16 + \ldots + 2^n = \Theta(n^{n+1})$

Exercise 2 : Find a theta notation for the number of times the statement x = x +1 is executed

a) or i = 1 to 2n

x = x + 1

$\Theta(n)$

b) for i = 1 to 2n

for j = 1 to n

x = x + 1

$\Theta(n^2)$

c) for i = 1 to n

for j = 1 to n

for k = 1 to n

x = x + 1

$\Theta(n^3)$

Exercise 3: Find the greatest common divisor of each pair of integers:

a) 20, 40

The positive divisors of 20 are 1, 2, 4, 5, 10, 20.

The positive divisors of 40 are 1, 2, 4, 5, 8, 10, 20, 40

gcd(20, 40) = 20

b) $3^2 \times 7^3, 3^2 \times 7^3$

A divisor of $3^2 \times 7^3 \times 11$ would be any integer n such that another unique integer m can be found with $nm = 3^2 \times 7^3 \times 11$.

$\gcd(3^2 \times 7^3 \times 11, 3^2 \times 7^3 \times 11) = 3^2 \times 7^3 \times 11$

c) 0, 17

A divisor of zero would be any integer n such that another unique integer m can be found with nm = 0. For example, n can be 1 or 17.

The positive divisors of 17 are 1, 17.

gcd(0, 17) = 17

Exercise 4 : Find the least common multiple of each pair of integers :

a) 20, 40

The prime factorization of 20 = $\mathbf{2^2 \times 5}$

The prime factorization of 40 = $2 \times \mathbf{2^2 \times 5}$

$\mathbf{lcm(20, 40) = 2^2 \times 2 \times 5 = 40}$

b) 5, 25

The prime factorization of 5 = 1×5.

The prime factorization of 25 = 5×5.

$\mathbf{lcm(5, 25) = 1 \times 5 \times 5 = 25}$

Exercise 5: Express each binary number in decimal.

a) 100000

$0 \times \mathbf{2^0} = 0$

$0 \times \mathbf{2^1} = 0$

$0 \times \mathbf{2^2} = 0$

$0 \times \mathbf{2^3} = 0$

$0 \times \mathbf{2^4} = 0$

$1 \times \mathbf{2^5} = 32$

$0 + 0 + 0 + 0 + 0 + 32 = \mathbf{32}$

b) 1001

$1 \times \mathbf{2^0} = 1$

$0 \times \mathbf{2^1} = 0$

$0 \times \mathbf{2^2} = 0$

$1 \times \mathbf{2^3} = 8$

$$1 + 0 + 0 + 8 = \mathbf{9}$$

<u>Exercise 6:</u> Express each decimal number in binary.

a) 43

The computation shows that the successive divisions by 2 with the remainders recorded at the right

2) 43	quotient = 21	remainder = 1	1's bit
2) 21	quotient = 10	remainder = 1	2's bit
2) 10	quotient = 5	remainder = 0	4's bit
2) 5	quotient = 2	remainder = 1	8's bit
2) 2	quotient = 1	remainder = 0	16's bit
2) 1	quotient = 0	remainder = 1	32's bit
0			

Binary number = 101011

<u>Exercise 7:</u> Add the binary numbers.

a) 101101 + 11011

		1	11	carry in
1001	1001	1001	1001	
+ 1111	+ 1111	+ 1111	+ 1111	
	0	000	11000	

<u>Exercise 8:</u> Express the hexadecimal number in decimal.

a) B4F

11*16 *16 + 4 * 16 + 15 * 1

<u>Exercise 9:</u> Add the hexadecimal numbers.

b) F0BA + E9AD

A + D = 10 + 13 = 23 = 16 + 7 = 17 base 16

1 + B + A = 1 + 11 + 10 = 22 = 16 + 6 = 16 base 16

1 + 0 + 9 = 10 = A

F + E = 15 + 14 = 29 = 16 + 13 = 1D base 16

1DA67

Formula

ANALYSIS OF ALGORITHMS

Analysis an algorithm refers to the process of deriving estimates for the time and space needed to execute the algorithm.

- To derive a theta notation, you must derive both big oh and omega notation.
- An other way to derive big oh, omega and theta estimations is to use known results:

Expression	Name	Estimate
$a_k n^k + a_{k-1} n^{k-1} + ,,, + a_1 n + a_0$	Polynomial	$\mathbf{\Theta(n^k)}$
$1 + 2 + \ldots + n$	Arithmetic Sum (Case k = 1 for Next entry)	$\mathbf{\Theta(n^2)}$
$1^k + 2^k + \ldots + n^k$	Sum of Powers	$\mathbf{\Theta(n^{k+1})}$
lg n!	Log n Factorial	$\mathbf{\Theta(n \lg n)}$

Definition: Let f and g be functions with domain {1, 2, 3, …}

We write $f(n) = O(g(n))$

We define f(n) is of order at most g(n) or f(n) is big oh of g(n) if there exists a positive constant C1 such that

$$|f(n)| \leq C1\,|g(n)|$$

For all but finitely many positive integers n.

We write $f(n) = \Omega(g(n))$

We define f(n) is of order at least g(n) or f(n) is omega of g(n) if there exists a positive constant C2 such that

$$|f(n)| \geq C2\,|g(n)|$$

For all but finitely many positive integers n.

We write $f(n) = \Theta(g(n))$

We define f(n) is of order g(n) or f(n) is theta of g(n) if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

- ***By replacing O by Ω and "at most" by "at least", we obtain the definition of what it means for the best-case, worst-case, or average-case time of an algorithm to be of order at least g(n). If the best-case time required by an algorithm is O(g(n)) and Ω(g(n)), we say that the best-case time required by an algorithm is Θ(g(n)) .***

Introduction to number of theory

- Let m and n be integers with n and m different to zero. A *common divisor* of m and n is an integer that divides both m and n. **The *greatest common divisor***, written **gcd(m, n)**

- Let m and n be positive integers. A *common multiple* of m and n is an integer that divides by both m and n. **The *least common divisor***, written **lcm(m, n)**

` lcd(m, n) is the smallest common multiple of m and n.

- ***The decimal number system****:* it represents integers using 10 symbols.

- ***The binary number system****:* it represents integers using bits (a bit is a binary digit, that is a 0 or a 1).

- ***The hexadecimal number system****:* it represents integers using 16 symbols

- The ***Euclidean algorithm*** is an odd, famous, and efficient algorithm for finding the greatest common divisor of two integers, The Euclidean algorithm is based on the fact that if $r = a \bmod b$, then **gcd(a, b) = gcd(b, r)**

Hexadecimal to Decimal Conversion Chart

Hexadecimal	Decimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15