Seoul National University

M1522.000900 Data Structure

Spring 2025, Kang

Homework 1: Mathematical Preliminaries & Algorithm Analysis

(Chapters 2 & 3)

Due: **23:59, April 6th (Sunday), 2025**

Reminders

- Lead TA: Jaehyeon Choi (snuds.ta@gmail.com)
- The points of this homework add up to 100.
- All assignments must be done individually.
- **Type your answer in English** and submit your assignment **in PDF format**.
  - Answer written in Korean may get 0 points.
  - You will get a 10% deduction on your grade for this homework if your submission is either hand-written or non-PDF format.
- Name your file as "(studentID)-(name)-HW1.pdf".
  (e.g., *202512345-GildongHong-HW1.pdf*)
- Whenever you are making an assumption, state it clearly.
- If you have any questions about the assignment, post them on eTL.

Submission

- Submit your assignment to **eTL**.
- The submission after the due date will be regarded as a late **submission, even if it is only one second late.** Late submissions are accepted **within only one week after the due date.**
- You do not need to specify whether to use the slip-days; they are automatically used.

Question 1 [10 points]

Consider a set $\boldsymbol{A}$ of people, and the following two relations R_1 and R_2 defined on $\boldsymbol{A}$:

- R_1: For $a, b \in \boldsymbol{A}$, $(a, b) \in R_1$ if and only if a is a senior to b.
- R_2: For $a, b \in \boldsymbol{A}$, $(a, b) \in R_2$ if and only if a and b are in the same department.

For each relation below, answer whether the relation does or does not satisfy each of the properties reflexive, symmetric, antisymmetric, and transitive.

(1) Relation R_1. [2.5 points]

(2) Relation R_2. [2.5 points]

(3) Relation $R_1 \cap R_2$. [2.5 points]

(4) Relation $R_1 \cup R_2$. [2.5 points]

Question 2 [10 points]

Prove the following statements by contradiction.

(1) A triangle ABC in a two-dimensional plane cannot have more than one right angle. [2 points]

(2) $\sqrt{3}$ is irrational. [2 points]

(3) There are infinitely many prime numbers. [2 points]

(4) For $a, b, c \in \mathbb{N}$, if $a^2 + b^2 = c^2$, then both a and b cannot be odd numbers. [2 points]

(5) Given a hash function $h: \boldsymbol{X} \to \boldsymbol{Y}$, if $|\boldsymbol{X}| > |\boldsymbol{Y}|$, then 'hash collision' occurs. (i.e., $\exists x_1, x_2 \in \boldsymbol{X},\ x_1 \neq x_2,\ h(x_1) = h(x_2)$). [2 points]

Question 3 [15 points]

The Tower of Hanoi is a game where the goal is to move all the disks from the leftmost pole to the rightmost pole, as shown in Figure 1. You can move one disk at a time to either of other two poles, and a disk can only be placed on an empty pole or on a larger disk. Let $T(n)$ be the minimum number of disk moves required to finish the game with n disks. Answer the following questions:

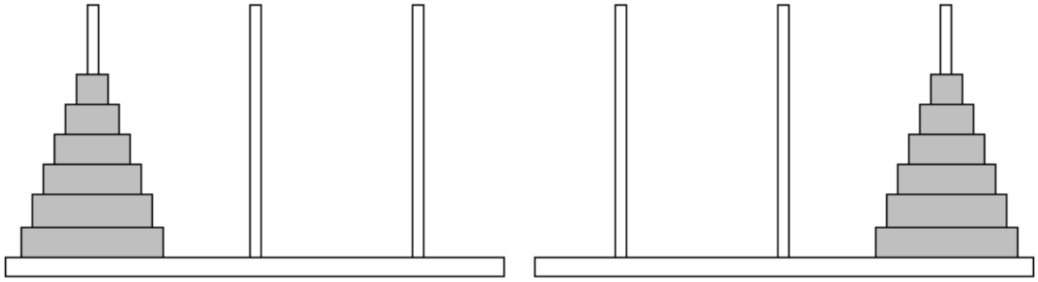

Figure 1. Tower of Hanoi game with 6 disks. The left figure shows the initial state, and the right figure shows the final state after moving all disks.

