

$$\text{Ex } T(n) = 2T(n/2) + n^2 \quad \text{Ex } T(n) = 2T(n/2) + n \quad \text{Ex } T(n) = \sqrt{n}T(n/2) + n$$

$\Rightarrow 2=2, b=2, f(n)=n^2 \quad \Rightarrow 2=2, b=2, f(n)=n \quad \Rightarrow 4, b=2, f(n)=n$

$n^2 \Leftrightarrow n^{\log_2 2}$ $f(n) = n^{\frac{\log_2 2}{2}}$ $t(n) \Leftrightarrow n^{\frac{\log_2 2}{2}}$

$n^2 > n$ $n = n^{\frac{\log_2 2}{2}}$ $n < n^{\frac{\log_2 2}{2}}$

$2f(n/2) \leq c \cdot f(n)$ $= \Theta(n^{\frac{\log_2 2}{2} \log n})$ $n < n^{\frac{\log_2 2}{2}}$

$2 \cdot \frac{n^2}{4} \leq c \cdot n^2$ $= \Theta(n \cdot \log n)$ $n < n^{\frac{\log_2 2}{2}}$

$\frac{1}{2} \leq c \cdot \frac{c \cdot b}{2}$ Case 2 Case 1

$\Rightarrow \Theta(f(n)) = \Theta(n^2)$ Case 3

$$\text{Ex } T(n) = c + T(n/2)$$

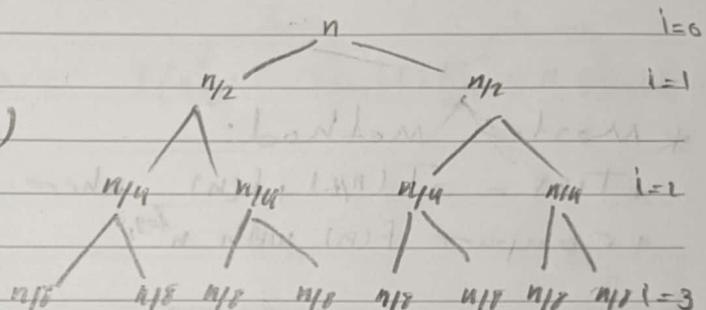
$c \Leftrightarrow n^{\frac{\log_1}{2}}$

$c = 1$

$T(n) = \Theta(n^{\frac{\log_1}{2} \log n})$

$= \Theta(\log n)$

$$\text{Ex } T(n) = 2T(n/2) + n$$



nodes count $i = 2^i = 2^i$

Height $i = \log n$

size input = $n + n/2 + n/4 + \dots + n/2^i$ $2^i = n \Rightarrow i = \log n$

$$\text{Cost} = \frac{n}{2^i}$$

$$\text{last leaves: } 2^{\frac{\log n}{2}}, n^{\frac{\log 2}{2}}$$

$$\text{Total complexity: } \sum_{i=0}^{\log n} n \left(\frac{1}{2}\right)^i + n^{\frac{\log 2}{2}} \cdot T(1) = n + \log n$$

$$(\log n + 1) n + n = n \log n = \Theta(\log n)$$

* master method

$$T(n) = f(n) + dT(n/2)$$

$$d=2, b=2, f(n)=n$$

$$n \Leftrightarrow n^{\log_2 2}$$

$$n \Leftrightarrow n^{\frac{\log_2 2}{2}}$$

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

* substitution

$$T(n) = 2T(n/2) + n \quad \text{and } \log n - d \cdot n + n \leq d \cdot n \log n$$

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

$$dn \geq n$$

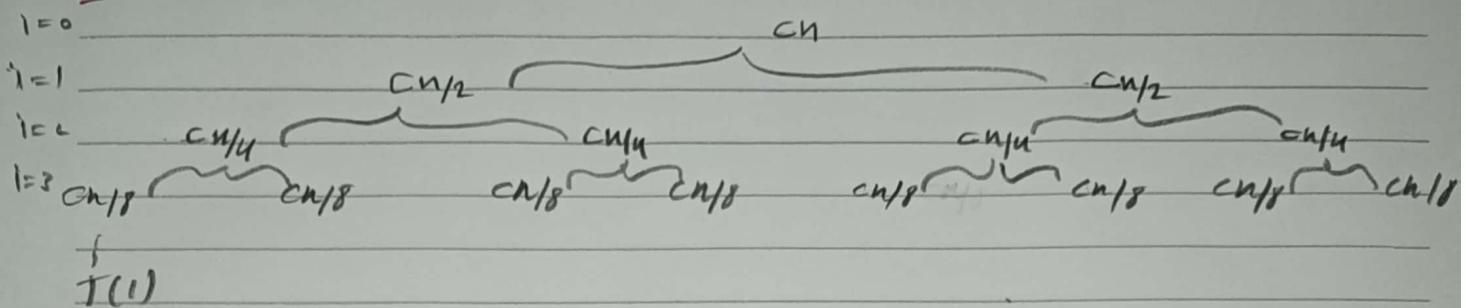
$$T(n/2) = dn/2 \log n/2$$

$$d > 1$$

$$T(n/2) = dn/2 \log n/2 - dn/2$$

$$2T(n/2) + n = dn \log n$$

$$\underline{\underline{EX}} \quad T(n) = 2T(n/2) + cn$$



$T(1)$

- Height $\Rightarrow i = n + n/2 + n/4 + \dots + n/2^i \Rightarrow n = 2^i \Rightarrow \log n = i$

- nodes $2^i = 2^n$

- input size $= cn/2^i$

- last level leaves $= 2^{\log_2 n}$

- cost $i = n \log n - 1$

Total complexity: $\sum_{i=0}^n cn + 2^{\log_2 n} T(1) = n \log n - n + nc \cdot T(1)$

$$\Rightarrow O(n \log n)$$

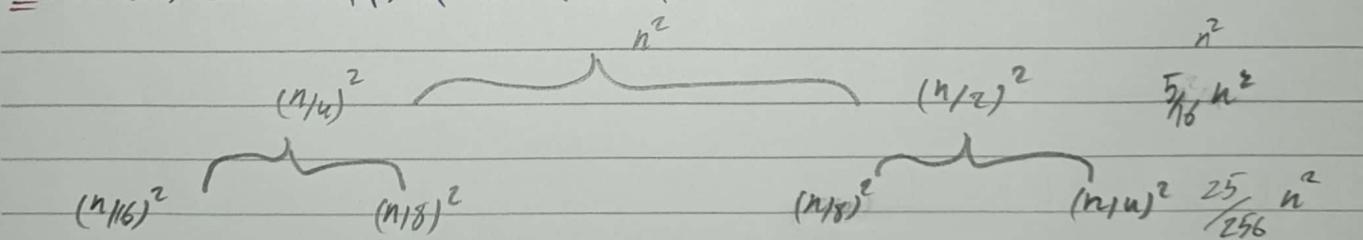
Height \times cost = total complexity

$$\underline{\underline{EX}} \quad T(n) = T\left(\frac{2n}{3}\right) + 1 \quad a=1, b=\frac{3}{2}, f(n)=1$$

$$1 \leq \log_{\frac{3}{2}} \frac{\log n}{1}$$

$$1 = 1 \Rightarrow O\left(n^{\frac{\log 1}{\log \frac{3}{2}}} \cdot \log n\right) = O(\log n)$$

$$\underline{\underline{EX}} \quad T(n) = T(n/4) + T(n/2) + n^2$$



$$n^2 \left(1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \dots \right) = O(n^2)$$

$$n^2 \sum_{i=0}^{n-1} r^n = \frac{1 - (5/16)^n}{1 - 5/16} = \frac{1}{1 - 5/16} = \frac{16}{11}$$

مالعادي

$$\underline{\text{Ex}} \quad T(n) = T(n/2) + c$$

$$\begin{aligned} & \leftarrow T(n) \quad \sum_{i=0}^{\log_2 n - 1} c + c \cdot T(1) \\ & \downarrow \\ & \leftarrow T(n/2) \quad (c \cdot \frac{\log n}{2} + c \cdot T(1)) = O(\log n) \\ & \downarrow \\ & \leftarrow T(1) \end{aligned}$$

$$\underline{\text{Ex}} \quad T(n) = T(n-1) + n$$

$$\begin{aligned} & n \quad T(n) \quad \sum_{i=0}^{n-1} i = \frac{n(n+1)}{2} = O(n^2) \\ & n-1 \quad T(n-1) \\ & n-2 \quad T(n-2) \\ & \vdots \\ & n \quad T(n) \end{aligned}$$

* Master method:

* $T(n) = aT(n/b) + f(n)$ where $a \geq 1$, $b > 1$ and $f(n) > 0$

* Compare $f(n)$ with $n^{\log_b a}$

• Case 1: $f(n) = O(n^{\log_b a - \epsilon})$ for some $\epsilon > 0$, then: $T(n) = O(n^{\log_b a})$

• Case 2: $f(n) = \Theta(n^{\log_b a})$, then: $T(n) = \Theta(n^{\log_b a} \log n)$

• Case 3: $f(n) = \Omega(n^{\log_b a})$, then: for $\epsilon > 0$ and if $aT(n/b) \leq cT(n)$ for some $c < 1$ and all sufficiently large n , then:
 $T(n) = \Theta(f(n))$

