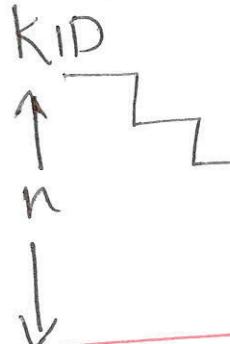


$$\boxed{\begin{aligned} T(n) &= T(n-1) + C \\ T(1) &= 1 \end{aligned}}$$

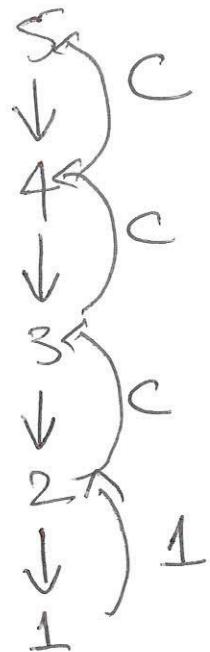
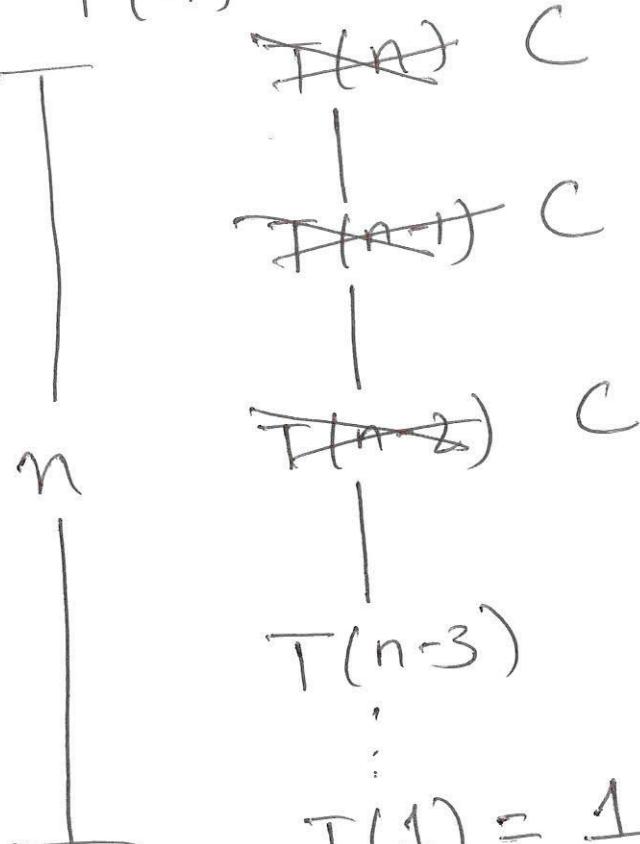


KID getting Candy

$$\boxed{\text{fib}(n) = n * \text{fib}(n-1)}$$

Finding Fibonacci of n

$$T(n) = T(n-1) + C$$



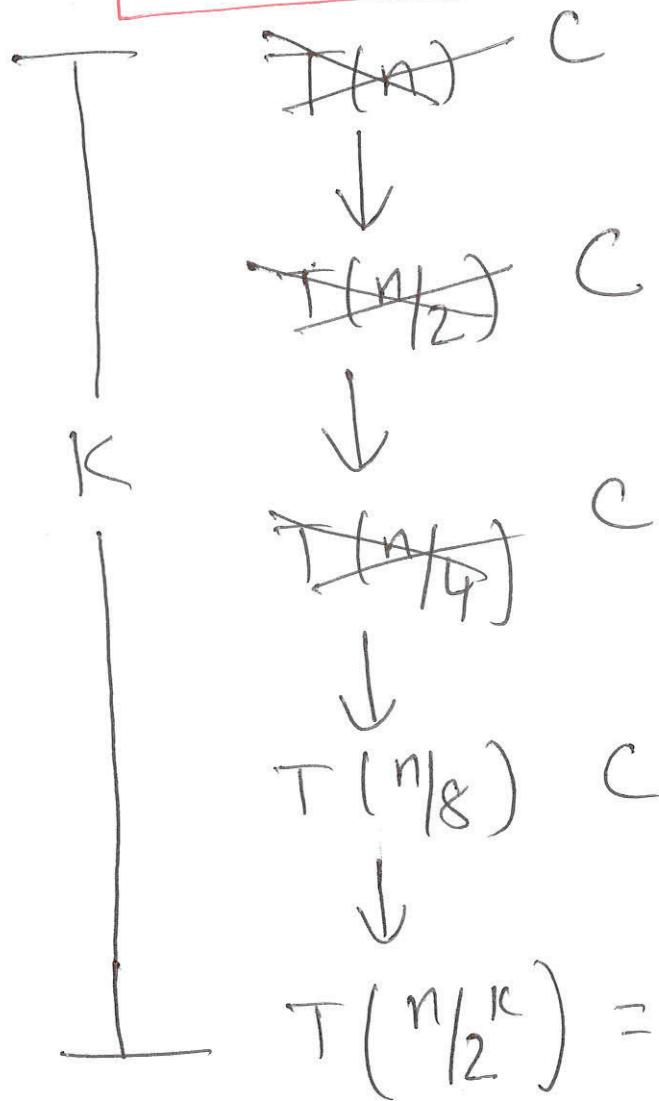
$$T(1) = 1$$

$$\begin{aligned} \text{Total Work} &= (\underbrace{C + C + C + C + \dots}_{n-1})^{n-1} + 1 \\ &= C(n-1)^{133} + 1 \\ &= Cn - C + 1 = \Theta(\cancel{C}n) \end{aligned}$$

$$T(n) = T(n/2) + C$$

$$T(1) = 1$$

FINDING A WORD IN DICTIONARY



16
↓
8
↓
4
↓
2

$$2^4 = 16$$

$$\boxed{\log_2 16 = 4}$$

$$\frac{n}{2^K} = 1 \quad \therefore n = 2^K \quad \boxed{K = \log_2 n}$$

$$\begin{aligned} \text{Total Work} &= K \cdot C \\ &= \log_2 n \cdot C = \Theta(\log_2 n) \end{aligned}$$