(1) Determine $T(10)$. [5 points]

(2) Express $T(n)$ as a recurrence relation. [5 points]

(3) Find a closed-form solution for $T(n)$. [5 points]

Question 4 [15 points]

Find the closed-form solution for $C(n)$, and prove it through mathematical induction by answering the following questions:

$$C(n) = \sum_{i=0}^{n-1} C(i)C(n-1-i), \qquad \text{for } n \geq 1, \qquad \text{with } C(0) = 1$$

(1) Derive the closed-form solution for $C(n)$. [5 points]

(2) **Basis:** Show that $C(n)$ holds for $n = 1$. [5 points]

(3) **Inductive step**: Assume that the closed-form solution holds for $n = k$. Use this assumption to prove that the solution also holds for $n = k + 1$. [5 points]

Question 5 [10 points]

The expressions below represent the time complexity of four different algorithms. For each expression, state the range of values of n for which it is the most efficient. Assume that n is in the range $[2, \infty)$. For example, given two expressions n^2 and $2n$, the former is more efficient for $n \in [1,2]$, while the latter is more efficient otherwise.

Use the following information:

- $\frac{n!}{5} = 1.1^n$ when $n = 3.1$
- $\frac{n!}{5} = 10 \log_2(\log_2 n)$ when $n = 2.04,\ 4.55$
- $\frac{n!}{5} = n^{\frac{2}{3}}$ when $n = 3.5$
- $1.1^n = 10 \log_2(\log_2 n)$ when $n = 2.1,\ 33.06$
- $1.1^n = n^{\frac{2}{3}}$ when $n = 1.2,\ 21.4$
- $10 \log_2(\log_2 n) = n^{\frac{2}{3}}$ when $n = 2.18,\ 152.89$

Expressions to Compare:

$\frac{n!}{5}$ $\quad$ 1.1^n $\quad$ $10 \log_2(\log_2 n)$ $\quad$ $n^{\frac{2}{3}}$

Question 6 [10 points]

Write down the time complexity of each code snippet in Big-Theta notation.

(1) [2.5 points]

```
void fun (int n) {
    int i=0, s=0;
    while (s<n) {
        ++i;
        s=s+i;
    }
}
```

(2) [2.5 points]

```
for (i=0; i<n; i++) {
    s++;
}
for (j=0; j<m; j++) {
    s++;
}
```

(3) [2.5 points]

```
for (i=0; i<n; i++) {
    for (j=n; j>i; j--) {
        s++;
    }
}
for (k=0; k<n; k++) {
    s++;
}
```

(4) [2.5 points]

```
int fun (int n) {
    if (n==0)
        return 1;
    else
        return (n + fun(n-1));
}
```

Question 7 [15 points]

Consider two programs A and B, with the following recurrence relations for their running time $T_A(n)$ and $T_B(n)$, where n is the input size. Assume that n is sufficiently large and $T_A(1) = T_B(1) = c$ for some positive constant c. Answer the following questions:

- Program A: $T_A(n) = 2T_A(n/2) + n^2$
- Program B: $T_B(n) = 3T_B(n/3) + n$

(1) Regarding Big-Oh notation, does program B improve in time complexity over program A? Justify your answer. [7.5 points]

(2) Programs A and B each run on machines X and Y, respectively, processing n inputs in exactly 1 hour. How many times faster is machine X compared to machine Y? Express your answer in terms of n. [7.5 points]

Question 8 [15 points]

A power function is a function of the form $f(x, n) = x^n$. The figures below show an implementation of the algorithm in Java. Answer the following questions.

(1) What is the time complexity of the power function in asymptotic terms in Figure 2? Express your answer in terms of n. [5 points]

```
public static long powerN(long x, int n) {
    if (n==0) return 1;
    return x*powerN(x, n-1);
}
```

Figure 2. Java implementation of a power function.

(2) If we modify the implementation of the power function from Figure 2 to Figure 3, what will its time complexity be in asymptotic terms? Express your answer in terms of n. [10 points]

```
public static long powerN(long x, int n) {
    if (n==0) return 1;
    if (n%2==0) {
        long a = powerN(x, n/2);
        return a*a;
    }
    else {
        long a = powerN(x, (n-1)/2);
        return x*a*a;
    }
}
```

Figure 3. Modified Java implementation of a power function.