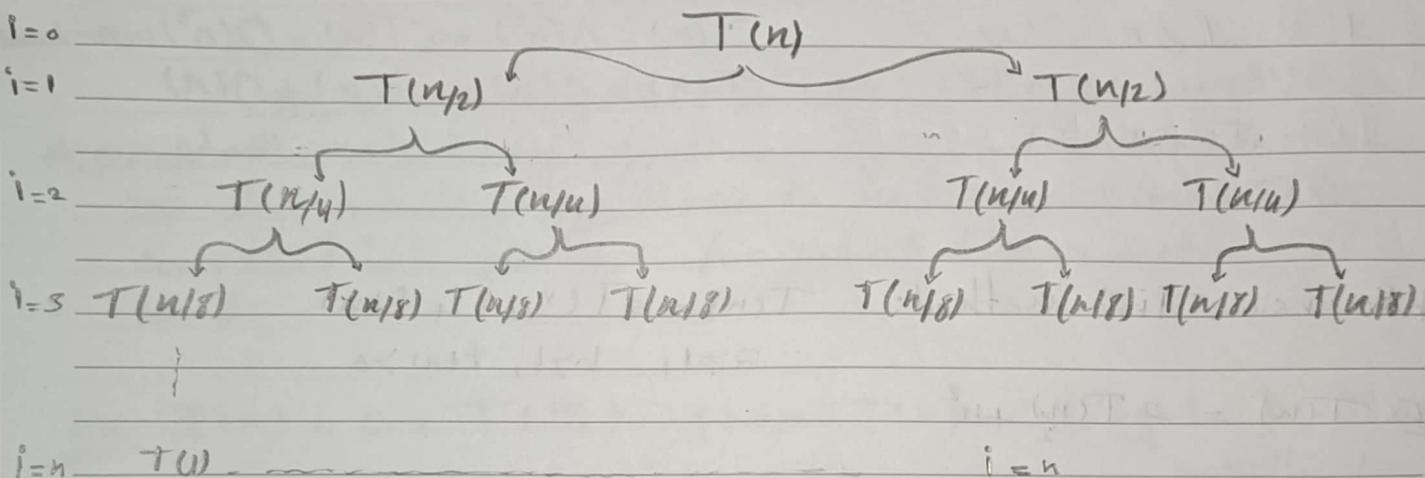
$$T(n) = n^2 \sum_{i=0}^{\log_b 2 - 1} \left(\frac{1}{2}\right)^i + n^{\log_b 2} \cdot T(1)$$

$$\sum_{i=0}^{\infty} x^i = \frac{1}{1-x}; x < 1 = 2n^2 + n \cdot T(1) = O(n^2)$$

مقدار تفرعات الдерج

$$T(n) = 2 \cdot T(n/2) + n^2$$

$$T(n) = 2 \cdot T(n/2) + F(n)$$



- Number of nodes at level $i = 2^i \Rightarrow 2^i$

- input size at level $i = \frac{n}{2^i} \Rightarrow \frac{n}{2^i}$

- Height of tree = $n + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \dots + \frac{n}{2^i} \quad || \quad \frac{n}{2^i} = \frac{n}{b^i}, 2^i = b^i = n$

~~أكمل~~ Number of leaves = $2^{\log_b n}$ \Leftarrow Height $\Leftarrow i = \log_b n$

- cost at $i = \frac{n^2}{2} = \frac{n^2}{b^i}$

- Total complexity equal total complexities of all levels.

$$\sum_{i=0}^{\log_b n - 1} n^2 \left(\frac{1}{2}\right)^i + n^{\log_b 2} \cdot T(1)$$

مقدار تفرعات الدرج $\times T(1)$

Ex $T(n) = n + T(n-1)$ is $O(n^2)$

$$T(n) \leq d \cdot g(n)$$

$$T(n) \leq 1 \cdot n^2$$

$$T(n-1) \leq d \cdot (n-1)^2$$

$$T(n-1) \leq dn^2 - 2n + d$$

$$T(n) \leq d \cdot n^2$$

$$n + T(n-1) \leq d \cdot n^2$$

$$n + dn^2 - 2nd + d \leq dn^2$$

$$\frac{d}{2d-1} \leq n$$

$$d-1 \leq n$$

$$= 2 \cdot 2^0$$

الحالات الممكنة هي

الحالات الممكنة هي

$$T(n) \leq d \cdot g(n)$$

$$T(n) \leq d \cdot n$$

$$T(n-1) \leq d \cdot (n-1)$$

$$T(n-1) \leq dn - d$$

$$T(n) \leq dn$$

$$n + T(n-1) \leq dn$$

$$n + dn - d \leq dn$$

$$n \leq d \text{ implies } n \leq d$$

$$T(n) = O(n^2) \Rightarrow T(n) = O(n^3) \text{ upper bound}$$

$$T(n) = O(n^2) \Rightarrow T(n) = O(n) \text{ lower bound}$$

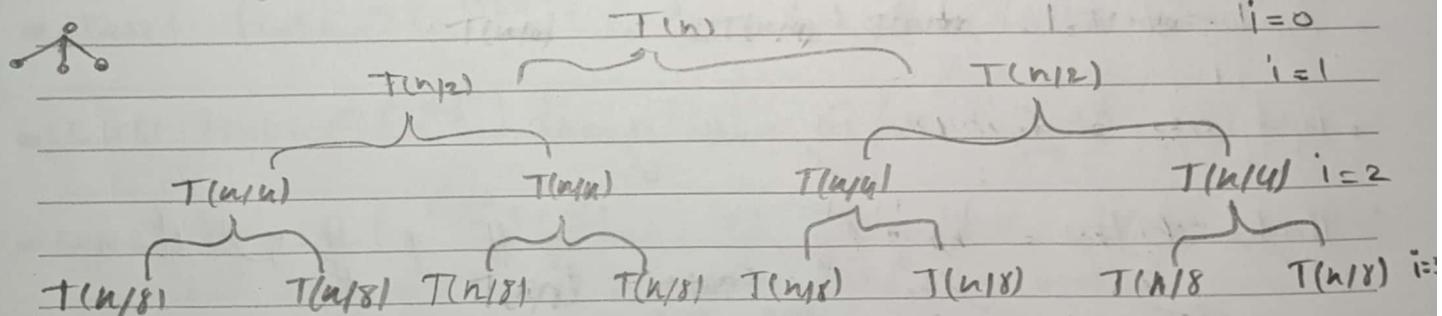
* Recursion Tree Method: $T(n) = aT(n/b) + f(n)$

$$a \geq 1, b \geq 1, f(n) > 0$$

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

الحالات الممكنة

'3' $\in \sqrt{n}$



• $2^0 + 2^1 + 2^2 + \dots + 2^i = 2^{i+1} - 1$ = nodes $\parallel 2^i = \text{nodes count}$

$T(1)$ • input size is n/b^i

• Height of tree: $\frac{n}{2^i} = 1 \Rightarrow 2^i = n \Rightarrow i = \log_2 n$

• cost at i : $T(n) = n^2, T(n/2) = (n/2)^2$

$\text{nodes} = 2^i \quad = \frac{n^2}{2^i}$
 $\Rightarrow \log_2 n \quad \text{total complexity of levels}$
 $\frac{n^2}{2^i} + n^{\log_2 n} * T(1)$

The substitution method.

1. Guess a solution

2. Use induction to prove that the solution works

دروجتى بى تېرىپىلىنىڭ كۈچىنىڭ ئەنچىلىقىسىز

* Guess a solution

$$T(n) = O(g(n))$$

induction goal: apply the definition of the asymptotic notation

$$T(n) \leq d \cdot g(n) \text{ for some } d > 0 \text{ and } n_0 \geq 0$$

ئۇنىڭ ئەنچىلىقىسىز كەلپاڭىزىلما

Induction hypothesis: $T(k) \leq d \cdot g(k)$ for all $k \leq n$

* prove the induction goal:

use the induction hypothesis to find some values of the constant "d" and "n₀" for which the induction goal holds

Ex $T(n) = c + T(\frac{n}{2})$ prove $T(n) = O(\log n)$

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

$$T(n) \leq d \cdot g(n)$$

$$T(n) \leq d \cdot \log n$$

$$T(\frac{n}{2}) \leq d \cdot \log \frac{n}{2}$$

$$T(\frac{n}{2}) \leq d \cdot (\log n - \log 2)$$

$$T(\frac{n}{2}) \leq d \cdot \log n - d$$

$$c + T(\frac{n}{2}) \leq d \cdot \log n$$

$$c + d \cdot \log n - d \leq d \cdot \log n$$

$$c \leq d$$

مۇنىڭ ئەنچىلىقىسىز كەلپاڭىزىلما

ئەنچىلىقىسىز كەلپاڭىزىلما

ئەنچىلىقىسىز كەلپاڭىزىلما

ئەنچىلىقىسىز كەلپاڭىزىلما

$$\underline{\text{Ex}} \quad T(n) = n + T(n/2)$$

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

$$T(n) = n + n/2 + n/4 + T(n/8)$$

$$T(n) = n \cdot (1/2)^{k-1} + T(n/2^k)$$

$$(\frac{1}{2})^{k-1} \Rightarrow \sum_{i=0}^{k-1} (\frac{1}{2})^i = \frac{1 - (\frac{1}{2})^k}{1 - \frac{1}{2}} = 2 - 2^{-k}$$

$$= n \times (2 - 2^{-k}) \quad ; \quad 2^{-k} = n \quad \log n = k$$

$$= n \times (2 - 2 \cdot \frac{1}{n})$$

$$= 2n - 2$$

$$\Rightarrow T(n) = 2n - 2 + T(1) \rightarrow \text{Base case}$$

$T(n) = O(n)$, * Bir önceki çözümde bir yede yarlılık gösterme
ve $(\frac{1}{2})^{k-1}$ eksikliği

$$\underline{\text{Ex}} \quad T(n) = n + 2T(n/2)$$

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

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

$$T(n) = n + n + n + 8(n/4 + 2T(n/8))$$

$$= n + n + n + 8T(n/8)$$

$$T(n) = n + n + n + 8[n/8 + 2T(n/16)]$$

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

$$T(n) = nk + 2^k T(n/2^k) \quad \text{Assume } 2^k = n \rightarrow \log n = k$$

$$= n \cdot \log n + n \cdot T(1) \quad T(1) = 1$$

$$= n \log n + n \rightarrow T(n) = O(n \log n)$$

$$\underline{\text{Ex}} \quad T(n) = n + T(n-1)$$

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

$$T(n) = n + n-1 + n-2 + \dots + 2 + T(n-k)$$

$$T(n) = n(n+1)/2 - 1 + T(n-k) \quad k=n$$

$$= \frac{n^2+n}{2} - 1 + T(0) \rightarrow T(n) = O(n^2)$$

* substitution method:

