

(edition 3rd) CLRS ← 2010

(harowitz) Fundamentals of Computer Algorithms ←

* الورقة ملحوظة من اسوان
* $n \log n$

: 0(n log n)
* معنى وصف الورقة

* دو وارط بازرسی
* دو وارط بازرسی
* دو وارط بازرسی
* دو وارط بازرسی
* دو وارط بازرسی

الخوارق ، ایجوری ، Quick sort ، Heap sort ، سایر الگوریتم ها *

counting Sort ← سایر الگوریتم ها *

radix Sort ←

Bucket Sort ←

selection الگوریتم - در $n \log n$ معنی و کاربرد پنجه *

Amortized Analysis - پنجه دارد *

(divided & conquer) دو، رئیس جوگ *

(recursion) (ریکورنس) (recursion)

dynamic programming لجستیک برنامه نویسی *

Largest common Subsequence

بلند ترین سکونت

(greedy) نیازمند جوگ *

کمترین حافظه

fractional جمله ای

✓
مکانیزم جوگ ←
(back tracking) ←
DFS

ردیابی جوگ ←

(branch & bound)

محدود کردن مقدارها ←
ردیابی جوگ ←

* الخوارزميات الخطية وال NON الخطية

BFS / DFS ← الخوارزميات -

Dijkstra, Floyd Warshall ← الخوارزميات -

Prim, Krushal ← الخوارزميات -

(maximum flow) الخوارزميات *

P الخوارزميات ←
NP الخوارزميات ←
NP-hard الخوارزميات ←
NP-complete الخوارزميات ←

• الخوارزميات

10 ← الخوارزميات , 3 ← الخوارزميات , 7 ← الخوارزميات

جیسم دوم - 1401/11/24

لورین: جیمه الورین ✓

اللورین \rightarrow جیله ای از سوران نه باشی مخصوصاً این اسید و مسیر را
کلید

سلیم \rightarrow ب نوع سریست و سلیم است از وروک مسلمه و
کلید کلید مسلمه

وروک: نسبت مدهونه سلیم است از
وروک: نسبت مدهونه سلیم است از

کلید

کلید: جیله ای از فن از عرق و روک نه

/

* ملای عیق از ملای (ملای) و لای اللورین خارج در ملای خود (ملای اسید)

بررسی اینکه کلید الورین خوب:

فقط لای نه باشد سلیم و بسازی است که عقار عدها، 10^6 است.

و نوع قامیور (لورین) نه بی ضعیت تراز ملای است.

$$10^6 \text{ ins/sec} = \frac{\text{نحوه}}{\text{ثانية}} \quad \left. \begin{array}{l} \text{نحوه} \\ \text{ثانية} \end{array} \right\} = A \text{ op}$$

$$10^7 \text{ ins/sec} = \frac{\text{نحوه}}{\text{ثانية}} \quad \left. \begin{array}{l} \text{نحوه} \\ \text{ثانية} \end{array} \right\} = B \text{ op}$$

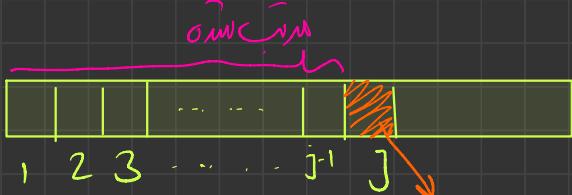
$$n \log n = \frac{\text{نحوه}}{\text{الوقت}} \times \text{نحوه}$$

$$A \text{ op} = \frac{(10^6)^2}{10^9} \approx 10^3 \text{ sec}$$

$$B \text{ op} = \frac{10^6 \log 10^6}{10^7} \approx 2 \text{ sec}$$

حقل زماني و اسماك دفعه الورقة ✓
insertion sort

incremental insertion - Insert incremental



Key = insertion key

این مرحله را باز از پیش نهاده کنیم که درست شد

Insertion-Sort(A, n) :

```
1   for(j=2 to n) do :  
2       key = A[j]  
3       i = j-1  
4       while( i>0 and key < A[i] ) :  
5           A[i+1] = A[i]  
6           i = i-1  
7       A[i+1] = key
```

الخطوة i+1 تسمى العنصر المدخل (Input Element)
الخطوات من 0 إلى i تسمى العناصر المدخلة (Input Elements)
Loop-invariant
والخطوة i+1 تسمى العنصر المدخل (Input Element)

أمثلة على (1, ..., j-1) و (j, ..., n) في insertion sort \rightarrow loop invariant

أمثلة على insertion sort \rightarrow for $i=0$ to $j-1$:
 $A[1, ..., j-1]$ هي الأجزاء التي تم ترتيبها حتى الآن،
و $A[1, ..., j-1, j, ..., n]$ هي الأجزاء التي لا تم ترتيبها بعد.