$$T(n) = 2T\left(\frac{n}{2}\right) + C$$

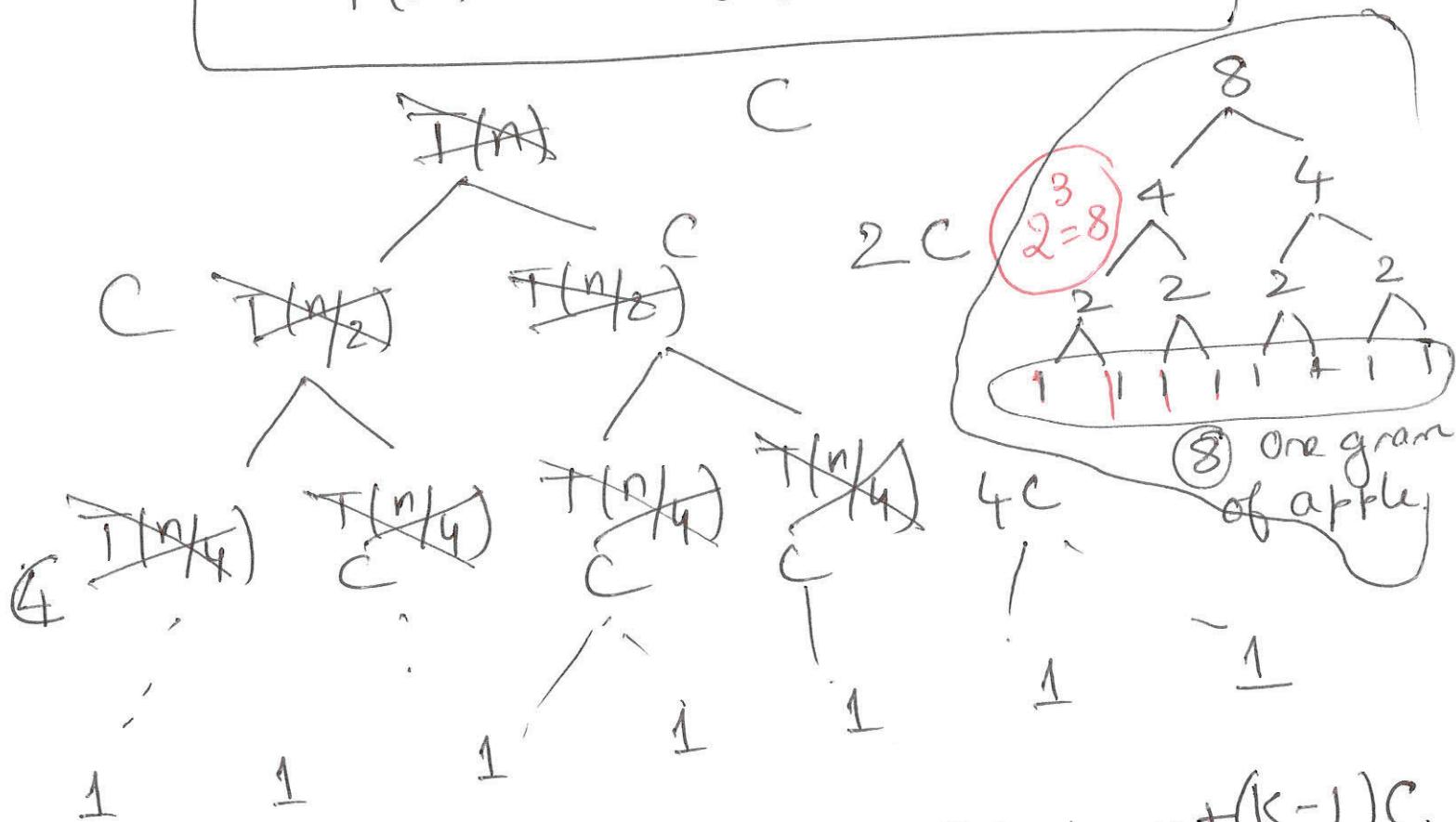
$$T(1) = 1$$

3

n grams of Apple.
cut apple by Half and eat-

$$T(n) \neq T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + C$$

$$T(n) = 2T\left(\frac{n}{2}\right) + C$$



$$\text{Total work } T(n) = C + 2C + 4C + 8C + \dots + (2^{k-1})C$$

$$T(n) = C \left(2^0 + 2^1 + 2^2 + \dots + 2^{k-1} \right)$$

$$= C (2^k - 1) = 2^k C$$

But $2^k = n$

$$T(n) = C(n+1) \\ = Cn = \Theta(n)$$

$$\begin{cases} T(n) = T(n/2) + n \\ T(1) = 1 \end{cases}$$

(4)

$$\begin{array}{ccc}
 T(n) & n & n/2^0 \\
 | & & \\
 T(n/2) & n/2 & n/2^1 \\
 | & & \\
 T(n/4) & n/4 & n/2^2 \\
 | & & \\
 \vdots & & \\
 T(1) = & n/2^K = 1 &
 \end{array}$$

$$\frac{n}{2^K} = 1$$

$$2^K$$

$$n = 2^K$$

$$K = \log_2 n$$

$$\text{Total work} = n \left(\frac{1}{2^0} + \frac{1}{2^1} + \frac{1}{2^2} + \frac{1}{2^3} + \dots + \frac{1}{2^K} \right)$$

$$= n \left(1 + \underbrace{0.5 + 0.25 + \dots}_{\frac{1}{136}} \overbrace{\frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{n} = 1} \right)$$

