Handwritten

1. (1) $T(n) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} 1 = \sum_{j=2}^{n} 1 + \sum_{j=3}^{n} 1 + \cdots + \sum_{j=n}^{n} 1 = (n-1) + (n-2) + \cdots + 1$

$= \frac{[1+(n-1)](n-1)}{2} = \frac{n(n-1)}{2}$.

Let $g(n) = n^2$, there exists $c_1 = \frac{1}{5}$ and $c_2 = \frac{1}{2}$ such that for all $n > (n_0 = 2)$

$c_1 g(n) \leq T(n) = \frac{n(n-1)}{2} \leq c_2 g(n)$.

Therefore, $T(n) \in \Theta(n^2)$ #

(2) $T(n) = \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{m-1} 1 = n + m$. $g(n) = n+m$, $c_1 = 0.9$, $c_2 = 1.1$, $n_0 = 1$; $m_0 = 1$.

$T(n) \in \Theta(n+m)$ #

(3) a) $T(-1) = 1$

$T(0) = 1$

$T(1) = T(0) + T(-1) = 2$

$T(2) = T(1) + T(0) = 2 + 1 = 3$

$T(3) = T(2) + T(1) = 3 + 2 = 5$

$\vdots$

b) Define function $F(x)$ where $x \in \mathbb{N}^0$: $\begin{cases} F = 1, & \text{where } x = 0 \text{ or } 1 \\ F(x) = F(x-1) + F(x-2), & \text{where } x \geq 2. \end{cases}$

Then $T(n) = F(n+1)$.

c) Finding the ratio $\frac{F(x+1)}{F(x)} = G_x$

$G_x = \frac{F(x+1)}{F(x)} = \frac{F(x) + F(x-1)}{F(x)} = 1 + \frac{1}{\left(\frac{F(x)}{F(x-1)}\right)} = 1 + \frac{1}{G_{x-1}}$

For sufficient large x, $G_{x-1} \doteqdot G_x$. Solving $G_x = 1 + \frac{1}{G_x}$ gives positive root $G_x = \frac{1+\sqrt{5}}{2}$.

d) Therefore, for sufficient large x, $F(x) \doteqdot G_x^{\,x} \cdot 1 = \left(\frac{1+\sqrt{5}}{2}\right)^x \doteqdot 1.618^x$

e) $T(n) = F(n+1) \doteqdot 1.618^{n+1} \in \Theta(1.618^n)$ # $(g(n) = 1.618^n,\ c_1 = 1,\ c_2 = 2,\ n_0 = 1)$

1.(4) $j(n) = \underline{2j(\frac{n}{2})} + \underline{n} \longrightarrow$ adding 1 to total for n times in every iteration.

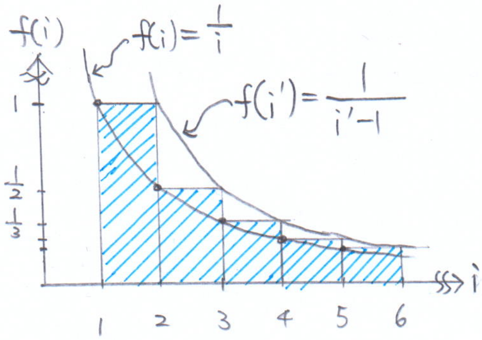
$\quad\quad\quad\quad\quad\longrightarrow$ calling $j(\frac{n}{2})$ twice in every iteration.

Analyzing by master theorem, $a=2$, $b=2$, $f(n)=n$.

$n^{\log_b a} = n^{\log_2 2} = n = f(n) \rightarrow$ case 2

Therefore, $\underline{T(n) \in \Theta(f(n)\log n) = \Theta(n\log n)}$ #

(5) $T(n) = \sum_{i=1}^{n} \sum_{j=1}^{floor(n/i)} 1 = \sum_{i=1}^{n} floor(\frac{n}{i}) \doteq \sum_{i=1}^{n} \frac{n}{i}$ where n is sufficiently large.