حالات پایه: یکی از سه

for $i = 1$ to n : در اینجا i میزد

لیست $A[1..n]$ باز نوشته شده: در اینجا i میزد

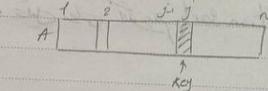
الgoritم insertion sort کردن $A[1..n]$ با این مرحله و مرور

در نظر آنکه $A[1..n]$ بوده اند و یعنی مورخ هر کدامیک

از پایه درست است

فایل سمعی و

Insertion sort | مرتبه کارهای ایجاد



ستون های متوالی $\leq t_j \leq j-1$ $T(n) \in \Theta(n)$

تفاوت تکرار جانبه $T(n) \in \Theta(n)$

$$E[X] = E[X] = \sum_k k \cdot P(X=k)$$

میانگین تصادفی

$$E[t_j] = \sum_{k=0}^{j-1} k \cdot P(t_j=k)$$

- میانگین تصادفی شاخص (Indicator random variable)

$$I\{A\} = \begin{cases} 1 & \text{اگر محدودیت برقرار است} \\ 0 & \text{دریز}\end{cases}$$

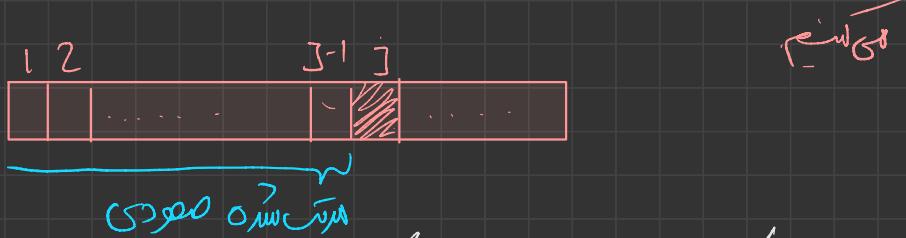
$$X_i = \begin{cases} 1 & \text{اگر محدودیت برقرار است} \\ 0 & \text{دریز}\end{cases} \Rightarrow t_j = \sum_{i=1}^{j-1} X_i$$

$$\text{PQPCO} \Rightarrow E[t_j] = \sum_{i=1}^{j-1} E[X_i], E[X_i] = P(\text{خطه کمینه } A[i]) = \frac{1}{i}$$

∴ $\text{few } \text{mild}$

• insertion sort

لائے نہار و فرضیہ کیستہ بڑا ی جائیں تاکہ بحر اسٹ و بڑا ی اج اپنے



مودع خانه‌ای ازین در باره ایجاد مسئله می‌شود

وَمُؤْمِنٌ بِرَبِّهِ وَالْمُحْسِنُونَ

کیمی و ارگانهای قبل از خلولیت مراد در آرایه همچو کتابی

کاری می‌شود و نهایت درجی از آرایه درجی سورک از تابعهای معرفی شد.

خود روشی دارایی‌های قابل از خود بزرگ درست.

سینت دنی مونکاگی این ۱-۲-۳ رئیس پرسکارا عرضی نمایند.

نایابی در این مسایل باید از این طبقه noise for $A[\Gamma_1, \dots, j]$ نظر داشته باشد.

فِي الْعَالَمِ اُولَئِكَ هُوَ الْمُرْسَلُونَ (١٠) إِنَّمَا يَنْهَا مَنْ يَرَى

in English by the Order of the

(For $\text{m}(\text{in})$) \rightarrow $\text{m}(\text{in})$ insertion sort \rightarrow $\text{m}(\text{out})$

ما يُطلب طلبه صيغة صيغة بعده از :

$\forall j \in [1, \dots, n]$ $\exists i \in [1, \dots, j]$ $A[i] < A[j]$ for $i < j$

عنصر أوله غير أول $A[1, \dots, n]$ اسفل عن $A[1, \dots, n]$

تحليل زمان الورق معجز

* The RAM model \rightarrow محاشرات (سوار ماسن سافل)

مع وظائف، ضد وظائف، مع مس، وظائف وبيانات، وعمليات

از مجموعات باي اى مفاهيم

تابع زمان الورق [ما هي؟]

بروبي \leftarrow insertion sort

$T(n) = n^2$ زمان الورق رام بـ n يعني تكرار امثلة بين n عدد

الورقة n عدد n يعني طابعه n زمان الورق رام بـ n تكرار سينات

طفل وورقة \leftarrow size

Insertion-Sort(A, n) :

1 for ($j=2$ to n) do : $\xrightarrow{j=2 \text{ or } j+1}$ $\xrightarrow{j < n}$ $\xrightarrow{n \cdot C_1}$