→ show that the solution of $T(n) = T(n-1) + n$ is $\mathcal{O}(n^2)$

$$\text{We guess } T(n) \leq c \cdot n^2$$

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

$$T(n) \leq c \cdot (n-1)^2$$

$$= cn^2 - 2cn + c + n$$

$$= cn^2 - n(1-2c) + c \leq cn^2$$

$$cn^2 - 3n + 4 \leq cn^2$$

$$T(n) = \mathcal{O}(n^2)$$

iteration methods:

$$T(n) = T(n-1) + n \rightarrow \Theta(n^2)$$

$$T(n) = T(n/2) + c \rightarrow \Theta(\log n)$$

$$T(n) = T(n/2) + n \rightarrow \Theta(n)$$

$$T(n) = 2T(n/2) + 1 \rightarrow \Theta(n)$$

\downarrow or doable
or doable

Ex $T(n) = c + T(n/2)$

$$T(n) = c + c + T(n/4)$$

$$T(n) = k \cdot c + T(n/2^k)$$

Assume $2^k = n \quad k = \log n$

$$T(n) = c \log n + T(1) \rightarrow \text{Base case}$$

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

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

$$T(n) = n + n/2 + n/4 + T(n/8)$$

$$T(n) = n + n/2 + n/4 + n/8 + \dots + n/2^k + T(n/2^k)$$

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

$$= n \sum_{i=0}^{k-1} \left(\frac{1}{2}\right)^i = \frac{1 - \frac{1}{2^k}}{1 - \frac{1}{2}} = 2 - \frac{1}{2^k} = n(2 - \frac{1}{2^k})$$

$$T(n) = 2n - 2 \cdot n + T(n/2^k) \quad k = \log n$$

$$T(n) = 2n - 2^{\log n} + T(1) \Rightarrow T(n) = 2n - \frac{1}{n} + T(1)$$

$\mathcal{O}(n)$ because $2^{\log n}$

Ex Func sum(int n) int {

if $n = 1 \Rightarrow 1$

return 1

{

return $n + \text{func}(n-1)$;

}

↓

$T(1) = 1$

$T(n) = 3 + T(n-1)$

$T(n) = 6 + T(n-2)$

$T(n) = 9 + T(n-3)$

⋮
 $T(n) = 3k + T(n-k)$

$$T(n) = 3n - 3 + T(1)$$

$$\left\{ \begin{array}{l} n-k=1 \\ k=n-1 \end{array} \right.$$

$$T(n) = 3n - 3 + 1 = T(n) = 3n - 2 \quad T(n) = O(n)$$

Ex complexity of Recursive Algorithm - The iteration Method

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

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

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

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

$$= C + C + C + T(n/8) \quad T(n) = 3C + T(n/8)$$

Assume $n = 2^k$ Assume n :

$$T(n) = C \log n + T(1) \quad T(n) = k.C + T(n/2)$$

$$= O(C \log n)$$

Assume $n = 2^k$

$$T(n) = \log n.C + T(n/2)$$

$$T(n) = \log n.C + T(1)$$

$$\Leftarrow T(n) = \log n.C + T(1)$$

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

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

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

$$= n + n + 4(n/4 + 2T(n/8))$$

$$= n + n + n + 8T(n/8)$$

$$= kn + 2^k T(n/2^k)$$

→ Assume $n = 2^k$

$$\rightarrow n \log n + nT(1) = O(n \log n)$$

Ex $T(n) = 1$ if $n=1$

$$T(n) = 2^3 T(n-8)$$

$$T(n) = 2T(n-1) \text{ if } n > 1$$

$$T(n) = 2^k T(n-k)$$

$$T(1) = 1$$

$$T(1) = 1 \Rightarrow n-k=1 \Rightarrow k=n-1$$

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

$$T(n) = 2^{n-1} + 1$$

$$T(n) = 2 \cdot 2T(n-2)$$

$$O(2^n)$$

Recursion Time complexity

int factorial (int n)

{ \rightarrow 1 unit of time $T(0) = 1$

if (n == 0)

$$T(n) = 3 + T(n-1) \text{ if } n > 1$$

{

return 1;

$$T(n-1) = 3 + T(n-2)$$

{

return n * factorial (n-1);

↓

1 unit of time

1 unit of time

$$T(n) = 3 + T(n-1) =$$

$$T(n) = 6 + T(n-2)$$

$$T(n) = 9 + T(n-3)$$

$$\vdots$$

$$T(n) \leq 3 \cdot k + T(n-k) \Rightarrow T(0) \leq 1 \quad \therefore n-k=0$$

$$T(n) = 3n + T(n-n) \Leftarrow n=k$$

$$T(n) = 3n + T(0)$$

$$T(n) = 3n + 1$$

int fibonaci (int n)

{

if (n <= 1)

return n;

return Fibonaci (n-1) + Fibonaci (n-2);

{ $\mathcal{O}(2^n)$

int Fibonaci (int n)

{

int n1, n2, n3;

for (int i=2; i<=n; i++)

{

$$n_3 = n_1 + n_2;$$

$$n_1 = n_2$$

$$n_2 = n_3$$

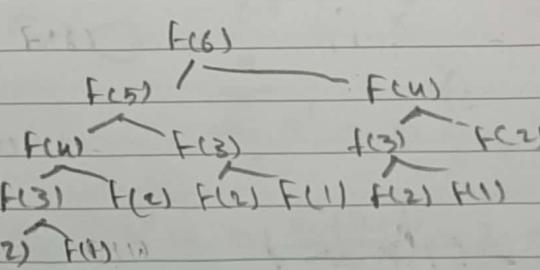
$\mathcal{O}(n)$

{

if (n <= 1)

return n;

$$n_1 = 0, n_2 = 1;$$



F(3) called 3 time

F(1) called 3 time

F(2) called 5 time

Ex for ($i=1$; $i \leq n$; $i++$)

{

for ($j=1$; $j \leq i$; $j++$)

{

for ($k=1$; $k \leq j$; $k++$)

{

$x = x + 1$;

}

{

$$\sum_{i=1}^n \sum_{j=1}^i \sum_{k=1}^j 1$$

$$= \sum_{i=1}^n \sum_{j=1}^i j$$

$$= \sum_{i=1}^n \frac{i(i+1)}{2}$$

$$\sum_{i=1}^n \frac{i^2 + i}{2} = \frac{1}{2} \sum_{i=1}^n i^2 + \frac{1}{2} \sum_{i=1}^n i = \frac{1}{2} \left[\frac{n(n+1)(2n+1)}{6} + \frac{n(n+1)}{2} \right]$$

$\mathcal{O}(n^3)$

Ex for $i=1$ to n do

for $j=i$ to n do

for $k=j$ to j do

$$m = i + j + k + m$$

$$\sum_{i=1}^n \sum_{j=i}^n \sum_{k=j}^j 1 = \sum_{i=1}^n \sum_{j=i}^n j$$

$$\sum_{i=1}^n \frac{n(n+1)}{2} - \frac{i(i-1)}{2} = \frac{1}{2} \sum_{i=1}^n n^2 + \frac{1}{2} \sum_{i=1}^n n + \sum_{i=1}^n \frac{i^2}{2} - \frac{1}{2} \sum_{i=1}^n i$$

$$\frac{1}{2} n^3 + \frac{3}{2} n^2 + \frac{1}{2} \left(\frac{n(n+1)(2n+1)}{6} \right) - \frac{1}{2} \left(\frac{n(n+1)}{2} \right)$$

$$6n^3 + 6n^2 + (n^2 + n)(2n+1) - 6n^2 - 6n$$

$$6n^3 + 6n^2 + \cancel{2n^3} + \cancel{n^2} + \cancel{2n^2} + n - \cancel{6n^2} - \cancel{6n} = 8n^3 + 3n^2 - 5n$$

$$= \mathcal{O}(n^3)$$

Ex for (int i=0; i<N; i++)

$$\sum_{i=0}^n \sum_{i+1}^n 1$$

for (int j=i; j<N; j++)

stmt;