By graph we know: $\int_1^n \frac{1}{i}\,di < \sum_{i=1}^{n} \frac{1}{i} < 1\times 1 + \int_2^n \frac{1}{i-1}\,di$

$\Rightarrow \ln n < \sum_{i=1}^{n} \frac{1}{i} < 1 + \ln(n-1)$

Since $\ln n \in \Theta(\ln n)$ and $1+\ln(n-1) \in \Theta(\ln n)$, $\sum_{i=1}^{n} \frac{1}{i} \in \Theta(\ln n)$

$\underline{T(n) \doteq \sum_{i=1}^{n} \frac{n}{i} = n \sum_{i=1}^{n} \frac{1}{i} \in \Theta(n \ln n)}$ #

2.(1) $T(n) = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} 2 = 2n^3 \in \underline{\Theta(n^3)}$ #

$\quad\quad\quad\quad\uparrow$ multiplication and addition.

(2) 矩陣乘法有結合律, AAAA = (AA)(AA) = A(AAA),

a) 將冪次(N−1)轉為二進制, [最大有值(為1)的位]即為[要乘的次數 a)加1]

b) 將矩陣相乘, 記錄結果, 再將矩陣相乘, 記錄結果, 直到達到 a 次, 可得最大乘冪矩陣 A^{2^a} 及較小乘冪之 $A^{2^{a-1}}$, $A^{2^{a-2}}$, … A^2.

c) 將(N−1)的二進制中為1的位數對應的相乘矩陣 A^{2^k} ($k \in N$) 相乘即為所求.

$\boxed{例}$: (N−1) = 372 (次方), 372 in binary is $\underset{8\ 7\ 6\ 5\ 4\ 3\ 2\ 1\ 0}{1,0111,0100}$, 共9位數 → a=8.

$\underset{1}{A\cdot A = A^2}$; $\underset{2}{A^2\cdot A^2 = A^4}$; $\underset{3}{A^4\cdot A^4 = A^8}$; $\underset{4,5,6,7}{\cdots}$; $\underset{8}{A^{128}\cdot A^{128} = A^{256}}$.

$A^{372} = A^{256}\cdot A^{64}\cdot A^{32}\cdot A^{16}\cdot A^4$ 算出 A^{372}, 其間共做 8+4=12 次矩陣乘法

↙ ex DEC 511 = BIN 1 1111 1111

可知每次都將問題對半切, 最差情況下要做 $\log_2(N-1)$次相乘組合小問題的結果, 即 $T(N) = T(\frac{N}{2}) + 2 \longleftarrow A^k\cdot A^k$ 的乘法, 和組合起來乘法各一次

由 master theorem, $\theta(N^{\log_b a}) = \theta(N^{\log_2 1}) = \theta(1) = \theta(f(N)) = \theta(2) = \theta(1)$. case 2

$\underline{\rightarrow T(N-1) = \theta(\log N)}$. #

3. (1) 若 $k \le n$，則將執行 k 次 while 迴圈，每次有 $Pop() \in O(1)$，則 $Grab(k) \in O(k)$.
若 $k > n$，〃 n 〃，則 $Grab(k) \in O(n)$

故，$Grab(k) \in O(\text{minimum}(k, n))$ #

(2) 由題目推測，若 stack.empty() 則無法執行 Pop()，亦不計入 operation 計算.

最佳情況：n 次呼叫皆成功執行 $O(1)$ 函數 (Push() 或 Pop())，又 $O(1)$ 不可更快、n 次呼叫不可省略 → $O(1)=\Theta(1)$; time comp. $= n \times \Theta(1) = \Theta(n)$.

最差情況：n 次呼叫有 i 次 Push()，有 $(n-i)$ 次 $Grab(k_j)$，k_j 為第 j 次呼叫的引數.
無論每次 $Grab(k_j)$ 的 k_j 有多大，其 time complexity $\in O(\min(k_j, n_j))$，
其中 n_j 為第 j 次呼叫 $Grab(k_j)$ 時 stack 中物件個數、
則 $(n-i)$ 次 $Grab(k_j)$ 呼叫加總之 time complexity $\in O(i)$
→ 最差 $i = n-1$，執行 $n-1$ 次 Push() 和 1 次 $Grab(k)$，$k \ge n-1$.
time complexity $\in O(1) \times i + O(i) = O(2i) = O(2n-2) \in O(n)$

由 $\Theta(n) \le \text{function}(n) \le O(n)$ 知 $\text{function}(n) \in \Theta(n)$. #