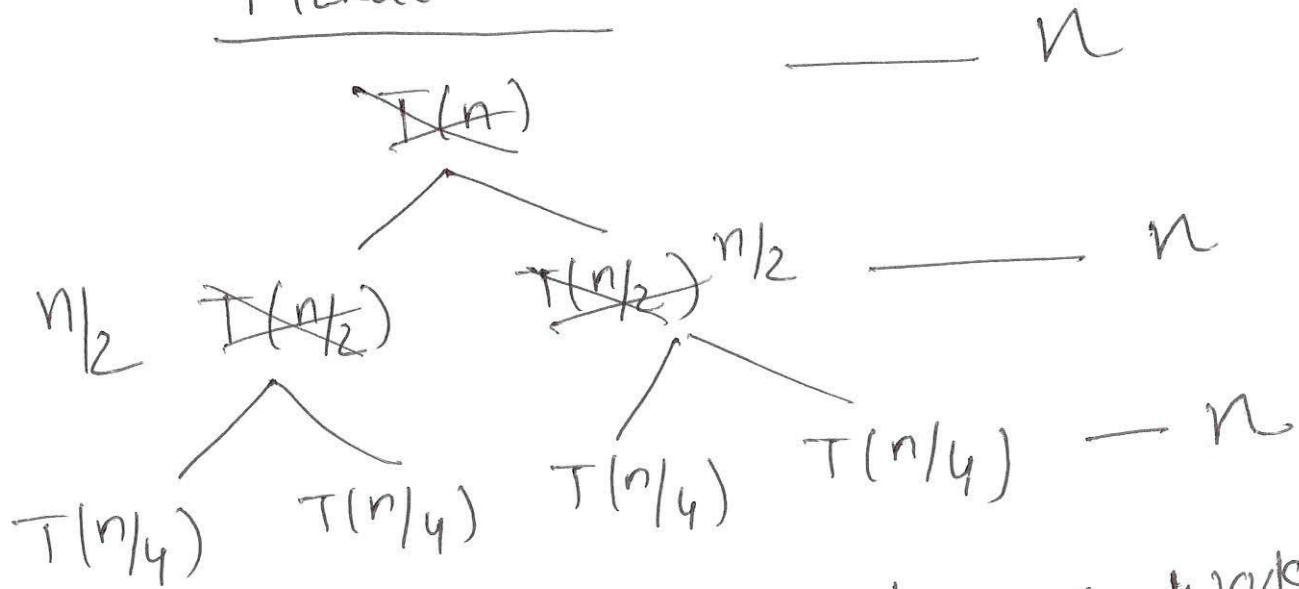
$$n^{(2)} = \Theta(n) \quad [T(n) = \Theta(n)]$$

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

$$T(1) = 1$$

5

MERGE SORT



In every level we are doing n work

$$T\left(\frac{n}{2^k}\right) = 1 \quad \frac{n}{2^k} = 1$$

$$k = \log_2 n$$

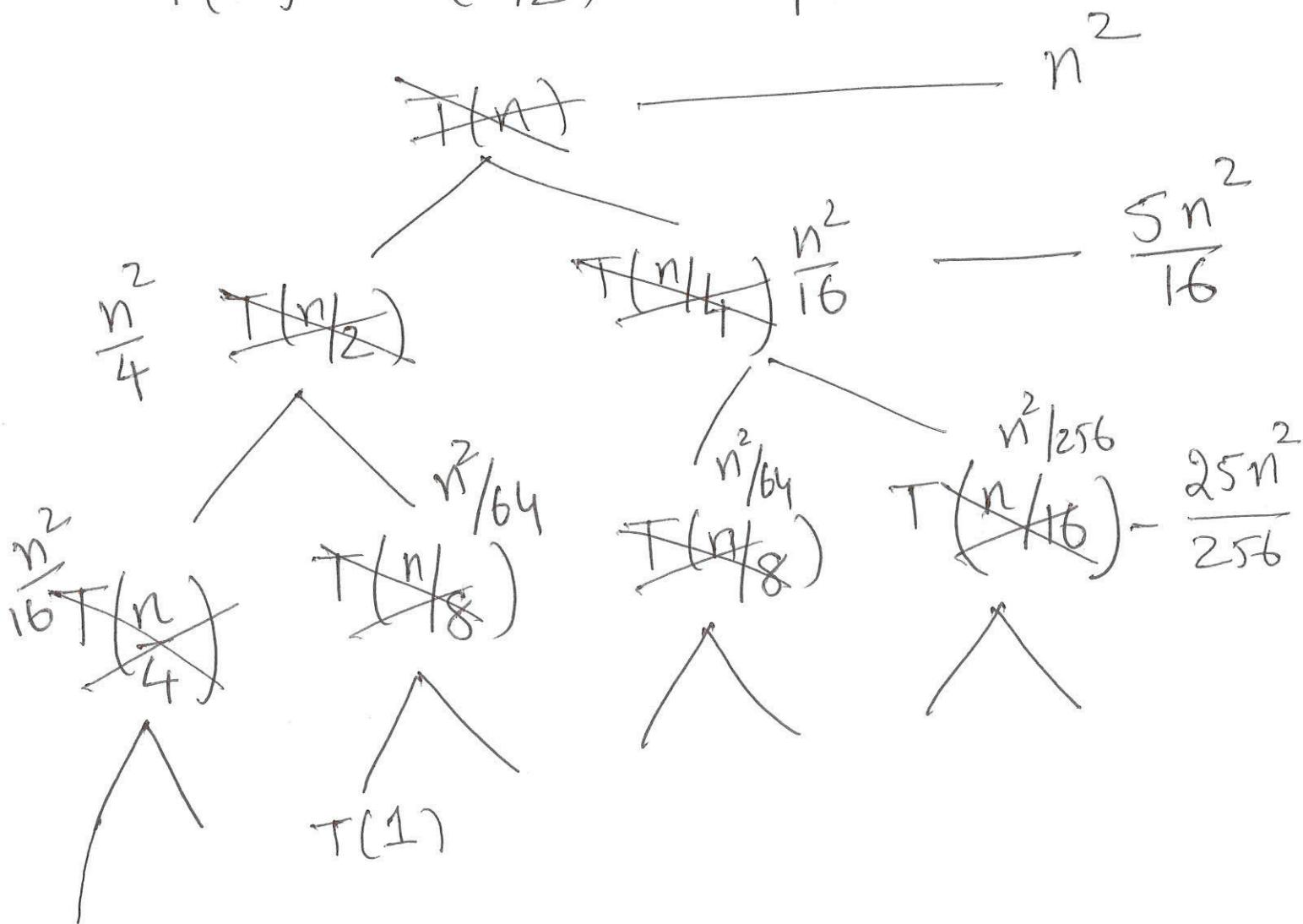
Tree has $\log_2 n$ levels.

So total work done $T(n) = n \cdot \log_2 n$

$$T(n) = \Theta(n \log_2 n)$$

⑥

$$T(n) = T(n/2) + T(n/4) + n^2$$

 $T(1)$

How MANY Leaves are there?
 This is not so obvious as we are
 recurring with different speeds

