Homework 2
CS601: Advanced Algorithms

Instructions: You may if you prefer solve the problem with one partner, but you need to make sure either one of you, if being asked, explain details of your solutions. Clearly write down your name(s). Clearly write down your answers! If you don't want to use latex or word to write your answer, you can hand write your solution, and scan or take pictures to submit, make sure it's clear. You don't need to submit for the optional question if there is any, that's just for you to practice.

Problem 1. [Category: Frequency of operations] Count the precise number of "fundamental/basic operations" executed in the code below. Your answer should be a function of n in closed form. Note that "closed form" means that you must resolve all Σ's, recursive relations, etc. An asymptotic answer (such as one that uses $O()$, Θ, and similar) is not acceptable. Justify your solution and show all your work. (credits to Algorithms Capstone Exams). **[10 points]**

```
for (int i = 1; i < n; i++)
{
    Perform 1 fundamental/basic operation;
    for (int j = i; j <= n; j++)
        Perform 1 fundamental/basic operation;
    //endfor j
}//endfor i
```

Problem 2. [Category: Asymptotic growth] Consider the following 15 functions of n (recall that log denotes the binary logarithm):

n	$n\ln(n)$	$n\sqrt{n}$	n^2	n^3
$(1.1)^n$	2^n	3^n	$n!$	n^n
$n\sqrt[3]{n}$	$n\log n$	$n^{\log(n)}$	2^{n+3}	4000

Arrange these functions in non-decreasing order of growth; that is, find an arrangement $f_1 \preceq f_2 \preceq ... \preceq f_{15}$ of the functions such that $f_1 \in O(f_2), f_2 \in O(f_3), ...f_{14} \in O(f15)$. For each i, you should write $f_i \equiv f_{i+1}$ if $f_i \in \Theta(f_{i+1})$, but $f_i \prec f_{i+1}$ otherwise. You do not need to argue for your answers. **[5 points]**

Problem 3. [Category: Asymptotic growth] Prove, using the definitions of big-O and Θ (and not any result from the textbook(s) or the slides) that if $f \in \Theta(n\log n)$ and $g \in O(n)$ (with f, g functions from non-negative integers to non-negative reals) then $f + g \in \Theta(n\log n)$. **[5 points]**

Problem 4. [Category: Analysis] Estimate the running times, in terms of n, of the following three programs. You should give tight bounds, of the form $\Theta(f(n))$ with f as simplified as possible. Justify your answers. (You can assume that arithmetic operations take time in $\Theta(1)$, no matter the size of the operands. Recall that the scope of language constructs is determined by indentation, so in (c), $j \leftarrow 2 * j$ is part of the while loop but not of the for loop.) **[15 points]**

(a)

```
z ← 0
for j ← 1 to n
    q ← 0
    s ← j * j
    while q ≤ s
        q ← 5 + q
        z ← z + q
```

(b)

```
z ← 0
for j ← 1 to 4 * n
    q ← 1
    s ← j * j
    while q ≤ s
        q ← 5 * q
        z ← z + q
```

(c)

```
z ← 0
j ← 1
while j < n
    for q ← 1 to j
        z ← z + q
    j ← 2 * j
```