2 key = $A[i]$ $\xrightarrow{(n-1) \cdot C_2}$

3 $i = j-1 \rightarrow (n-1) \cdot C_3$

4 while ($i > 0$ and $key < A[i]$) : $\left(\sum_{j=2}^n (t_j + 1) \right) C_4$

5 $A[i+1] = A[i] \rightarrow (\sum_{j=2}^n t_j), C_5$

6 $i = i-1 \rightarrow (\sum_{j=2}^n t_j) C_6$

7 $A[i+1] = key \rightarrow (n-1) \cdot C_7$

الآن نحن في مرحلة انتهاء الدالة ونرجع الى الخطوة 1 من البداية
والمقدار t_j هو المقدار الذي انتهى من خوض دالة $t_j = j-1$, C_6

$T(n) = a \times n + b$: $\xrightarrow{\text{معمل المعرف}}$ $t_j = 0 : C_6 \cup C_5$

$t_j = j-1 : C_6 \cup C_5$

$T(n) = C_1 n + C_2(n-1) + C_3(n-1) + \sum_{j=2}^n (t_j + 1) \cdot C_4 + (C_5 + C_6) \left[\sum_{j=2}^n t_j + \right]$

$C_7(n-1)$

$\rightarrow T(n) = a'n^2 + b'n + c'$

يُعرف بـ divide and conquer و هو يعتمد على طريقة divide and conquer

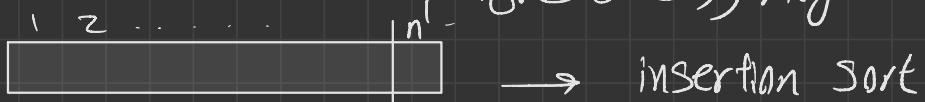
• Merge sort

يسعى الى divide and conquer

الآن نحن نعمد الى insertion sort و merge sort

لذلك ، insertion sort و merge sort هما الـ building blocks

لـ merge sort



لـ insertion sort $O(n^2)$ و لـ merge sort $O(n \log n)$

لـ divide and conquer $O(n \log n)$

الآن نحن في مرحلة الـ التجزئة
الآن نحن في مرحلة الـ التجزئة

merge-sort (A, i, j) :

1 if ($j > i$) :

2 $q = \lceil \frac{i+j}{2} \rceil$

3 merge-sort (A, i, q)

4 merge-sort ($A, q+1, j$)

5 merge (A, i, q, j)

• Chambers

divide and conquer strategy

۱- (۶) مسیر ← مسله (مکانیکی نیز بحث کنید) (مکانیکی نیز بحث کنید) مسیر ← اینجا ←

جواب ۲

۳ - ۶۹ /

الخطوة التالية هي تجميع (Merge) التي تجمع بين المảngين المصنفين A[i:j] و A[j:k] ل形成 المảng المصنف A[i:k].

$f(j > i)$:

$$q = \left\lceil \frac{i+j}{2} \right\rceil$$

merge-sort(A, i, q)

merge-Sort(A, q+l, j)

Merge(A, i, q, j)

- 1. ملکہ حسنہ
- 2. باریکی
- 3. ملکہ حسنہ
- 4. باریکی
- 5. ملکہ حسنہ

Merge(A, i, q, j):

1 $n1 = q - i + 1, n2 = j - q$

2 $L[n1+1] = \infty, R[n2+1] = \infty$

3 for ($l = 1$ to $n1$):

4 $|L[l] = A[i+l-1]$

5 for ($r = 1$ to $n2$):

6 $|R[r] = A[q+r]$

7 $l = r = 1$

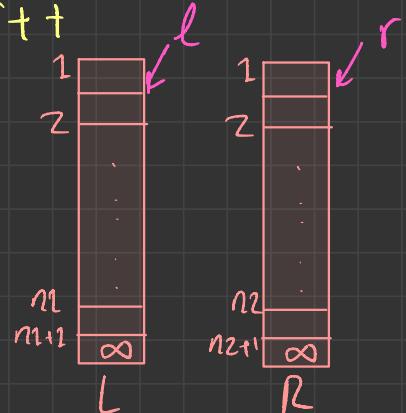
8 for ($k = i$ to j):

9 if ($|L[l] < R[r]$):

10 $|A[k] = L[l], l++$

11 else:

12 $|A[k] = R[r], r++$



ابدأ دوري للعرس \rightarrow Merge

فقط: آرایه ورودی A از i^{th} قسمت j^{th} سطر $q+1^{th}$ است، $q \leq i \leq n$

١

حاج، دریافت شد (شاید) ممکن نباشد

لابى ملأة : دراسات اجتماعية للعلوم وبرأى أفراد

$\approx \sqrt{1} \text{ m/s } (\text{just like } k-i \text{ do for } \alpha_i, \dots, k-1)$

Second one is the $\tilde{f}_{\text{opt}} \sim \text{Cat}(R[1, \dots, n+1], [1, \dots, n+1])$

A $\overset{\text{def}}{=}$ $\{R, L\}$ $\overset{\text{def}}{=}$ $\{R[r], L[l]\}$ $\overset{\text{def}}{=}$ $\{R[r], L[l]\}$