We are solving n problem as $\frac{n}{2} + \frac{n}{4} = \frac{3n}{4}$ problem
 So Number of leaves must be less than n smaller than

$$\begin{aligned} & n^2 + \frac{5n^2}{16} + \frac{25n^2}{256} + \dots + \frac{5^K n^2}{16^K} \\ & \leq n^2 \left(1 + \frac{5}{16} + \frac{5^2}{16^2} + \frac{5^3}{16^3} + \dots + \frac{5^K}{16^K} \right) \leq 2n^2 \end{aligned}$$

$$T(n) = T(n-1) + T(n-2) + C$$

$$T(1) = 1$$

$$T(0) = 0$$

$$\left[\begin{array}{l} \text{fib}(n) = \text{fib}(n-1) + \text{fib}(n-2) \\ \text{fib}(1) = 1 \\ \text{fib}(0) = 0 \end{array} \right]$$

$$\left[T(n) = \Theta(1 \cdot 6^n) \right]$$

$$T(n) = T(n-1) + T(n-1) + C$$

$$T(1) = 1$$

$$\left[\begin{array}{l} \text{Th(int n, int s, int t, int d)} \\ \quad \text{if}(n) \\ \quad \quad \text{Th}(n-1, s, d, t) \\ \quad \quad \text{Th}(n-1, \frac{s \rightarrow d}{t}, s, d) \end{array} \right]$$

$$\left[T(n) = \Theta(2^n) \right]$$

3.11. MASTER THEOREM

3.11 Master Theorem

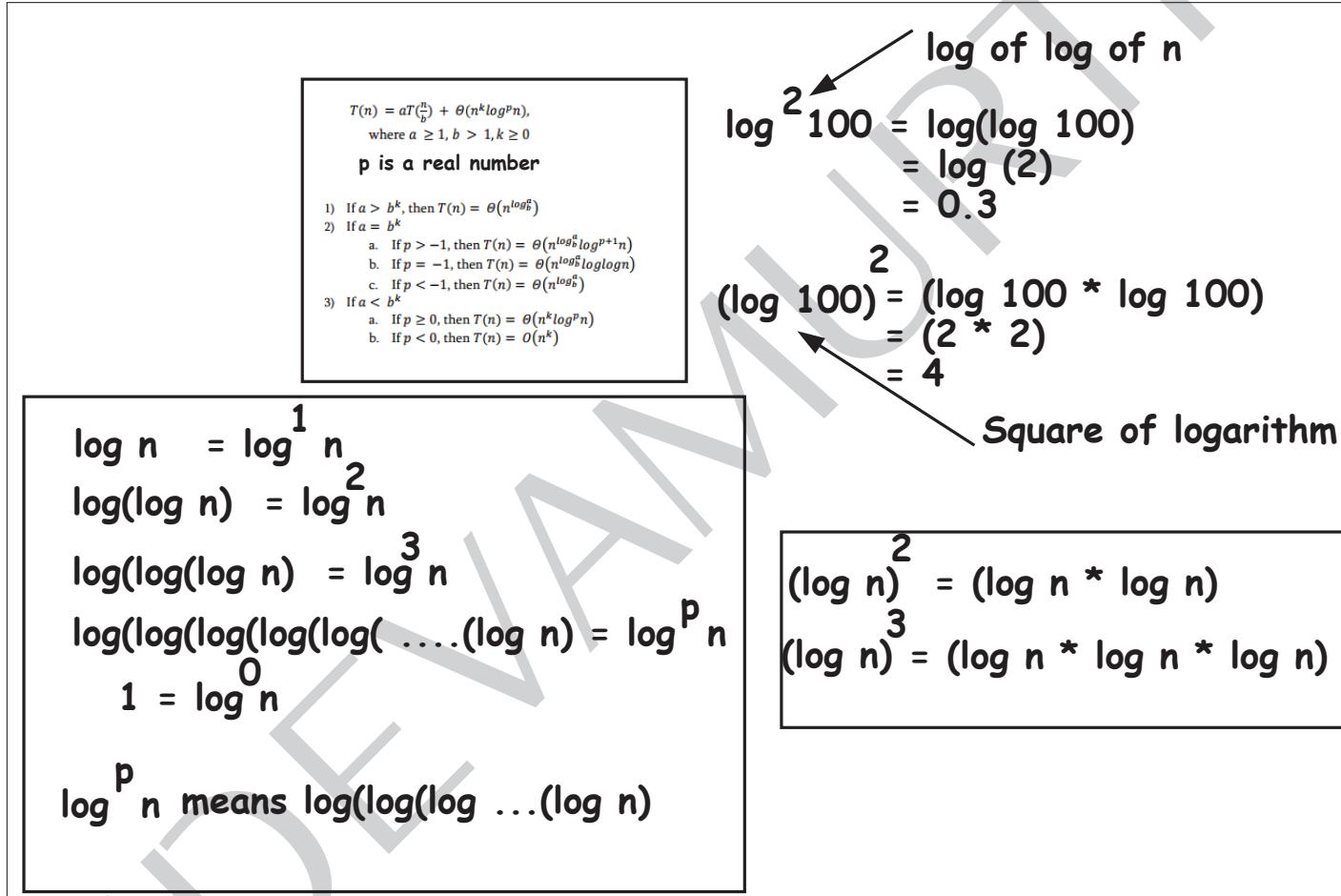


Figure 3.17: Master Theorem

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$

2) If $a = b^k$

a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$

b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$

c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$

3) If $a < b^k$

a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$

b. If $p < 0$, then $T(n) = O(n^k)$

$$1. T(n) = 3T\left(\frac{n}{2}\right) + n^2$$

$$2. T(n) = 7T\left(\frac{n}{2}\right) + n^2$$

$$3. T(n) = 4T\left(\frac{n}{2}\right) + n^2$$

$$4. T(n) = 3T\left(\frac{n}{4}\right) + n \log n$$

$$5. T(n) = 4T\left(\frac{n}{2}\right) + \log n$$

$$6. T(n) = T(n-1) + n$$

$$7. T(n) = 4T\left(\frac{n}{2}\right) + n^2 \log n$$

$$8. T(n) = 5T\left(\frac{n}{2}\right) + n^2 \log n$$

$$9. T(n) = 3T\left(\frac{n}{3}\right) + n/\log n$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$

2) If $a = b^k$

a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$

b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$

c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$

3) If $a < b^k$

a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$

b. If $p < 0$, then $T(n) = O(n^k)$

$$10) T(n) = 2T(n/4) + C$$

$$11) T(n) = T(n/4) + \log n$$

$$12) T(n) = T(n/2) + T(n/4) + n^2$$

$$13) T(n) = 2T(n/4) + \log n$$

$$14) T(n) = 3T(n/3) + n \log n^2$$

$$15) T(n) = 8T((n-\sqrt{n})/4) + n^2$$

$$16) T(n) = 2T(n/4) + \sqrt{n}$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$