$$\sum_{i=0}^n n - i = \sum_{i=0}^n (n-i)$$

Σ

$$\sum_{i=0}^n n - \sum_{i=0}^n i = n^2 - n(n+1)/2$$

$$= \frac{2n^2 - n^2 - n}{2} = \frac{n^2 - n}{2}$$

$O(n^2)$

Ex for (int i=0; i<N; i++)

for (int j=i; j<N; j++)

for (int k=j; k<N; k++)

stmt;

$$\sum_{i=0}^n \sum_{j=i}^n \sum_{k=j}^n 1 = \sum_{i=0}^n \sum_{j=i+1}^n (n-j+1)$$

$$\sum_{i=0}^n \sum_{j=i+1}^n (n-j) = \sum_{i=0}^n \sum_{j=i+1}^n n - \sum_{i=0}^n \sum_{j=i+1}^n j$$

$$\sum_{i=0}^n n(n-i) - \sum_{i=0}^n \frac{n(n+1)}{2} \cdot \frac{(i+1)(i+1-1)}{2}$$

$$\sum_{i=0}^n n^2 - ni = \sum_{i=0}^n \frac{n^2 + n}{2} - \frac{i^2 + i}{2}$$

$$\sum_{i=0}^n n^2 - n \sum_{i=0}^n i - \frac{1}{2} \sum_{i=0}^n i^2 - \frac{1}{2} \sum_{i=0}^n n + \frac{1}{2} \sum_{i=0}^n i^2 + \frac{1}{2} \sum_{i=0}^n i$$

$$n^3 - n \cdot \frac{n(n+1)}{2} - \frac{1}{2} n^3 - \frac{1}{2} n^2 + \frac{1}{2} \frac{n(n+1)}{2} + \frac{1}{2} \frac{n(n+1)(2n+1)}{6}$$

$$n^3 - \frac{n^3 + n^2 - n^3}{2} - \frac{n^2}{2} + \frac{n^2 + n}{2} + \frac{n(n+1)(2n+1)}{6}$$

$$\frac{n^3}{2} - \frac{n^2}{2} - \frac{n^2}{2} - \frac{n^2}{2} + \frac{n^2}{2} + \frac{n}{2} = \frac{2n \cdot n(n+1)(2n+1)}{6} = O(n^3)$$

Ex $x=0$
for $i=1$ to n do $\sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^n 1$
 for $j=1$ to n do
 for $k=1$ to n do
 $x := x + 1$ $\sum_{i=1}^n \sum_{j=1}^n n = \sum_{i=1}^n n^2 = n^3$

(formula: $\sum_{i=1}^n 1 = n$)

Ex int $i, x=0;$
for ($i=n*n$; $i<=n*n*n$; $i+=3$) {
 $x+=1;$
 }
 $\sum_{i=n^2}^{n^3} 1 = n^3 - n^2 + 1 = F(n)$
 $f(n) \in O(n^3)$

Ex show that $5n$ belongs to $O(n)$

$$5n \leq cn$$

$$\frac{1}{5} \leq c$$

$$c=1$$

$$n \geq 1$$

$$c=6$$

$$5n \leq 6n$$

$$5 \leq 6$$

$$\therefore f(n) \in O(n)$$

* $\sum_{i=1}^n i \approx n^2$ * $\sum_{i=0}^{n-1} 2^i = 2^n - 1$ * $\sum_{i=1}^n i^2 \approx n^3$

Ex for $k=1$ to $n/2$ do $\sum_{k=1}^{n/2} 1 + \sum_{j=1}^{n^2} 1 = \frac{n}{2} + n^2 = O(n^2)$

{

g:

for $j=1$ to n do

{

3 --

Ex for $k=1$ to $n/2$ do

{

for $j=1$ to n do

{

3 --

$$\sum_{k=1}^{n/2} \sum_{j=1}^{n^2} 1 = \sum_{k=1}^{n/2} n^2$$

$$n^2 \sum_{k=1}^{n/2} 1 = n/2 \cdot n^2 = O(n^3)$$

Ex $f(n) = 4^n$, $g(n) = 8^n$ prove

$$4^n \leq C \cdot 8^n \text{ for all } n \geq 2 \quad 4^n \leq \frac{1}{n} \cdot 8^n \text{ for } n \geq 2$$

$$\frac{4^n}{8^n} \leq C \text{ for all } n \geq 2 \quad 1 \leq \frac{2^n}{4} \text{ for } n \geq 2$$

$$\frac{1}{2^n} \leq C \text{ for all } n \geq 2 \quad 1 \leq 2^{n-2} \quad \begin{cases} n=2 \\ n \geq 2 \end{cases}$$

$$\text{choose } (C = \frac{1}{4}, k = 2) \quad f(n) \in O(g(n))$$

$$4^n \in O(8^n)$$

Ex $f(n) = 3n+1$, $g(n) = n$ prove $f(n) = \Omega(g(n))$

$$3n+1 \geq Cn$$

$$3 + \frac{1}{n} \geq C \quad C=2 \text{ choose}$$

$$k=5 \text{ choose}$$

$$3 + \frac{1}{n} \geq 2$$

$$3 + \frac{1}{5} \geq 2 \quad 3.2 \geq 5 \rightarrow f(n) = \Omega(n)$$

Ex prove 2^{n+1} is $O(2^n)$

$$2^{n+1} \leq C \cdot 2^n \quad 2^{n+1} \leq C \cdot 2^n \quad n \geq k$$

$$1 \cdot 2 \leq C$$

$$2^{n+1} \leq 3 \cdot 2^n \quad n \geq 1$$

$$2^{n+1} \leq 2 \cdot 2^n \quad \text{for all } n \geq 1 \quad f(n) \in O(g(n)) \text{ choose } C=3$$

Ex init i, j, y, = 0;

for ($i=1$; $i \leq 2^n$; $i++$) {

 for ($j=1$; $j \leq i$; $j++$) {

$$y = y + 2$$

$$\sum_{i=1}^{2^n} \sum_{j=1}^i 1 = \sum_{i=1}^{2^n} i = \frac{2^n(2^n+1)}{2}$$

{}

$$f(n) = \frac{4n^2 + 2n}{2} = 2n^2 + n$$

Ex $f(n) = O(g(n))$

- if $f(n) \leq c_1 g(n)$ for all $n \geq k$
when c_1 are positive
- $\frac{f(n)}{g(n)} \leq c$ for all $n \geq k$

Show that $f(n) = n^2 + 2n + 1$ is $O(n^2)$

$$\frac{n^2 + 2n + 1}{n^2} \leq c \text{ for all } n \geq 1$$

$$\frac{1+2+1}{1} \leq c \Rightarrow 4 \leq c$$

$$\frac{n^2 + 2n + 1}{n^2} \leq c \Rightarrow n^2 + 2n + 1 \leq cn^2 \text{ for all } n \geq 1$$

$$\Rightarrow f(n) = O(g(n)) = O(n^2)$$

Ex

for ($i=0$; $i < n$; $i++$) {

 for ($j=i+1$; $j < n-1$; $j++$) {

 loop body

$$\sum_{i=0}^{n-1} \sum_{j=i+1}^{n-1} 1$$

$$\sum_{i=0}^{n-1} ((n-1)-(i+1)+1) = n-1-i-1$$

$$\sum_{i=0}^{n-1} (n-1-i) = \sum_{i=0}^{n-1} n - \sum_{i=0}^{n-1} 1 - \sum_{i=0}^{n-1} i$$

$$(n^2 - n - \frac{n(n-1)}{2}) = n^2(2n^2 - 2n - n^2 - n) = \frac{3n^2 - 3n}{2}$$

$$f(n) = O(n^2)$$

Ex $f(n) = 2n - 2$ $g(n) = n$ prove $f(n) = O(g(n))$

$$2n - 2 \leq cn$$

$$\text{choose } c = 1, k = 3$$

$$2n - 2 \leq 3n$$

$$2n \leq 3n + 2$$

$$1 \leq \frac{3}{2} + \frac{1}{n} \text{ for all } n \geq 1$$

$$f(n) = O(n)$$

summation formulas:

$$\sum_{i=1}^n ar^{i-1} = a \left(\frac{1-r^n}{1-r} \right) \quad * \quad \sum_{i=m}^n c = c(n+1-m)$$

$$\sum_{i=m}^n i = \frac{n(n+1)}{2} - \frac{m(m-1)}{2} \quad * \quad \sum_{i=1}^n i = \frac{n(n+1)}{2}$$

$$\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6} \quad * \quad \sum_{i=0}^n x^i = \frac{(x^{n+1}-1)}{x-1}$$

$$\sum_{0 \leq i \leq n} 2^i = \frac{2^{n+1}-1}{2-1} = 2^{n+1}-1$$

$$\text{Ex int count = 0; } T(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} c$$

for (int i=0; i<n; i++)

 for (int j=0; j<i; j++)

 count++

$$= \sum_{i=0}^{n-1} (i-0+1).c$$

$$= \sum_{i=0}^{n-1} c(i+1) = c \sum_{i=0}^{n-1} i = c \left(\frac{(n-1)n}{2} - \frac{3 \cdot 2}{2} \right) = c \cdot \frac{(n^2-n-6)}{2}$$

$$T(n) = O(n^2)$$

cost: time

Ex $i=1$

sum = 0

while ($i < n$)

{

$$j=1; \quad C_4 \quad n \quad 3C_{11}n^2 + 4C_{10}n + 3C_9 = T(n)$$

 while ($j < n$) $C_5 \quad n(n+1)$

{

 sum += 1; $C_6 \quad n \times n$

 j = j + 1; $C_7 \quad n \times n$

{

 i += 1; $C_8 \quad n$

{

Inception Sort

* void InceptionSort (int [] arrNumbers)