$\ell > r \leq l$ \leftarrow $(k \leq i)$: $\overline{b_{new}} \in \mathcal{N}_l$

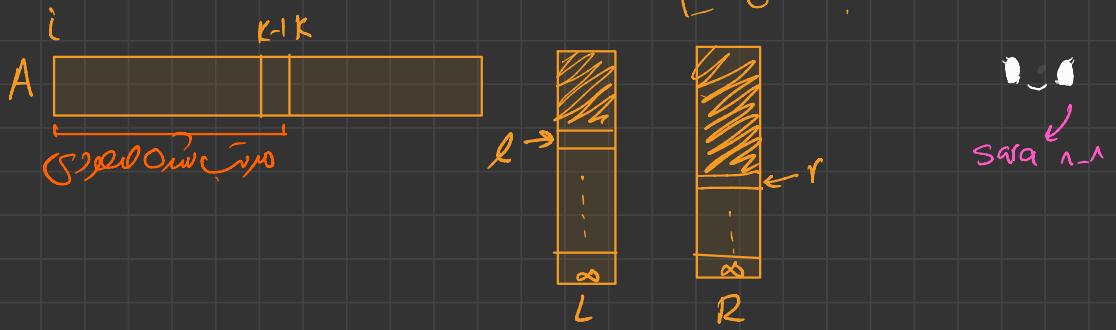
در اینجا ۱۰۸ حل for ΔK ^{لعل} ^{نیز} در آرک

الكلمات المقطعة هي $\{w_1, w_2, \dots, w_n\}$

Good one-to-one correspondence between $R[1, \dots, n+1]$ and $\{1, \dots, n+1\}$

A σ -semiregular R, L \sqcup \sqcap $\sqcup\sqcap$ $\{R\}$, $\{L\}$ $\{R\}$

• الْمُؤْمِنُونَ



لهم اشهد أنك أنت ربنا لا إله إلا أنت رب العالمين
أنت على كل شئ قدير

اپنے مرتبہ : merge-sort

پانی اسکے : $i = j$ $\leftarrow i = j$

اگر صاف پانی سطح ($i < j$) if بروز رسانی و از المعرفت خارج ہو سوں (و ازان) دوسری دست تحریک باقیہ ہمارا نہ چل سکتا ہے عذر طرد۔ سو سے سویں

اس سے پانی سطح اسے

فہری اسکے مرتبہ merge-sort : کہاں کہاں دست میں
مکالمہ

اے

: merge-sort , merge слож

Merge(A, i, q, j):

1 $n_1 = q - i + 1$, $n_2 = j - q \rightarrow C_1$ число
2 $L[n_1 + 1] = \infty$, $R[n_2 + 1] = \infty \rightarrow C_2$ число
3 for ($l = 1$ to n_1) :
4 $| L[l] = A[i + l - 1] \rightarrow C_3 n_1$
5 for ($r = 1$ to n_2) :
6 $| R[r] = A[q + r] \rightarrow C_4 n_2$
7 $l = r = 1 \rightarrow C_5$
8 for ($k = i$ to j) :
9 if ($L[l] < R[r]$):
10 $| A[k] = L[l], l++ \rightarrow C_6 \cdot n$
11 else:
12 $| A[k] = R[r], r++$

الخطوة
merge

$$\leq C_1 + C_2 + \max\{C_3, C_4\}(n_1 + n_2) + C_6n$$

$$\Rightarrow \text{الخطوة} \leq a.n + b \rightarrow \Theta(n)$$

لذلك $\Theta(n)$ هو الحد الأقصى للوقت.

جامعة الملك عبد الله بن عبد العزیز

(1401, 12, 8)

Subject: _____
Year: ١٤٠١ Month: ١٢ Date: ٨

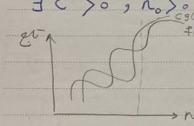
طاهر البرقى
جامعة الملك عبد الله بن عبد العزیز

Merge-Sort (A, i, j)
1. if $i < j$ then
2. $\ell = \lfloor \frac{i+j}{2} \rfloor$
3. Merge-Sort (A, i, ℓ)
4. Merge-Sort ($A, \ell+1, j$)
5. Merge (A, i, ℓ, j)
 $\Theta(n)$

\Rightarrow دالة تكامل: $T(n) = C_1 + C_2 + T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T\left(\left\lceil \frac{n}{2} \right\rceil\right) + \Theta(n)$
 $T(n) = T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T\left(\left\lceil \frac{n}{2} \right\rceil\right) + \Theta(n)$ دالة تكامل
 $T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$

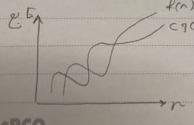
مقدار دالة تكامل: $T(n) = \Theta(n)$
 $T(n) \in \Omega(n)$ $T(n) \in \Theta(n)$
 $T(n) \leq O(n)$ $T(n) \in \Theta(n)$