2) If $a = b^k$

a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$

b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$

c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$

3) If $a < b^k$

a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$

b. If $p < 0$, then $T(n) = O(n^k)$

$$17) T(n) = 2T(n/4) + n^{0.5}$$

$$18) T(n) = 16T(n/4) + n!$$

$$19) T(n) = 3T(n/2) + n$$

$$20) T(n) = 4T(n/2) + cn$$

$$21) T(n) = 3T(n/3) + n/2$$

$$22) T(n) = 4T(n/2) + n/\log n$$

$$23) T(n) = 7T(n/3) + n^2$$

$$24) T(n) = 8T\left(\frac{n}{3}\right) + 2^n$$

$$25) T(n) = 16T(n/4) + n$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 3T\left(\frac{n}{2}\right) + n^2$$

a = 3

b = 2

k = 2 p = 0

$$\begin{array}{c} a \\ \hline 3 \end{array} \quad \begin{array}{c} b^k \\ \hline 2^2 \end{array} \quad p = 0 \quad \text{Case 3a} \quad \Theta(n^2 \log^0 n) \\ 3 \leq 4 \\ \underline{\text{Case 3}} \quad \quad \quad = \Theta(n^2)$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 7 T\left(\frac{n}{2}\right) + n^2$$

\uparrow \uparrow \uparrow
 $a = 7$ $b = 2$ $k = 2$ $p = 0$

$$\begin{array}{c} \underline{a} \quad \frac{b^k}{2^2} \\ 7 \quad 2^2 \\ 7 > 4 \\ \text{Case 1} \end{array}$$

$\boxed{T(n) = \Theta(n^{\log_2 7})}$

(3)

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 4T\left(\frac{n}{2}\right) + n^2$$

$$a = 4$$

$$b = 2$$

$$k = 2$$

$$p = 0$$

$$\frac{a}{b} = \frac{4}{2} = 2 \quad p = 0 \quad T(n) = \Theta\left(n^{\log_2 4 \cdot (0+1)} \log n\right)$$

$$\frac{4}{4} = 1$$

Case 2

Case 2a

$$T(n) = \Theta(n^2 \log n)$$

$$\log_2 \frac{4}{4} = \log_2 2^2 = 2 \log_2 2 = 2$$

(4)

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 3T\left(\frac{n}{4}\right) + n \log n$$

$$a = 3$$

$$b = 4$$

$$k = 1$$

$$p = 1$$

$$\boxed{n^1 \log^1 n}$$

$$\frac{a}{b} = \frac{3}{4}^1$$

$$p = 1$$

Can $\approx 3a$

$$T(n) = \Theta(n^1 \log^1 n)$$

$$\boxed{T(n) = \Theta(n \log n)}$$

Case 3

(5)

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a} \log^p n)$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 4T\left(\frac{n}{2}\right) + \log^n n$$

$\boxed{a=4}$ $\boxed{b=2}$ $\boxed{k=0}$ $\boxed{p=1}$

$$\begin{array}{c} a \quad b \\ \hline 4 \quad 2^0 \\ 4 \quad 2 \\ 4 > 1 \\ \boxed{\text{Case 1}} \end{array} \quad T(n) = \Theta\left(n^{\log_2 4}\right)$$

$\boxed{T(n) = \Theta(n^2)}$

$$\log_2 4 = \log_2 2^2 = 2 \log_2 2 = 2$$

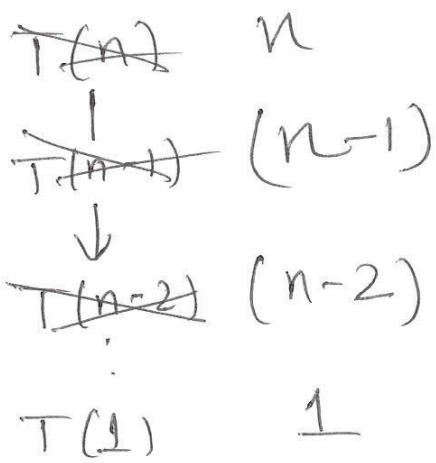
$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$T(n) = T(n-1) + n$
 CANNOT BE SOLVED BY MASTER THEOREM



$$\begin{aligned} \text{Total Work} &= \cancel{n} + 1 + 2 + 3 + \dots + n \\ &= \frac{n(n+1)}{2} = \Theta(n^2) \end{aligned}$$

7

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 4T\left(\frac{n}{2}\right) + n^2 \log n$$

$$\boxed{a=4}$$

$$\boxed{b=2}$$

$$\boxed{k=2}$$

$$\boxed{p=1}$$

$$\begin{array}{r} a \\ \hline 4 \\ b \\ \hline 2 \\ 2 \end{array}$$

$$P=1$$

$$4=4$$

$$\boxed{Ca \geq 2a}$$

$$T(n) = \Theta\left(n^{\log_2 4} \log^{1+1} n\right)$$

$$\boxed{T(n) = \Theta(n^2 \log^2 n)}$$

$$\boxed{Ca \geq 2}$$

$$\log_2 4 = \log_{150} 2^2 = 2 \log_2 2 = 2$$

2

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = \Theta(n^k)$

$$T(n) = ST\left(\frac{n}{2}\right) + n^2 \log n$$

$$\boxed{a=5}$$

$$\boxed{b=2}$$

$$\boxed{k=2}$$

$$\boxed{p=1}$$

$$\begin{array}{c} a \\ \xrightarrow{s} \\ 5 \\ \xrightarrow{k} \\ 2^k \\ 2 \\ 5 > 4 \end{array}$$

Case 1

$$\boxed{T(n) = \Theta(n^{\log_2 5})}$$

Q1

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 3T\left(\frac{n}{3}\right) + n/\log n$$

$$\boxed{a=3}$$

$$\boxed{b=3}$$

CANNOT APPLY
MASTER THEOREM

(10)

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 2T\left(\frac{n}{4}\right) + C$$

$\boxed{a=2}$ $\boxed{b=4}$ $n^0 \log^0 n$
 $\boxed{k=0}$ $\boxed{p=0}$

$$\frac{a}{2} \frac{b^k}{4^0}$$

$$T(n) = \Theta\left(n^{\log_4 2}\right)$$

$$2 > \frac{1}{4}$$

CASE 1

$$\boxed{T(n) = \Theta(n^{1/2})}$$

$$\log_4 2 = \frac{\log_2 2}{\log_2 4} = \frac{1}{2 \log_2 2} = \frac{1}{2}$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = T\left(\frac{n}{4}\right) + \log_n^{\frac{1}{4^0}} n$$