{

n+1 For (int i=1; i < arrNumbers.Length; i++)

{

n int j = i-1;

n int key = arrNumbers[i];

n*xn while (j >= 0 && key < arrNumbers[j])

{

n*xn arrNumbers[j+1] = arrNumbers[j];

j = j-1

{

n arrNumbers[j+1] = key;

2n²+n+1 {

60, 34, 25, 12, 22

Worst O(n²)

60, 60, 25, 12, 22

Best O(n)

34, 60, 25, 12, 22

34, 60, 60, 12, 22

j=-1 34, 34, 60, 12, 22

i ↗ 25, 34, 60, 12, 22

25, 34, 60, 60, 22

25, 34, 34, 60, 22

j=-1 25, 25, 34, 60, 22

↗ 12, 25, 34, 60, 22

12, 25, 34, 60, 60

12, 25, 34, 60, 60

12, 25, 84, 84, 60

12, 25, 25, 34, 60

12, 22, 25, 34, 60

Selection Sort

void SelectionSort (int [] arr) ترتيب ورق

{

```
int minIndex = 0;
```

```
for (int i=0; i < arr.Length - 1; i++)
```

{ minIndex = i;

```
for (int j=i+1; j < arr.Length; j++)
```

{

```
if (arr[i] > arr[j])
```

{

```
swap (arr[i], arr[j]);
```

```
minIndex = j;
```

{

{

```
swap (arr[j], arr[i]);
```

{

Worst case $O(n^2)$

Best case $O(n^2)$

Bubble Sort

void BubbleSort (int [] arr)

{

```
for (int i=0; i < arr.Length; i++) {
```

```
for (int j=0; j < arr.Length; j++) {
```

```
if (arr[j] > arr[j+1]) {
```

```
swap (arr[j], arr[j+1]);
```

{

{

Best case and worst case $O(n^2)$

Alg1 int LinearSearch (int[] arrNumbers, int value)

```

    {
        for (int i=0; i<arrNumbers.length; i++) n+1
        {
            if (arrNumbers[i] == value) n
            {
                return i; 1
            }
        }
        return -1; f(n) = O(n)
    }

```

Best $O(1)$, Average $O(n)$, Worst $O(n)$

Alg2

int BinarySearch (List<int> lstNumbers, int value)

```

    {
        int start = 0;
        int End = lstNumbers.Count;
        int mid;
        for each number in list
        while (End >= start)
        {
            mid = (start + End) / 2; k = n
            if (lstNumbers[mid] == value) b = log n
            {
                return mid; O(log n) = f(n)
            }
        }

```

- Best $O(1)$
- Worst $O(\log n)$
- Average $O(\log n)$

if (lstNumbers[mid] > value)

```

    {
        End = Mid - 1; 3
    }

```

if (lstNumbers[mid] < value)

```

    {
        Start = Mid + 1; 3
    }

```

- Algoritmaların doğrudan 10, 20 sayıları gibi çok fazla girdiye dek hizli değiştiğine, ne kadar kötü duruma gittiği örenmek.

	Time	cost
<u>Ex</u> $i=1;$	1 unit of time	c_1
$\text{sum} = 0;$	1	c_2
$\text{while } (i \leq n) \{$	$n+1$	c_3
$j=1;$	n	c_4
$\text{while } (j \leq n) \{$	$n(n+1) = n^2+n$	c_5
$\text{sum} = \text{sum} + j;$	n	c_6
$j=j+1;$	n	c_7
$\}$	$n \times n$	دعا داده ایشان طبق ماتریسی
$i=i+1;$		
$\}$		
$2n^2 + 5n + 3 = f(n)$	$\rightarrow O(n^2) = f(n)$	

* Asimetrik Notasyonların karşılaştırması:

"Geçerlik"

$$f(n) = \underset{\Omega}{\Theta}(g(n)) \text{ ve } g(n) = \underset{\Omega}{\Theta}(h(n)) \text{ ise } f(n) = \underset{\Omega}{\Theta}(h(n))$$

"Döngüdeki veya 'yansıma'"

$$\alpha f(n) = \Theta(f(n)), \quad \omega f(n) = \Theta(f(n)), \quad \Omega f(n) = \Omega(f(n))$$

"simetri"

$$g(n) = \Theta(h(n)) \text{ olduğu durumda } f(n) = \Theta(g(n))$$

Transpose symmetry

$$g(n) = \Omega(f(n)) \text{ olduğu durumda } f(n) = \Theta(g(n))$$

$$g(n) = \omega(f(n)) \text{ olduğu durumda } f(n) = \Theta(g(n))$$

Ex For ($i = \frac{n}{2}; i \leq n; i++$)
 For ($j = 1; j \leq n; j = 2 \times j$)
 For ($k = 1; k \leq n; k = k \times 2$) $\log n$
 Stmt;
 $f(n) = O[n(\log n)^2]$

Ex For ($i = 1; i \leq n; i++$)
 For ($j = 1; j \leq n; j = j + i$)
 Stmt;
 i j
 1 $1 \rightarrow n$ $n/2$ time
 2 $1 \rightarrow n$ $n/3$
 3 $1 \rightarrow n$ $n/4$
 ;
 $n(1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n})$ $\leftarrow n$ $n \rightarrow n$ n/n
 $n(\log n) \Rightarrow f(n) = O(n \log n)$

Ex int $n = 2^{2^k}$
 For ($i = 1; i \leq n; i++$)
 {
 For ($j = 2; j \leq n; j++$)
 while ($j \leq n$)
 {
 Stmt;
 }
 $j = j^2$
 Stmt;
 }
 $2^k = n$
 $\log n = 2^k$
 $O(n \log n) \Leftrightarrow \log n = k$

Ex For ($i = 0; i \leq n; i++$)
 {
 For ($j = 0; j \leq n; j++$)
 {
 $c[i, j] = 0;$
 For ($k = 0; k \leq n; k++$)
 {
 $c[i, j] = A[i, j] + B[i, k]$ $n \times n = n^2$
 $n \times n \times (n+1) = n^3 + n^2$
 $c[i, j] = A[i, j] + B[i, k]$ $n \times n \times n = n^3$
 $2n^3 + 3n^2 + 2n + 1 = f(n) \rightarrow O(n^3)$
 }
 }

Ex

```
int i, j, k, n;
```

```
for (i=1; i<=n; i++)
```

{

```
    for (j=1; j<=i; j++)
```

{

```
        for (k=1; k<=100; k++)
```

{

stmt;

i	j	k
1	1	1x100

2	2	2x100
---	---	-------

3	3	3x100
---	---	-------

$$100 \left(\frac{n(n+1)}{2} \right) \Rightarrow O(n^2)$$

}

}

{

Ex int i, j, k, n;

```
for (i=1; i<=n; i++)
```

{

```
    for (j=1; j<=i2; j++)
```

{

```
        for (k=1; k<=n/2; k++)
```

{

stmt?

i	j	k
1	1	n/2.1

2	4	n/2.4
---	---	-------

3	9	n/2.9
---	---	-------

$$(1+4+9+\dots+n^2) = \frac{n}{2} \left(\frac{n(n+1)(2n+1)}{6} \right)$$

$$= O(n^3)$$

}

{

Ex for (i=1; i<n; i=i*2)

{

stmt;

{

 $i = 1, 2, 4, \dots, 2^k$

$$2^k = n \Rightarrow O(\log n)$$

Ex for (i=n/2; i<n; i++)
 $\frac{n}{2}$

```
    for (j=1; j<=n/2; j++)
```

 $\frac{n}{2}$

```
        for (k=1; k<=n; k=k*2) logn
```

stmt;

$$O(n^2 \log n)$$

Exi=0while (i < n)
 $i++$ stmt:
 $i++$

$$f(n) = 3n + 2$$

$$F(n) = O(n)$$

 $\alpha = 1$ while ($a < b$)
 $a = a * 2$ stmt:
 $a = a * 2$

$$2^k \geq b \Leftrightarrow k \geq \log_2 b$$

$$k = \log_2 b \Rightarrow f(n) = O(\log b)$$

Ex i=1; $k=1$ while ($k < n$){stmt: $k=k+i$ $i++$ }For (i=1, k=1; k < n; i++){ $k = k + i;$ }Ex while ($m != n$){if ($m > n$) $m = m - n;$ else $n = n - m;$ } $m = 16 \quad n = 2$ 16212210282624222

$$\frac{m(m+1)}{2} + m + 1 \geq n$$

$$m^2 \geq n$$

$$m = \sqrt{n}$$

$$f(n) = O(\sqrt{n})$$

Ex Agorithm test (n){if ($n < 5$)print (n); $\rightarrow O(1)$ elsefor (i=0; i < n; i++) i++ + i + ... + i = n * i = O(n)print (i);Best: O(1)Worst O(n)

Ex $P=0$

for ($i=1, i \leq n, i=i*2$) $1, 2, 2^2, 2^3 \rightarrow 2^k \geq n \rightarrow O(\log n)$

{}

$P++;$ $\rightarrow O(\log n)$

{}

 $P = \log n$

for ($j=1; j \leq P; j=j*2$) $1, 2, 2^2, 2^3, \dots, 2^m \geq P \rightarrow m = \log P$