$\exists c > 0, n_0 > 0 \mid \forall n > n_0 \quad f(n) < c g(n) \iff f(n) \in O(g(n))$

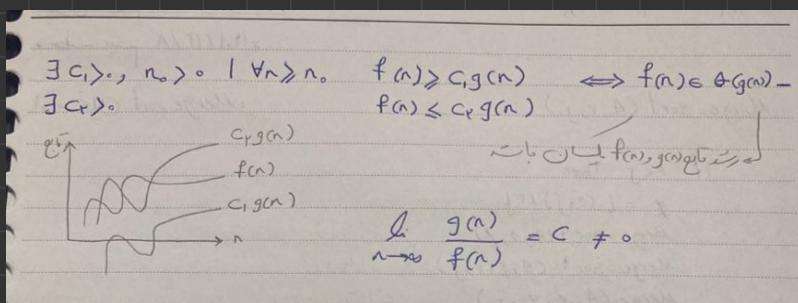


$\lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = \infty$ $c \neq 0$

$\exists c > 0, n_0 > 0 \mid \forall n > n_0 \quad f(n) > c g(n) \iff f(n) \in \Omega(g(n))$



$\lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = 0$ $c \neq 0$



١٤٥ / ١٤٦ / ١٤٧ : مراجعة

أداة مراجعة

Little O:

$$f(n) \in O(g(n)) \iff \forall c > 0, \exists n_0 > 0 \mid \forall n > n_0, f(n) \leq c \cdot g(n)$$

f(n) ≤ c · g(n)

$$\lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = \infty$$

if $f(n) \leq 5n^2 + n \in \boxed{O(n^2)}$ $\Rightarrow f(n) \in O(n^3)$

f(n) ≤ n^3

$$\in \boxed{\Omega(n)} \Rightarrow f(n) \in \Omega(\log n)$$

Ω(n)

Little Omega:

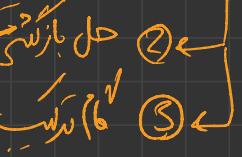
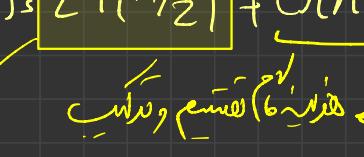
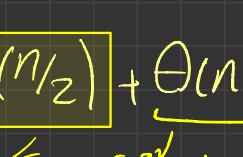
$$f(n) \in \omega(g(n)) \iff \forall c > 0, \exists n_0 > 0 \mid \forall n > n_0, f(n) \geq c \cdot g(n)$$

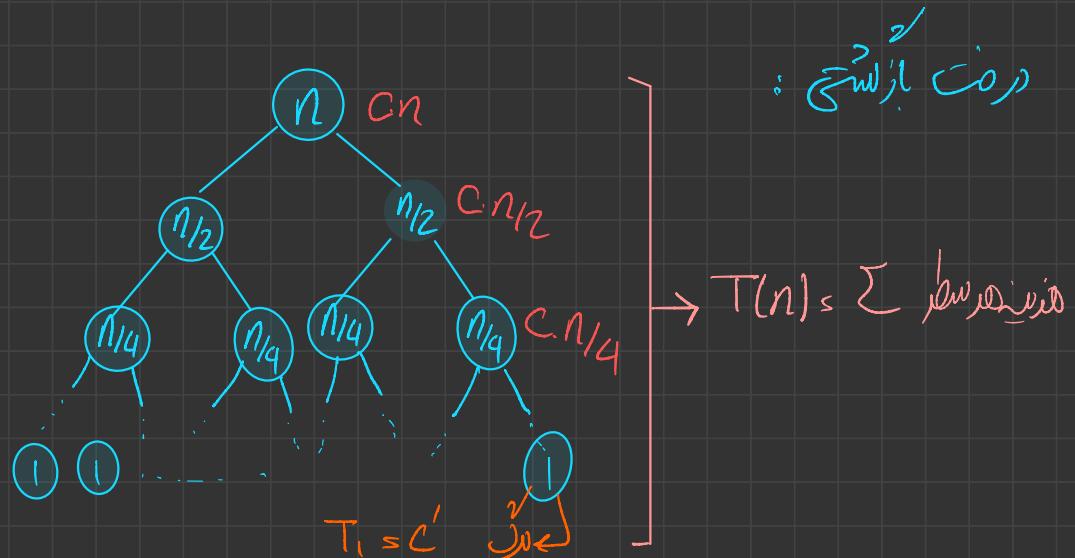
* $f(n) \in \Theta(f(n) + o(f(n))) \quad \triangleq \quad \heartsuit$

$f(n) = n^{1+\sin n}$
 $g(n) = n$

 ← حل بازگشتی فرمون توانی خواهد بود
 
 : حل بازگشتی بولجی دارد
 

 (Recursion tree)  
 (The Master theorem) 