\uparrow \uparrow $n^0 \log^{\frac{1}{4^0}} n$
 $a=1$ $b=4$ $k=0$ $p=1$

$$\begin{aligned}
 & \frac{a}{1} = \frac{b}{4^0} \quad p=1 \\
 & 1 = 1 \quad \boxed{\text{case 2a}}
 \end{aligned}
 \quad
 \begin{aligned}
 T(n) &= \Theta\left(n^{\log_4^{\frac{1}{4^0}} 1+1} n\right) \\
 &= \Theta\left(n^0 \log^2 n\right) \\
 \boxed{T(n)} &= \boxed{\Theta(\log^2 n)}
 \end{aligned}$$

$$\log_{\text{any base}} \frac{1}{4^0} = 0$$

$$\begin{aligned}
 \frac{4^0}{\log_4^{\frac{1}{4^0}}} &= 1 \\
 \boxed{\log_4^{\frac{1}{4^0}} = 0}
 \end{aligned}$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = T(n/2) + T(n/4) + n^2$$

Cannot be solved by master rule

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 2T\left(\frac{n}{4}\right) + \log n$$

$\overset{a=2}{\boxed{}}$ $\overset{b=4}{\boxed{}}$ $\overset{\log n}{\boxed{}}$ $\overset{\cancel{k=0}}{\boxed{}}$ $\overset{p=1}{\boxed{}}$

$$\begin{array}{r} a \\ \hline 2 \end{array} \quad \begin{array}{r} b \\ \hline 4 \end{array} \quad \begin{array}{r} k \\ \hline 0 \end{array}$$

$$T(n) = \Theta(n^{\log_4 2})$$

$$\begin{array}{r} 2 \\ \hline 2 \end{array} \quad \begin{array}{r} 1 \\ \hline 1 \end{array}$$

$2 > 1$

$$T(n) = \Theta(n^{1/2})$$

$$\log_4 2 = \frac{\log_2 2}{\log_2 4} = \frac{1}{2}$$

$$T(n) = aT\left(\frac{n}{b}\right) + \theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 3T\left(\frac{n}{3}\right) + n \log n$$

$$\boxed{a=3}$$

$$\boxed{b=3}$$

$$\boxed{k=1}$$

$$\boxed{p=1}$$

$$\begin{array}{c} a \\ \hline 3 \\ \frac{b^k}{3^1} \\ \hline 3 = 3 \end{array} \quad \boxed{p=1} \quad \boxed{\text{Case 2a}} \quad \boxed{\text{Case 2}}$$

$$T(n) = \Theta\left(n^{\log_3^3 \log^{1+1} n}\right)$$

$$\boxed{T(n) = \Theta(n \log^2 n)}$$

(15)

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 8T\left(\frac{(n-\sqrt{n})}{4}\right) + n^2$$

MT cannot be used

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 2T\left(\frac{n}{4}\right) + \sqrt{n}$$

$n^{1/2} \log^0 n$

a = 2

b = 4

k = 1/2

p = 0

$$\begin{array}{cccc} a & b & k \\ \hline 2 & 4^{1/2} & 0 \\ 2 = 2 & \boxed{\text{case 2a}} & T(n) = \Theta\left(n^{\log_4^2 \log^{0+1} n}\right) \\ \underline{\text{case 2}} & \boxed{T(n) = \Theta(n^{1/2} \log n)} & \end{array}$$

$$\log_4^2 = \frac{\log_2^2}{\log_2^{159}} = \frac{1}{2}$$

(17)

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 2T\left(\frac{n}{4}\right) + n^{0.51}$$

$$\boxed{a=2}$$

$$\boxed{b=4}$$

$$\boxed{k=0.51}$$

$$\boxed{p \geq 0}$$

$$\begin{array}{r} a \\ \hline 2 \end{array} \quad \begin{array}{r} b \\ \hline 4 \end{array} \quad \begin{array}{r} k \\ \hline 0.51 \end{array} \quad \boxed{p=0} \quad \text{Case 3a}$$

$$\begin{aligned} T(n) &= \Theta\left(n^{0.51} \log^0 n\right) \\ &= \Theta\left(n^{0.51}\right) \end{aligned}$$

$$2 \cancel{<} 2.02$$

$$\boxed{\text{Case 3}}$$

(18)

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 16T\left(\frac{n}{4}\right) + n!$$

$$a = 16$$

$$b = 4$$

$$\Theta(n!)$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 3T\left(\frac{n}{2}\right) + n$$

$\overset{a=3}{\nearrow}$ $\boxed{b=2}$ $n \underset{\substack{\downarrow \\ k=1}}{\log^0 n}$ $\boxed{k=1}$ $\boxed{p=0}$

$$\frac{a}{3} \frac{b^k}{2^k}$$

$$3 \quad 2$$

$$3 > 2$$

Case 1

$$T(n) = \Theta(n^{\log_2 3})$$

(20)

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 4T\left(\frac{n}{2}\right) + cn$$

$\frac{a=4}{n^{\log_2^1} \log^0 n}$ $\frac{b=2}{k=1}$ $\frac{p=0}{p=0}$

$$\frac{a}{4} \cdot \frac{b^k}{2} = \frac{4}{4} \cdot \frac{2^1}{2} = 2 \quad T(n) = \Theta(n^{\log_2 4})$$

$$4 > 2 \quad T(n) = \Theta(n^2)$$

$\boxed{\text{Case 1}}$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 3T\left(\frac{n}{3}\right) + \frac{n}{2}$$

$\boxed{a=3}$ $\boxed{b=3}$ $\frac{1}{2} n \log^0 n$

$$\begin{array}{c} a \\ \hline 3 \\ b \\ \hline 3 \\ 1 \end{array}$$

$\boxed{a=3}$ $\boxed{b=3}$ $\boxed{k=1}$ $\boxed{p=0}$

$b=0$ $T(n) = \Theta\left(n^{\log_3^3 \log^{0+1} n}\right)$

Case 2a

$\boxed{\text{Case 2}}$

$T(n) = \Theta(n \log n)$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 7T\left(\frac{n}{3}\right) + n^2$$

$$\boxed{a=7}$$

$$\boxed{b=3}$$

$$\boxed{k=2}$$

$$\boxed{p=0}$$

$$\begin{array}{r} a \\ \hline 7 \\ b \\ \hline 3 \\ ^k \\ 2 \\ \hline 7 > 6 \end{array}$$

$$T(n) = \Theta\left(n^{\log_3 7}\right)$$

$$\approx \Theta\left(n^{\log_3 9}\right)$$

Case 1

$$\boxed{T(n) \approx \Theta(n^2)}$$

$$\log_3 9 = \frac{\cancel{\log_7 7}}{\cancel{\log_7 165}} \cdot \cancel{\log_7 1} \quad \log_3 3^2 = 2$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 8T\left(\frac{n}{3}\right) + 2^n$$

$$T(n) = \Theta(2^n)$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$T(n) = 16T\left(\frac{n}{4}\right) + n$$

a = 16 b = 4 k = 1 p = 0

$$\begin{array}{c} a \quad b^k \\ \hline 16 \quad 4^1 \\ 16 > 4 \end{array}$$

$$T(n) = \Theta\left(n^{\log_4 16}\right)$$

$$= \Theta(n^2)$$

Case 1

3.12. PROBLEM SET

3.12 Problem set

Problem 3.12.1.

