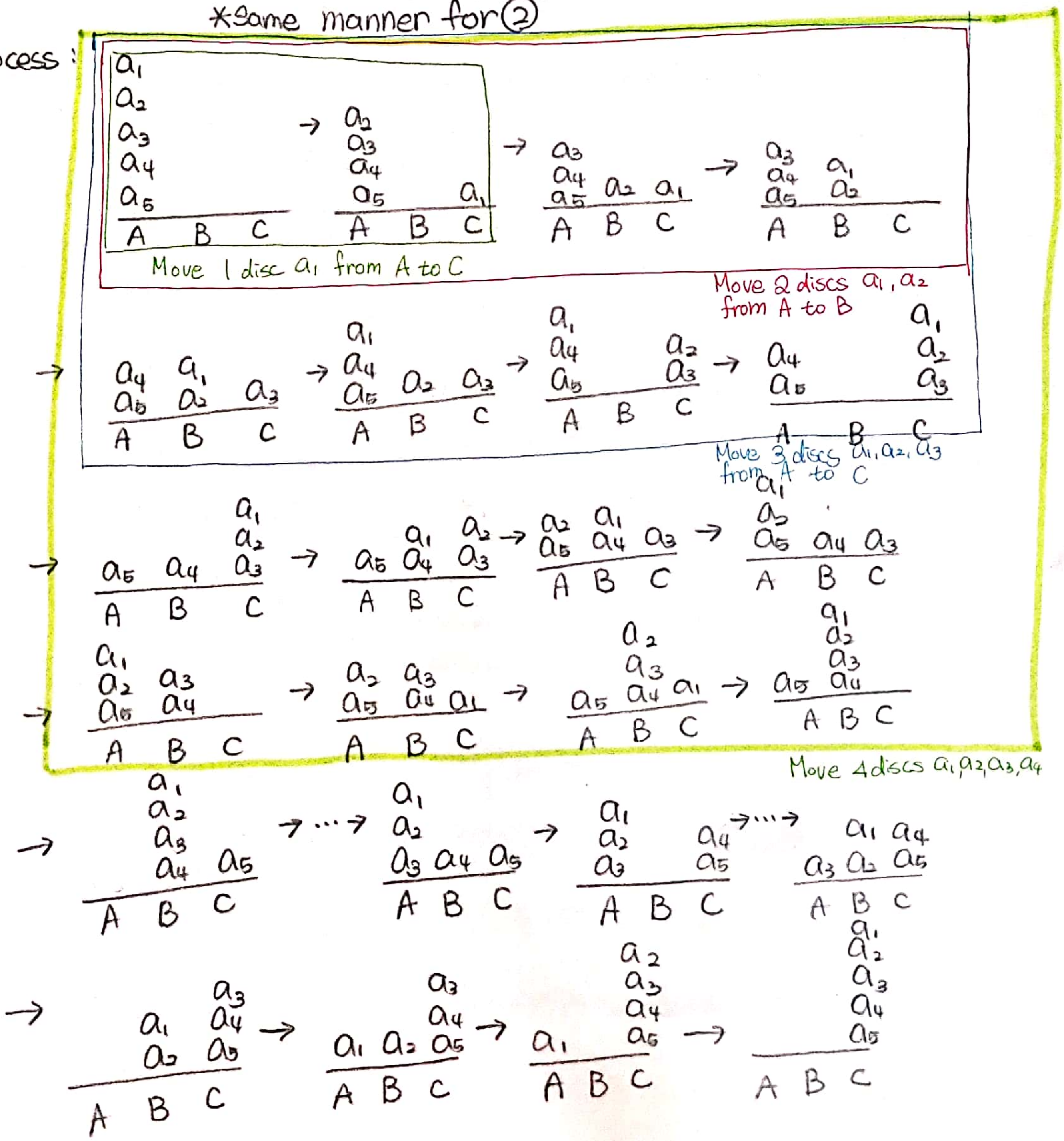
20171035
Lee Somin

Data Structure
Assignment 01

1. Hanoi Tower when 5 discs are used.
   let the rods be rod A, B, C and discs be a_1, a_2, a_3, a_4, a_5.
   Problem: move 5 discs a_1, a_2, a_3, a_4, a_5 from A to C
   Sub Problems: ① Move 4 discs from A to B
   > Move a_5 from A to C
   ② Move 4 discs from B to C
   ↳ sub problems of ①, ②: ①-1 Move 3 discs from A to C
   > Move a_4 from A to B
   ①-2 Move 3 discs from C to B
   *Same manner for ②