$2T(n/2)$  حل بازگشتی
 ①  حل بازگشتی
 ②  حل بازگشتی
 ③  حل بازگشتی
 
 : حل بازگشتی
 
 : حل بازگشتی
 
 $T(n) \leq 2T(n/2) + \Theta(n)$

هذا	هذا	الآن
$C \cdot n$	1	$R/2^k = 1 \Rightarrow k = \log_2 n$
$Cn = C \cdot n/2 + C \cdot n/2$	2	
$Cn < 4 \times Cn/4$	3	$\text{لذلك } i = \log_2 k + 1 \text{ يساوى } \log_2 n + 1$
\vdots	\vdots	
Cn		\log_2^n

الآن $T(n) = \log_2^n \times Cn = C \cdot n \cdot \log_2^n$.

لذلك $n \times C' = C'n$

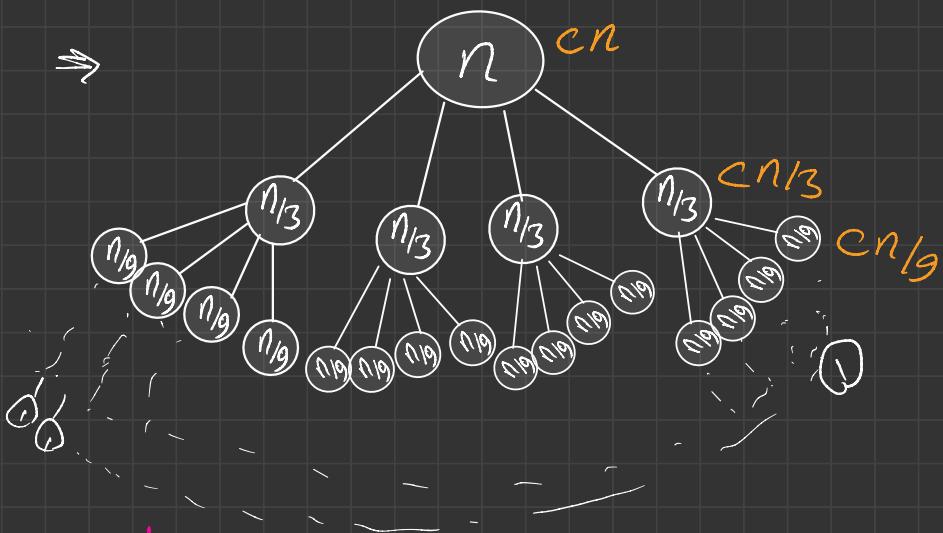
$$\begin{aligned} \text{نقطة الماء} &= 2^{i-1} \\ \text{نقطة الماء} &\leq 2^{((\log_2^n + 1) - 1)} = n \end{aligned}$$

$T(n) = Cn \log_2^n + Cn \in \Theta(n \log n) \Leftarrow \text{الآن}$

$\hookrightarrow \text{Merge Sort} \quad \text{هذا الورقة}$

$$T(n) = 4T\left(\frac{n}{3}\right) + \Theta(n), \quad T(1) = C$$

Call tree \Rightarrow



Height | Level

C_n

1

$$\underbrace{C_n}_{\text{Root}} = n/3^{i-1}$$

$4C_{n/3}$

2

$16C_{n/9}$

3

$$\text{Work done} \stackrel{>}{\rightarrow} 1 \leq \frac{n}{3^{i-1}} \Rightarrow i = \log_3^n + 1$$

$$\text{Cost of work} = \sum_{i=1}^{\log_3^n} C \cdot n \cdot \left(\frac{4}{3}\right)^{i-1} =$$

$$\log_3^n = C \cdot n \cdot \frac{\left(\frac{4}{3}\right)^{\log_3^n} - 1}{\frac{4}{3} - 1} = 3C \cdot n \left(n^{\log_3^{\frac{4}{3}} - 1}\right)$$

1

$$a^{\log_b c} = b^{\log_c a} \quad \text{and} \quad \Rightarrow \text{Cost of work} \in (n \cdot n^{\log_3^{\frac{4}{3}}})$$