Implement computing changes for US coins as explained in figure 3.18

Computing changes for US coins

UscoinsBase.java

```
abstract class UscoinsBase{
    protected final int[] C = {25, 10, 5, 1} ;
    protected final int CS = C.length ;
    protected final int ITERATIVE = 1 ;
    protected final int RECURSIVE = 2 ;

    abstract protected ArrayList<Integer> computeChange(int n, int method) ;
    protected void testBench();
}
```

$(n > 0 \&& n < 100)$
 Must work from 1 cents to 99 cents
 Both iterative and recursive routines
 must return change in descending order

Uscoins.java

```
class Uscoins extends UscoinsBase{
    Uscoins() {
        testBench();
    }

    @Override
    protected ArrayList<Integer> computeChange(int n, int method) {
        if (method == ITERATIVE) {
            return itr(n);
        } else {
            return rec(n);
        }
    }
}
```

This must be recursive routine
 Do not call itr(n) inside rec(n)

Min number of coins for 70 cents	4
The coins are:	25 25 10 10
Min number of coins for 71 cents	5
The coins are:	25 25 10 10 1
Min number of coins for 72 cents	6
The coins are:	25 25 10 10 1 1
Min number of coins for 73 cents	7
The coins are:	25 25 10 10 1 1 1

You cannot change anything in UscoinsBase.java
 All tests in UscoinsBase.java must pass
 email only: Uscoins.java

Figure 3.18: Computing changes for US coins

3.12. PROBLEM SET

Problem 3.12.2.

CHAPTER 3. RECURSION

Draw on a paper the stack trace of **Tower of Hanoi** problem for 3 disks. Show clearly your final solution. How many steps are required? No code is required.

3.12. PROBLEM SET

Problem 3.12.3. Write a procedure called **length** as shown in the figure 3.19. You must use the code given in 3.20 Show on paper, how your code works. Note that you cannot use any loop statements and array length is not known.

Finding length

s	0	1	2	3	4	5
	5	1	0	4	2	3

1. s is an integer array
2. Length of the array is NOT given. **YOU CANNOT USE a.length**
3. If the length of the array is 6 (as shown above)
the content of the array is guaranteed to be between 0 to 5.
THERE IS NO REPETITION of numbers

The top level call is as follows:

```
int a[6] = {5,1,0,4,2,3};  
int y = length_easy(a, 3);
```

Your task is to find length which is defined as follows:

You start from a[x], in this case x = 3, a[3] = 4, and keep looping until you get x, which is 3. The number of times you looped, in this example, is y =

```
a[3] = 4  
a[4] = 2  
a[2] = 0  
a[0] = 5  
a[5] = 3
```

y = Length is = 4

One way, to write, using while loop is:

```
private static int length_easy(int [] s, int x) {  
    int l = 0;  
    int gx = x;  
    while (true) {  
        if (s[x] == gx) {  
            return l;  
        }  
        x = s[x];  
        ++l;  
    }  
}
```

```
void test_length_easy() {  
    int s[6] = {5,1,0,4,2,3};  
    int y = length_easy(s, 3);  
    assert (y == 4);  
}
```

Now write "length" subroutine as follows:

```
int length (int [] s, int x) {  
}
```

0. Content of array s should be exactly same after executing procedure length
1. You cannot change interface of length function
2. You cannot use global/static variables
3. You cannot use any loop statements like while, do, for and goto
4. You cannot use any subroutine. Only guts should be written in above procedure
5. Your code should not be more than 10 lines

```
void test_length() {  
    int s[6] = {5,1,0,4,2,3};  
    int b[6] = {5,1,0,4,2,3};  
    int y = length (s, 3);  
    assert(y == 4);  
    for (int i = 0 ; i < 6; i++) {  
        assert(s[i] == b[i]);  
    }  
}
```

3.12. PROBLEM SET

Length.java

```
class Length {
    private static final IntUtil u = new IntUtil();
```

```
private static int length_easy(int [] s, int x) {
    int l = 0 ;
    int gx = x ;
    while (true) {
        if (s[x] == gx) {
            return l ;
        }
        x = s[x] ;
        ++l ;
    }
}
```

```
public static void testbed() {
    int s[] = {5,1,0,4,2,3} ;
    int y = length_easy(s,3) ;
    System.out.println("length_easy y = " + y);
    u.myassert(y == 4) ;
    int b[] = {5,1,0,4,2,3} ;
    int x = length(s,3) ;
    System.out.println("length x = " + x);
    u.myassert(x == y) ;
    for (int i = 0; i < s.length; ++i) {
        u.myassert(s[i] == b[i]);
    }
    System.out.println("Assert passed");
}
```

```
private static int length(int [] s, int x) {
    //WRITE YOUR CODE ONLY HERE
    }
```

```
public static void testbed1() {
    int s[] = {5,1,0,4,2,3} ;
    int b[] = {5,1,0,4,2,3} ;
    int l = s.length ;
    for (int j = 0; j < l ; ++j) {
        int y = length_easy(s,j) ;
        System.out.println("length_easy y = " + y);
        int x = length(s,j) ;
        System.out.println("length x = " + x);
        u.myassert(x == y) ;
        for (int i = 0; i < s.length; ++i) {
            u.myassert(s[i] == b[i]);
        }
        System.out.println("Assert passed");
    }
}
```

```
public static void main(String[] args) {
    System.out.println("Length.java");
    testbed();
    testbed1();
}
```