Process:



비고

```
Hanoi (n, A, B, C)
powern [n];
discMovement [n]; int disc;
for i=0 to n-1 {
    if i==0          <discMovement[i]=0;
        power[i]=2;
    else
        power[i]=power[i-1]*2 }
for k=1 to power[n-1]
  { for j=0 to n-1
      { if ((k+power[j-1]/2)/power[j]==0
            { disc=j; break;} }
    if n-disc %2==1
        switch discMovement[disc]%3
            case 0 : move A→B; break;
            case 1 : move B→C; break;
            case 2 : move C→A; break;
    else
        switch discMovement [disc] %3
            case 0 : move A→C; break;
            case 1 : move C→B; break;
            case 2 : move B→A; break;
    discMovement[disc]++;
  }
```

MEMO

2. Technical Report

- for Hanoi (n, A, B, C)

Time Complexity: $T(n) = c_1 n + c_2 2^n \cdot n$
$= \Theta(n \cdot 2^n)$

Input:

n	number of discs.
A	rod where discs start from
B	temporary rod.
C	rod where discs should end

variable:
power[n] = array with 2^{i+1} stored
discMovement[n] = number of movements made on ithdisc
int disc = disc number that shoubl be moved on the step.

code explanation:

line#	Description
3 - 7	initialize power[] with 2^{i+1} and discMovement[] with 0
9 - 11	find & Assign the disc# that should be moved on the step. ex) kth disc should be moved on step $2^k x - 2^{k-1}$. $(x \in \mathbb{N})$
12 - 21	find where from and to move the disc according to the pattern. ex) if n-disc % 2 == 0, disc moves in order of A→C→B→A→… else, (n-disc%2 == 1) disc moves in order of A→B→C→A→…
22	count up the movement of the disc.

Principle: I analyzed which disc moves on specific step, and found out that kth disc is moved on the step $2^k x - 2^{k-1}$ $(x \in \mathbb{N})$ when this problem is solved recursively.
Then, I found out that kth disc start with movement A→C when n-k is even, and A→C when n-k is odd.
Lastly, the disc that started with A→C moves in order of A→C→B→A→C→B, … and the other A→B→C→A→B→C …

3. $T(n) = T(\frac{n}{2}) + c$

let's say $n = 2^k$ then,

$$T(n) = T(\tfrac{n}{2}) + c$$
$$= T(\tfrac{n}{2}\cdot\tfrac{1}{2}) + c + c$$
$$\vdots$$
$$= T(1) + c \cdot k$$

k can be written as $\log_2 n$

Therefore,

$$T(n) = c \cdot \log_2 n + c_1$$
$$= \Theta(\log_2 n)$$

$\therefore$ Time Complexity of $T(n) = \Theta(\log_2 n)$.

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

let's say $n = 2^k$ then,

$$T(n) = 2T(\tfrac{n}{2}) + c$$
$$= 2\left(2T(\tfrac{n}{2}\cdot\tfrac{1}{2}) + c\right) + c = 2\cdot 2T(\tfrac{n}{2}\cdot\tfrac{1}{2}) + 2c + c$$
$$\vdots$$
$$= 2^k T(1) + c\sum_{i=1}^{k} 2^{i-1} = 2^k T(1) + c \cdot 2^k$$
$$= (T(1) + c)\, 2^k$$

k can be written as $\log_2 n$.

Therefore,

$$T(n) = (T(1) + c)\, 2^{\log_2 n} = (T(1) + c)\cdot n$$
$$= \Theta(n)$$

$\therefore$ Time complexity of $T(n) = \Theta(n)$.