{}

stmt; $O(\log \log n)$

{}

Ex for ($i=0, i < n, i++$) $(n+1) \rightarrow 2n \cdot \log n + n+1$

{}

for ($j=1, j \leq n, j=j*2$) $n \cdot \log n \rightarrow O(n \cdot \log n)$

{}

stmt; $n \cdot \log n$

{}

{}

- for ($i=0, i < n, i++$) $\rightarrow O(n)$

- for ($i=0, i < n, i+=2$) $\rightarrow \frac{n}{2} \rightarrow O(n)$

- for ($i=n, i>1, i-1$) $\rightarrow O(n)$

- for ($i=1, i < n, i=1*2$) $\rightarrow O(\log n)$

- for ($i=1, i < n, i=i*5$) $\rightarrow O(\log_5 n)$

- for ($i=n, i>1, i=i/2$) $\rightarrow O(\log_2 n)$

- for ($i=n/2, i < n, i++$) $\rightarrow \frac{n}{2} \rightarrow O(\log n)$

$P = 0$

Ex $\text{for } (i=1; p < n; i++)$

{

$$P = P + i$$

}

i P

$$1 \quad 1$$

$$2 \quad 1+2$$

$$3 \quad 1+2+3$$

:

!

n

$$1+2+\dots+k$$

$P > n$

$$\frac{k(k+1)}{2} > n$$

$$k^2 > n$$

$$k > \sqrt{n}$$

Ex $\text{for } (i=1; i < n; i=i*2)$

{

stmt;

}

i

1

$$2^k > n$$

$$1 \times 2$$

$$k > \log n$$

$$1 \times 2 \times 2$$

$$k = \log n$$

Ex $\text{for } (i=1; i \geq n; i++)$

{

stmt;

}

$$i = 1+1+1+\dots = 1 \times n = k$$

$$\log(n)$$

2^k

$$O(\log n) = f(n)$$

Ex $\text{for } (i=n; i \geq 1, i=i/2)$

{

stmt;

}

i

n

$\frac{n}{2}$

$$n < \frac{1}{2^n}$$

$\frac{n}{4}$

$$k = \log n \rightarrow O(\log n)$$

$\frac{n}{8}$

$\frac{n}{16}$

$\frac{n}{32}$

$\frac{n}{64}$

$\frac{n}{128}$

$\frac{n}{256}$

$\frac{n}{512}$

$\frac{n}{1024}$

$\frac{n}{2048}$

$\frac{n}{4096}$

$\frac{n}{8192}$

$\frac{n}{16384}$

$\frac{n}{32768}$

$\frac{n}{65536}$

$\frac{n}{131072}$

$\frac{n}{262144}$

$\frac{n}{524288}$

$\frac{n}{1048576}$

$\frac{n}{2097152}$

$\frac{n}{4194304}$

$\frac{n}{8388608}$

$\frac{n}{16777216}$

$\frac{n}{33554432}$

$\frac{n}{67108864}$

$\frac{n}{134217728}$

$\frac{n}{268435456}$

$\frac{n}{536870912}$

$\frac{n}{1073741824}$

$\frac{n}{2147483648}$

$\frac{n}{4294967296}$

$\frac{n}{8589934592}$

$\frac{n}{17179869184}$

$\frac{n}{34359738368}$

$\frac{n}{68719476736}$

$\frac{n}{137438953472}$

$\frac{n}{274877906944}$

$\frac{n}{549755813888}$

$\frac{n}{1099511627776}$

$\frac{n}{2199023255552}$

$\frac{n}{4398046511104}$

$\frac{n}{8796093022208}$

$\frac{n}{17592186044416}$

$\frac{n}{35184372088832}$

$\frac{n}{70368744177664}$

$\frac{n}{140737488355328}$

$\frac{n}{281474976710656}$

$\frac{n}{562949953421312}$

$\frac{n}{1125899906842624}$

$\frac{n}{2251799813685248}$

$\frac{n}{4503599627370496}$

$\frac{n}{9007199254740992}$

$\frac{n}{18014398509481984}$

$\frac{n}{36028797018963968}$

$\frac{n}{72057594037927936}$

$\frac{n}{144115188075855872}$

$\frac{n}{288230376151711744}$

$\frac{n}{576460752303423488}$

$\frac{n}{1152921504606846976}$

$\frac{n}{2305843009213693952}$

$\frac{n}{4611686018427387904}$

$\frac{n}{9223372036854775808}$

$\frac{n}{18446744073709551616}$

$\frac{n}{36893488147419103232}$

$\frac{n}{73786976294838206464}$

$\frac{n}{147573952589676412928}$

$\frac{n}{295147905179352825856}$

$\frac{n}{590295810358705651712}$

$\frac{n}{1180591620717411303424}$

$\frac{n}{2361183241434822606848}$

$\frac{n}{4722366482869645213696}$

$\frac{n}{9444732965739290427392}$

$\frac{n}{18889465931478580854784}$

$\frac{n}{37778931862957161709568}$

$\frac{n}{75557863725914323419136}$

$\frac{n}{15111572745828646683832}$

$\frac{n}{30223145491657293367664}$

$\frac{n}{60446290983314586735328}$

$\frac{n}{120892581966629173470656}$

$\frac{n}{241785163933258346941312}$

$\frac{n}{483570327866516693882624}$

$\frac{n}{967140655733033387765248}$

$\frac{n}{1934281311466066775530496}$

$\frac{n}{3868562622932133551060992}$

$\frac{n}{7737125245864267102121984}$

$\frac{n}{15474250491728534204243968}$

$\frac{n}{30948500983457068408487936}$

$\frac{n}{61897001966914136816975872}$

$\frac{n}{123794003933828273633951744}$

$\frac{n}{247588007867656547267903488}$

$\frac{n}{495176015735313094535806976}$

$\frac{n}{990352031470626189071613952}$

$\frac{n}{1980704062941252378143227904}$

$\frac{n}{3961408125882504756286455808}$

$\frac{n}{7922816251765009512572911616}$

$\frac{n}{15845632533320018241548223232}$

$\frac{n}{31691265066640036483096446464}$

$\frac{n}{63382530133280072966192892928}$

$\frac{n}{126765060266560145932385785856}$

$\frac{n}{253530120533120291864771571712}$

$\frac{n}{507060241066240583729543143424}$

$\frac{n}{101412048213248116745888628848}$

$\frac{n}{202824096426496233491777257696}$

$\frac{n}{405648192852992466983554515392}$

$\frac{n}{811296385705984933967109030784}$

$\frac{n}{1622592771411969867934218061568}$

$\frac{n}{3245185542823939735868436123136}$

$\frac{n}{6490371085647879471736872246272}$

$\frac{n}{12980742171295758943473744492544}$

$\frac{n}{25961484342591517886947488985088}$

$\frac{n}{51922968685183035773894977970176}$

$\frac{n}{103845937370366071547789955940352}$

$\frac{n}{207691874740732143095579911880704}$

$\frac{n}{415383749481464286191159823761408}$

$\frac{n}{830767498962928572382319647522816}$

$\frac{n}{1661534977925857144764639295055632}$

$\frac{n}{3323069955851714289529278590111264}$

$\frac{n}{6646139911703428578058557180222528}$

$\frac{n}{13292279823406857156117114360445056}$

$\frac{n}{26584559646813714312234228720890112}$

$\frac{n}{53169119293627428624468457441780224}$

$\frac{n}{106338238587254857248936914883560448}$

$\frac{n}{212676477174509714497873829767120896}$

$\frac{n}{425352954349019428995747659534241792}$

$\frac{n}{850705908698038857991495319068483584}$

$\frac{n}{1701411817396077715982990638136967168}$

$\frac{n}{3402823634792155431965981276273934336}$

$\frac{n}{6805647269584310863931962552547868728}$

$\frac{n}{13611294539168621727863925105095737456}$

$\frac{n}{27222589078337243455727850210191474912}$

$\frac{n}{54445178156674486911455700420382949824}$

$\frac{n}{108890356313348973822911400840765899648}$

$\frac{n}{217780712626697947645822801681531799296}$

$\frac{n}{435561425253395895291645603363063598592}$

$\frac{n}{871122850506791790583291206726127197184}$

$\frac{n}{1742245701013583581166582413452254394368}$

$\frac{n}{3484491402027167162333164826904508788736}$

$\frac{n}{6968982804054334324666329653809017577472}$

$\frac{n}{13937965608108668649332659307618035154848}$

$\frac{n}{27875931216217337298665318615236070309696}$

$\frac{n}{55751862432434674597330637230472140619392}$

$\frac{n}{111503724864669349194661274460944281238784}$

$\frac{n}{223007449729338698389$

* Frequency Time method

Arithmetic operations \rightarrow 1 unit of time ($+, -, \times, \div$)

Assignment operator \rightarrow 1 unit of time ($x=5$)

Comparison \rightarrow 1 unit of time ($x>y$)

return statement \rightarrow 1 unit of time

Ex $\text{return } a+b; \quad O(1)$

Ex $i=1; \quad i \quad \text{if}$
 $n=10; \quad i \quad (1 \times 2) \quad \left. \begin{array}{l} \\ 1 \times 2 \\ 1 \times 2 \times 2 \\ \vdots \\ 2^k \end{array} \right\} \Rightarrow \text{Ans} = \log n + 2$
 $\text{while}(i < n)$
 $\{ \quad i = i + 1 \quad 1 \times 2 \times 2 \times \dots \times 2^k \quad f(n) = O(\log n)$
 $i * = 2 \rightarrow 2^k$
 $\exists \quad : \quad - \quad 2^k \quad , \quad 2^k > n$
 $2^k = n$
 $k = \log n$