Figure 3.20: Write code only in the procedure **length**

CHAPTER 3. RECURSION

Problem 3.12.4. Write a procedure called **fibS** that does not uses recursion as shown in the figure 3.21. You must use the code given in 3.21

```
private static int fibS(int n) {
    setKount();
    //WRITE YOUR CODE HERE

    System.out.println("fib of " + n + ": = " + a + " max Stack used (num executed)= " + getKount());
}
```

```
for (int i = 2; i < 30; ++i) { //Run with 300 to see hangs at 50
    int a = fibS(i);
}
```

fib of 2: = 1 max Stack used (num executed)= 2
fib of 3: = 2 max Stack used (num executed)= 2
fib of 4: = 3 max Stack used (num executed)= 3
fib of 5: = 5 max Stack used (num executed)= 3
fib of 6: = 8 max Stack used (num executed)= 4
fib of 7: = 13 max Stack used (num executed)= 4
fib of 8: = 21 max Stack used (num executed)= 5
fib of 9: = 34 max Stack used (num executed)= 5
fib of 10: = 55 max Stack used (num executed)= 6
fib of 11: = 89 max Stack used (num executed)= 6
fib of 12: = 144 max Stack used (num executed)= 7
fib of 13: = 233 max Stack used (num executed)= 7
fib of 14: = 377 max Stack used (num executed)= 8
fib of 15: = 610 max Stack used (num executed)= 8

fib of 16: = 987 max Stack used (num executed)= 9
fib of 17: = 1597 max Stack used (num executed)= 9
fib of 18: = 2584 max Stack used (num executed)= 10
fib of 19: = 4181 max Stack used (num executed)= 10
fib of 20: = 6765 max Stack used (num executed)= 11
fib of 21: = 10946 max Stack used (num executed)= 11
fib of 22: = 17711 max Stack used (num executed)= 12
fib of 23: = 28657 max Stack used (num executed)= 12
fib of 24: = 46368 max Stack used (num executed)= 13
fib of 25: = 75025 max Stack used (num executed)= 13
fib of 26: = 121393 max Stack used (num executed)= 14
fib of 27: = 196418 max Stack used (num executed)= 14
fib of 28: = 317811 max Stack used (num executed)= 15
fib of 29: = 514229 max Stack used (num executed)= 15

Figure 3.21: Fibonacci number using stack

3.12. PROBLEM SET

Problem 3.12.5. Write a procedure called `th` that does not uses recursion as shown in the figure 3.22. You must use the code given in 3.22

```

public class TowerOfHanoi {
    private int w = 0 ;
    public void resetKount() {w = 0; }
    public void incKount() {++w ; }
    public void P() {System.out.println("Num move = " + w);}
    TowerOfHanoi(int n, boolean recursive) {
        resetKount();
        if (recursive) {
            th_r(n,1,2,3);
        }else {
            th(n,1,2,3);
        }
    }

    private void th_r(int n, int src, int aux, int dst) {
        if (n > 0) { //at least one disk
            th_r(n-1,src,dst,aux) ; //Move n-1 from 'src' to 'aux'(using dst)
            incKount();
            System.out.println(src + "->" + dst);
            th_r(n-1,aux,src,dst) ; //move n-1 from 'aux' to 'dst'(using src)
        }
    }

    private void th(int n, int src, int aux, int dst) {
        //WRITE YOUR CODE
    }

    private static void testBench() {
        for (int i = 1; i < 10; ++i) {
            System.out.println("----- " + i + "-----");
            {
                TowerOfHanoi t = new TowerOfHanoi(i,true);
                t.P();
            }
            {
                TowerOfHanoi t = new TowerOfHanoi(i,false);
                t.P();
            }
        }
    }

    public static void main(String[] args) {
        System.out.println("TowerOfHanoi.java");
        testBench();
        System.out.println("TowerOfHanoi.java DONE");
    }
}

```

Figure 3.22: Tower of Hanoi using stack

3.12. PROBLEM SET

Problem 3.12.6. Solve the following recursion equations using **Master Theorem**

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

- 1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$
- 2) If $a = b^k$
 - a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$
 - b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$
 - c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$
- 3) If $a < b^k$
 - a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$
 - b. If $p < 0$, then $T(n) = O(n^k)$

$$\textcircled{1} \quad T(n) = 16T\left(\frac{n}{4}\right) + \frac{n}{2}$$

$$\textcircled{2} \quad T(n) = 4T\left(\frac{n}{2}\right) + n$$

$$\textcircled{3} \quad T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{2 \log n}$$

$$\textcircled{4} \quad T(n) = 3T\left(\frac{n}{2}\right) + \frac{n}{2}$$

$$\textcircled{5} \quad T(n) = T\left(\frac{n}{2}\right) + \frac{n}{n}$$

$$\textcircled{6} \quad T(n) = 2^n T\left(\frac{n}{2}\right) + n$$

$$\textcircled{7} \quad T(n) = 2T\left(\frac{n}{2}\right) + n \log n$$

$$\textcircled{8} \quad T(n) = 2T\left(\frac{n}{4}\right) + n^{0.51}$$

$$\textcircled{9} \quad T(n) = 0.5 T\left(\frac{n}{2}\right) + 1/n$$

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^k \log^p n),$$

where $a \geq 1, b > 1, k \geq 0$

p is a real number

1) If $a > b^k$, then $T(n) = \Theta(n^{\log_b^a})$

2) If $a = b^k$

a. If $p > -1$, then $T(n) = \Theta(n^{\log_b^a} \log^{p+1} n)$

b. If $p = -1$, then $T(n) = \Theta(n^{\log_b^a} \log \log n)$

c. If $p < -1$, then $T(n) = \Theta(n^{\log_b^a})$

3) If $a < b^k$

a. If $p \geq 0$, then $T(n) = \Theta(n^k \log^p n)$

b. If $p < 0$, then $T(n) = O(n^k)$

$$10) T(n) = 6T\left(\frac{n}{3}\right) + n^2 \log n$$

$$11) T(n) = 64T\left(\frac{n}{8}\right) - \underline{n^2} \log n$$

$$12) T(n) = 7T\left(\frac{n}{3}\right) + n$$

$$13) T(n) = 4T\left(\frac{n}{2}\right) + \log n$$

$$14) T(n) = \sqrt{2} T\left(\frac{n}{2}\right) + \log n$$

$$15) T(n) = 2T\left(\frac{n}{2}\right) + \sqrt{n}$$

$$16) T(n) = 3T\left(\frac{n}{2}\right) + n$$

$$17) T(n) = 3T\left(\frac{n}{3}\right) + \sqrt{n}$$

$$18) T(n) = 4T\left(\frac{n}{2}\right)^{180} + Cn$$

$$19) T(n) = 3T\left(\frac{n}{4}\right) + n \log n$$