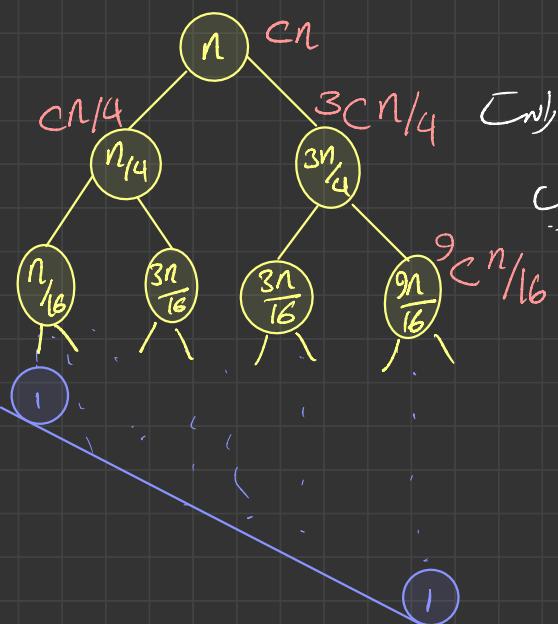
$$\text{زیر مجموعه های کوچک} = 4^{i-1} \Rightarrow \text{زیر مجموعه} = 4^{(\log_3 n + 1) - 1} = 4^{\log_3 n}$$

$$\text{زیر مجموعه های کوچک} = 4^{i-1} \times 4^{\log_3 n} = C' n^{\log_3 4}$$

(1401 12, 15) دلیل همچنین

$$T(n) \geq T\left(\frac{n}{4}\right) + T\left(\frac{3n}{4}\right) + \Theta(n)$$

: از



* زیر مجموعه ای از تابع $T(n) = cn + cn + cn = 3cn$

که در تابع $T(n) = cn + T(n/4) + T(3n/4)$ داشته باشد

دارد.

نام	جواب
cn	1
$cn/4 + 3n/4 = cn$	2
cn	3

لذا $T(n) \geq cn$ نیز بدلیل این است که $T(n) \geq cn + cn + cn = 3cn$

نحوی ایجاد شده است که n/a^{i-1} است

$$\text{تعداد مراحل} \leq 1 \rightarrow i = n/a^{i-1} \rightarrow \boxed{\log_4 n + 1 = i} \quad \leftarrow \text{تعداد مراحل ایجاد شده است}$$

کامپیوشن دارای این تابع است: در اینجا فرمول:

مرحله	تعداد مراحل
$c_1 n$	1
$c_2 n$	2
\vdots	\vdots
c_n	\log_4^n

$$\text{تعداد مراحل} \leq \log_4^n \times c_1 n = c_1 n \log_4^n$$

$$\text{تعداد مراحل} \leq c' \sqrt{n} \quad \begin{array}{l} \text{تعداد مراحل ایجاد شده است} \\ \text{با اینکه} \end{array} \quad \leftarrow \text{تعداد مراحل ایجاد شده است} \quad \log_4^n$$

$$\text{تعداد مراحل} \leq 2^{(\log_4^n + 1) - 1} = 2^{\log_4^n} = n^{\log_4 2} = \underline{\sqrt{n}} \quad \checkmark$$

$$T(n) \leq c_1 n \cdot \log_4^n + c' \sqrt{n} \in \Theta(n \log n)$$

کامپیوشن دارای این تابع است: در اینجا فرمول ایجاد شده است

تعداد مراحل ایجاد شده است

$$\text{سازمانی سیستم} = \left(\frac{3}{4}\right)^{i-1} n$$

سازمانی سیستم $\rightarrow i \leq \left(\frac{3}{4}\right)^{i-1} n \rightarrow i \leq \log_{\frac{4}{3}} n + 1$

$$\text{وقت زمانی} = C \cdot n \cdot \log_{\frac{3}{4}} n$$

$$\text{وقت زمانی} = C' \cdot n^{\log_{\frac{4}{3}} 2}$$

$$T(n) \text{ عالی} \rightarrow Cn \log_{\frac{4}{3}} n + C'n^{\log_{\frac{4}{3}} 2} \in \Theta(n^{\log_{\frac{4}{3}} 2})$$

$$\Rightarrow T(n) \in \Omega(n \log n)$$

$$T(n) \in O(n^{\log_{\frac{4}{3}} n})$$

$$T(n) \in \Theta(n \log n) \quad \begin{matrix} / \\ \text{برای} \\ \text{حالات} \end{matrix}$$

$$\Leftarrow T(n) \in O(n \log n) \quad \begin{matrix} / \\ \text{برای} \\ \text{حالات} \end{matrix}$$