Ex $\text{sum} = 0$
 $\text{for } (i=0; i < n; i++) \quad \{ \quad n+1 \quad \left. \begin{array}{l} \\ n \\ n \end{array} \right\} \Rightarrow 2n+2 = f(n)$
 $\text{sum} = \text{sum} + i; \quad n \quad f(n) = O(n)$

Ex $\text{for } (i=0; i < n; i++)$
 $\{ \quad \text{for } (j=0; j < i; j++)$
 $\{ \quad \text{stmt}; \quad \left. \begin{array}{l} b \\ 1 \\ 2 \\ \vdots \\ n \end{array} \right\} \quad \text{time}$
 $\} \quad \left. \begin{array}{l} 0^* \\ 0^- \\ 1x \\ 2 \\ \vdots \\ n \end{array} \right\}$

$$0+1+2+\dots+n = \frac{n(n+1)}{2}$$

$$f(n) = \frac{n^2}{2} + \frac{n}{2} \quad f(n) = O(n^2)$$

Ex prove that $2n^2 + 3n + 6 \neq O(n^3)$

$$C_1 n^3 \leq 2n^2 + 3n + 6 \leq C_2 n^3$$

$$2n^2 + 3n + 6 \leq C_2 n^3$$

$$C_1 n^3 \leq 2n^2 + 3n + 6$$

if one side disproved so no need to prove other side

$$\text{Hence this is not possible}$$

$$\text{and } C_1 n^3 \leq 2n^2 + 3n + 6 \text{ for } n \geq n_0 \text{ is not true for any}$$

Ex prove that $2n^2 + 5n + 6 = O(n^2)$

$$\downarrow f(n) \quad \downarrow g(n)$$

Ex proved that $2n+10 = O(n)$

$$f(n) \leq c.g(n)$$

$$2n^2 + 5n + 6 \leq Cg(n)$$

$$2n+10 \leq c.n$$

$$2n^2 + 5n + 6 \leq 2n^2 + 5n^2 + 6n^2$$

$$10 \leq cn - 2n$$

$$2n^2 + 5n + 6 \leq 13n^2$$

$$10 \leq n(c-2)$$

$$c=13 \quad n_0=1 \text{ is in}$$

$$\frac{10}{c-2} \leq n$$

$$c-2$$

Ex prove that $3n+2 = \Omega(n)$

$$c > 3, n_0 = 10$$

$$3n+2 \geq c g(n)$$

$$2n+10 \leq 12n$$

$$3n+2 \geq cn$$

$$c=12, n=1$$

$$2 \geq n(c-3)$$

$$\frac{2}{(c-3)} \geq n$$

$$n_0 = 2, c=4$$

Ex ~~Ex~~ $5n^2 + 2n - 3 = \Omega(n^2)$

$$f(n) \geq g(n).c$$

$$c.n^2 \geq 5n^2 + 2n - 3$$

We need to prove $2n-5 \geq 0$

Ex prove that $5n^2 = \Omega(n^2)$
 $f(n) \geq c g(n)$

$\boxed{c=5, n=2}$ Proved

$$5n^2 \geq nc$$

$$5n \geq c$$

$$n_0 = 1, c=5$$

Proved

Ex $100n + 5 \stackrel{?}{=} \Omega(n^2)$

$$f(n) \geq c g(n)$$

$$100n + 5 \geq c.n^2 \text{ it is wrong}$$

Hence this is not possible and $c.n^2 \leq 100n + 5$
for $n \geq n_0$ is not true for any combination
of c and n_0 .

Ex prove that $\frac{1}{2}n^2 - 3n = \Theta(n^2)$

Ex $T(n) = n^2 + n \geq cn^2$?

$$c_1 n^2 \leq \frac{1}{2} n^2 - 3n \leq c_2 n^2$$

$$\begin{aligned} n^2 + n &\geq cn^2 \\ n^2 + n &> c n^2 \quad c=1 \text{ Olson} \\ n=1 &\text{ Olson} \end{aligned}$$

$$c_1 \leq \frac{1}{2} + \frac{3}{n} \leq c_2$$

$$1^2 + 1 \geq 1$$

$$2 \geq 1 \quad n \geq n_0$$

$$\frac{1}{2} - \frac{3}{n} \geq c_1, \quad n \geq 7, \quad c_1 \leq \frac{1}{14}$$

لحوظة تدق الخوارزمية

لحوظة تدق الخوارزمية

c minimal possible value

k, n_0 يمكن أن يكون أي عدد

n = g(n) \sqrt{n}

Ex $T(n) = 2n^2 + 5n + 3$

"O(n)"

$$T(n) \leq c f(n), \quad n \geq n_0$$

$$n^2 = f(n), \quad c \geq 2, \quad n_0 \geq 1$$

"Ω(n)"

$$T(n) \geq c g(n)$$

$$2n^2 + 5n + 3 \geq c f(n) \Rightarrow T(n) = \Theta(n^2)$$

"Θ(n)"

$$f(n) = g(n) = n$$

$$T(n) \leq c f(n) \Leftrightarrow T(n) \leq \Omega(n) = n^2, \quad c \leq 2, \quad n_0 = 1$$

$$T(n) = O(n^2), \quad n_0 = 1 \text{ given } n \geq 1$$

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

Ex prove that $\frac{1}{2}n^2 - \frac{1}{2}n = \Theta(n^2)$

$$c_1 n^2 \leq \frac{1}{2} n^2 - \frac{1}{2} n \leq c_2 n^2$$

for lower bound:

* for upper bound

$$\frac{n^2}{2} - \frac{n}{2} \geq c_1 n^2$$

$$\frac{n^2}{2} - \frac{n}{2} \leq c_2 n^2$$

$$\text{if we assume } c_1 = \frac{1}{4}$$

$$c_1 = \frac{1}{4} \Rightarrow \frac{1}{2} n^2 - \frac{1}{2} n \leq \frac{n^2}{2}$$

$$\frac{n^2}{2} - \frac{n}{2} \geq \frac{1}{4} n^2$$

$$n \geq 1 \rightarrow c_2 = \frac{1}{2}, \quad n_0 = 1$$

$$c_1 = \frac{1}{4}, \quad n_0 = 2$$

proved //

proved //

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

$$\underline{\underline{Ex}} \quad f(n) = 2n^2 + 3n + 4$$

$$2n^2 + 3n + 4 \geq 1 \times n^2$$

$$\underline{\underline{Ex}} \quad f(n) = n^2 \log n + n$$

$$1 \times n^2 \leq 2n^2 + 3n + 4 \leq 8n^2$$

$$1 \times n^2 \log n \leq n^2 \log n + n \leq 10n^2 \log n$$

$$\Theta(n^2), \Omega(n^2), \mathcal{O}(n^2)$$

$$\mathcal{O}(n^2 \log n), \Omega(n^2 \log n), \Theta(n^2 \log n)$$

$$\underline{\underline{Ex}} \quad f(n) = n!$$

$$\underline{\underline{Ex}} \quad f(n) = \log n!$$

$$n! = n(n-1)(n-2) \dots 3 \cdot 2 \cdot 1$$

$$\log(1 \times 1 \times \dots \times 1) \leq \log(1 \times 2 \times \dots \times n) \leq \log(n)$$

$$1 \times 1 \times \dots \times 1 \leq 1 \times 2 \times \dots \times n \leq n \times n \times \dots \times n \quad 1 \leq \log n! \leq \log n^n$$

$$\Omega(1), \mathcal{O}(n^n)$$

$$1 \leq \log n! \leq n \log n$$

we cannot say that $n!$ is always greater than n^{10} and less than n^{100}

$$\Omega(1), \mathcal{O}(n \log n)$$

$$\underline{\underline{Ex}} \quad f(n) = 2n + 5 \in \mathcal{O}(n)$$

$$f(n) = 5n^2 - 3n \in \mathcal{O}(n^2)$$

$$2n \leq 2n+5 \leq 2n+5n$$

$$4n^2 \leq 5n^2 - 3n \leq 5n^2$$

thus $n > 4$ is in

$$2n \leq 2n+5 \leq 8n$$

thus $n > 1$ is in

$$\underline{\underline{Ex}} \quad \frac{n^2}{2} - \frac{n}{2} \stackrel{?}{=} \mathcal{O}(n^2)$$

$$\frac{n^2}{2} - \frac{n}{2} \leq \frac{n^2}{2} \quad (\forall n \geq 0) \quad c_1 = \frac{1}{2}$$

$$\frac{n^2}{2} - \frac{n}{2} \geq \frac{n^2}{2} - \frac{n}{2} \cdot \frac{n}{2} \quad (\forall n \geq 2) \quad c_2 = \frac{1}{4}$$

$\underline{\underline{Ex}}$ $n \neq \mathcal{O}(n^2)$ i.e. $c_1 n^2 \leq n \leq c_2 n^2 \Rightarrow$ only holds for $n \leq \frac{1}{c_1}$,

$$6n^3 \neq \mathcal{O}(n^2) \quad c_1 n^2 \leq 6n^3 \leq c_2 n^2 \quad \text{thus } n \leq c_2 / c_1$$

$$1 < \log n < \sqrt{n} < n < \log n^2 < n^3 < \dots < 2^n < 3^n < \dots$$

* Omega " Ω "

$$f(n) \geq C * g(n) \quad \forall n > n_0$$

$$\underline{\underline{Ex}} \quad f(n) = 2n+3$$

$$f(n) \geq \Omega(n)$$

$$\underline{\underline{Ex}} \quad 2n+5 \in \Omega(n) \text{ olduguunu gostermez.}$$

$$2n+3 \geq kn$$

$$\frac{2n+3}{n} \geq k \quad \forall n > n_0$$

$$f(n) \geq Cg(n)$$

$$2n+3 \geq 1 \times \log n$$

$$\frac{2n+3}{\log n} \geq 1 \quad \forall n > n_0$$

$$2n+5 \geq n$$

$$(n > 0) \text{ icin}$$

$$C=1, n_0=0$$

$$f(n) = \Omega(n)$$

$$f(n) = \Omega(\log n)$$

$$f(n) = \Omega(n^2) \times$$

$$\underline{\underline{Ex}} \quad 5n^2 - 3n = \Omega(n^2) \text{ olduguunu gostermez.}$$

$$(n > 0) \quad 5n^2 - 3n \geq n^2$$

$$C=1$$

$$n_0=0 \text{ degelerle tam saglar.}$$

$$\underline{\underline{Ex}} \quad 5n^2 = \Omega(n)$$

$$\underline{\underline{Ex}} \quad 100n + 5 = \Omega(n^2)$$

$$0 \leq Cn^2 \leq 100n + 5$$

$$100n + 5 \leq 100n + 5n \quad n > 1$$

$$Cn^2 \leq 105n$$

$$n(Cn - 105) \leq 0$$

$$n=0$$

$$f(n) = 5n^2$$

$$n_0=0$$

n. can't be smaller than a constant (n_0)

Theta Θ :

$$c_1 g(n) \leq f(n) \leq c_2 g(n)$$

$$f(n) = 2n+3$$

$$cn < 2n+3 < 5n$$

$$f(n) = O(n)$$

$$c_1 g(n) \quad f(n) \quad c_2 g(n)$$

$$\underline{\underline{Ex}} \quad f(n) = 2n^2 + 3n + 4 \quad f(n) = ? \quad O(n)$$

$$2n^2 + 3n + 4 \leq 2n^2 + 3n^2 + 4n^2$$

$$2n^2 + 3n + 4 \leq cn^2$$

$$\downarrow \downarrow$$

$$(n > 1)$$

$$c g(n) \Rightarrow f(n) = O(n^2)$$

lower bound

upper bound

$$1 \leq \log n \leq \ln n \leq h \leq n \log n \leq h^2 \leq n^3 \leq \dots \leq 2^n \leq 3^n \leq n^n$$

avg bound

- $O(1)$ constant
- $O(n)$ linear
- $O(n^2)$ quadratic
- $O(n^3)$ cubic
- $O(\log n)$ logarithmic

$$\text{Ex } f(n) = 2n + 3 \quad f(n) = O(n)?$$

$$2n+3 \leq 2n+3n$$

$$2n+3 = 2n+3n$$

$$3 \leq 3n$$

$$n \geq 1$$

$$\rightarrow f(n) = O(n)$$

$$f(n) = O(n^2)$$

$$f(n) = 5$$

$$f(n) = 3n \rightarrow O(n)$$

$$f(n) = 1$$

$$f(n) = n + 5000$$

$$f(n) = n + 3$$

$$f(n) = 2n + 3$$

$$f(n) = \frac{n}{50} + 20$$

$$\text{Ex } 2n^2 = O(n^3) \text{ için } c, n_0 \text{ değerlerini bulunuz.}$$

$n_0 \geq 1$

$$f(n) \leq C \cdot g(n) \rightarrow 2n^2 \leq Cn^3$$

$$c=1 \text{ için } n_0=2 \text{ şartı sağlar.}$$

$$\text{Ex } \frac{1}{2}n^2 + 3n \text{ için istenilen } O(n^2) \text{ olduğunu gösteriniz.}$$

$n_0 \geq 1$

$$c=1 \text{ için:}$$

$$\frac{1}{2}n^2 + 3n \leq n^2$$

$$3n \leq \frac{1}{2}n^2$$

$$6 \leq n$$

$$n_0 = 6$$

özelik kumesini sağlayan her tane n_0 ve c çift olduğunu öneksi değildir, deki bir çift olması notasyonun doğruluğunu için yetelidir.

$$\text{Ex } 3n^2 + 2n + 5 = O(n^2) \text{ olduğunu gösteriniz.}$$

$n_0 \geq 1$

$$3n^2 + 2n + 5 = 3n^2 + 2n^2 + 5n^2$$

$$8n^2 + 2n + 5 = 10n^2$$

$$3n^2 + 2n + 5 \leq 10n^2$$

$$c=10 \quad n_0=1$$

* Bir Fonksiyonun Büyümesi (Growth)

Fonksiyonların büyümeye hızı matemатике Big O olarak ifade edilen fonksiyon gösterilir.

Tanımı: f ve g fonksiyonları reel sayılarından reel sayılarla tanımlı iki fonksiyon olsan.

$f(x)$ fonksiyonu için C ve k sabit olmak üzere $O(g(x))$ aşağıdaki şekilde şeklinde tanımlanır.

$$f(x) \leq C * g(x) + k$$

$x > k$ olmak üzere

$$f(n) \leq C * g(n) + n > n_0$$

Ex $f(x) = x^2 + 2x + 1$ fonksiyonun büyümeye fonksiyonun x^2 olduğunu yani $O(n^2)$ olduğunu gösteriniz.

$x > 1$ için (bu syu zamanda $x > k$):

$$x^2 + 2x + 1 \leq x^2 + 2x^2 + x^2$$

$$x^2 + 2x + 1 \leq 4x^2$$

$$(C=4)$$

$$(k=1)$$

$$\rightarrow$$

$$x^2 + 2x + 1 = 4x^2$$

$$x = k = 1$$

$f(x) \leq cx^2 + k$, $x > k$ durumunu sağlarsa,

$$\Rightarrow f(x) \text{ is } O(x^2)$$

" C " اما يقرئ "و" وهو n معنده n يدخل

* Definitions:

$$f(n) = O(g(n))$$

$$f(n) \leq n \cdot g(n) + n > n_0$$

* $f(n) \leq c \cdot g(n)$ for all $n > n_0$ * $f(x) \leq n \cdot g(x) + k \forall x > k$

$$\text{Ex } f(n) = 2n + 6$$

$$f(n) = O(n)?$$

$$f(n) = O(g(n))$$

$$g(n) = n$$

$$c = 4$$

$$2n + 6 \leq 4n$$

$$6 = 2n$$

$$n = 3 \rightarrow n > 1$$

$$c > 4$$

$$\text{Ex } f(n) = 2n + 3$$

$$f(n) \stackrel{?}{=} O(n)$$

$$2n + 3 \leq 10n$$

$$f(n) < g(n)$$

$$f(n) = 2n + 3$$

$$f(n) \stackrel{?}{=} O(n)$$

$$2n + 3 \leq 2n + 3n$$

$$3 \leq 3n$$

$$n > 1 \leftarrow$$

$$f(n) = O(n)$$

$$f(n) = O(n^2)$$

$$2n + 3 \leq 2n^2 + 3n^2$$

$$2n + 3 \leq 5n^2$$

Algoritma nedir? Bir problemi çözme için bir prosedür veya formül olur.

- problemi çözme için takip edilmesi gereken yürüteçler kümeleridir.

Program? Bir programda algoritmanın gerçekleştirilmesidir.

Algoritmaların Özellikleri

Input: Girdi, bir kümedir.

Output: Çıktı, bir kümedir.

Definiteness: kesinlik (algoritmanın adımlarının belirlili olması)

Correctness: Doğruluk (bütün girdiler için algoritma doğru olması)

Finiteness: sınırlılık (algoritma adımlarının sınırlı olması)

Effectiveness: Uygulanabilirlik (her adımdan ve sonucu ulusun yeter)

Generality: Genelleştirilebilirlik (bir problem içimdeki icin)

What Big O is not?

- * Big O is not going to give you an exact answer on how long a piece of code will take to run.

- * Does not consider the performance of hardware.

- * An algorithm's space complexity: specifies the total amount of space or memory required to execute an algorithm as a function of the size of the input

- * An algorithm's time complexity: specifies how long it will take to execute an algorithm as a function of the input size.

- * Things Affect program speed and Efficiency

Program Speed

Algorithm

Hardware (CPU, RAM, OS)

→ performance

$O(n^2)$

$O(n)$

$O(\log n)$

$O(1)$

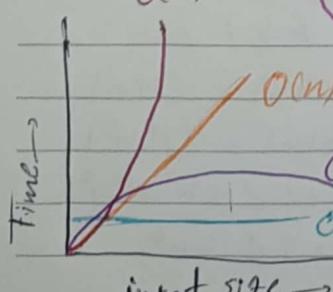
Time
Space

Complexity

→ Time vs Input
Space vs Input

$\rightarrow O(n)$

input size





Invoking installer

```
Running Intel® HAXM installer
```

```
Intel HAXM installation failed!
```

```
For more details, please check the installation log: C:\Users\lenovo\AppData\Local\Temp\haxm_install-20240407_2304.log
```

```
Intel® HAXM installation failed. To install Intel® HAXM follow the instructions found at: https://github.com/intel/haxm/wiki/Installation-Instructions-on-Windows
```

```
Done
```

I