$$\exists C > 0, n_0 > 0 \mid \forall n > n_0, T(n) \leq C \cdot n \log n$$

از همکاری شما بزرگتر نگاه
نامن کلام ملسا همکاری

روش اثبات برگشت

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

$$T(1) = c' \quad T(2) = c''$$

$$\therefore T(n) \in O(n \lg n)$$

$\exists c, c', c'', n_0, n_0$

$n=2$

$$T(2) \leq c \cdot 2 \cdot 1 \cdot 2$$

$$\approx \sqrt{2}c$$

$$\rightarrow \boxed{c \cdot \frac{T(2)}{2}}$$

$$\rightarrow \approx \sqrt{2}c$$

$$(n=2)$$

$$T(n) \leq c \cdot n \lg n$$

از این استوار است

$$T\left(\frac{n}{4}\right) \leq c \cdot \frac{n}{4} \lg \frac{n}{4}$$

$$T\left(\frac{3n}{4}\right) \leq c \cdot \frac{3n}{4} \lg \frac{3n}{4}$$

$$-2c + \frac{3}{4}c \lg^3 2 + c'' \leq 0 \rightarrow$$

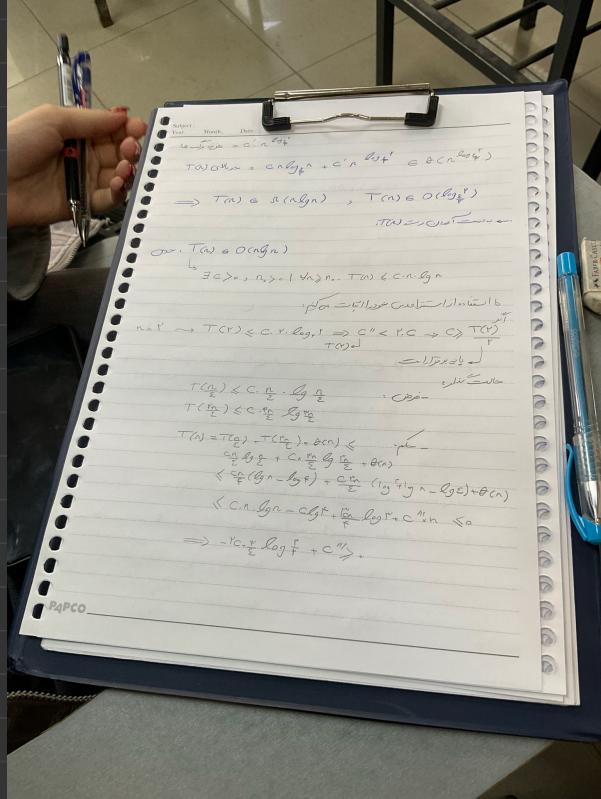
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{3n}{4}\right) + \Theta(n)$$

$$\leq \frac{c}{4}n \lg \frac{n}{4} + c \cdot \frac{3n}{4} \lg \frac{3n}{4} + \Theta(n)$$

$$\leq c \frac{n}{4} (\lg^n - \lg^4) + c \frac{3n}{4} (\lg^3 + \lg^n - \lg^4) + \Theta(n)$$

$$\leq c n \lg n - c n \lg^4 + c \frac{3n}{4} \lg^3 + c'' \cdot n$$

$$\leq c n \lg n$$



: (The master theorem) وَهُوَ

مُعْلِّمٌ مُفْتَحٌ از عَوْرَةِ بَيْنِ

$$(b > 1) \quad T(n) \leq cT\left(\frac{n}{b}\right) + f(n) \quad \leftarrow$$

$f(n) \in n^{\log_b^a}$ مُعْلِّمٌ مُفْتَحٌ

$$f(n) \in \Theta(n^{\log_b^a - \varepsilon}) \quad \text{اَنْ} \quad \text{لَمْ} \quad \text{يُكُونْ} \quad \varepsilon > 0 \quad \text{مُعْلِّمٌ مُفْتَحٌ}$$
$$T(n) \in \Theta(n^{\log_b^a}) \quad \text{اَنْ}$$

$$T(n) \in \Theta(n^{\log_b^a} \cdot \log n) \quad \text{اَنْ} \quad \downarrow \quad f(n) \in \Theta(n^{\log_b^a}) \quad \text{اَنْ}$$

$$\exists \varepsilon > 0 \quad \text{وَ} \quad c < 1, \quad f(n) \in \Omega(n^{\log_b^a + \varepsilon}) \quad \exists \varepsilon > 0 \quad \text{وَ}$$

$$T(n) \in \Theta(f(n)) \quad \text{اَنْ} \quad af\left(\frac{n}{b}\right) \leq c \cdot f(n)$$

(1401, 12, 80) \Rightarrow الحل

$$T(n) \leq 4 \cdot T\left(\frac{n}{2}\right) + \Theta(n)$$

مقدار : 1 جملة

$$f(n) \in \Theta(n) \quad a=4 \quad b=2$$

$$f(n) \in \Theta(n) \quad ? \quad n^{\log_b^a} = n^{\log_2^4} = n^2$$

$$f(n) \in \Theta(n^1) \in \Theta(n^{\log_2^4 - \varepsilon})$$

\leftarrow الخطوات

$$\in \Theta(n^{2-\varepsilon}) \Rightarrow 2-\varepsilon > 1 \Rightarrow \boxed{\varepsilon < 1}$$

نحوه دارد

$$T(n) \leq \Theta(n^{\log_2^4}) \leq \Theta(n^2) \quad : \quad \text{نحوه الخطوات}$$

$$T(n) \geq 2T\left(\frac{n}{2}\right) + n^2 \quad : \quad 2 \text{ جملة}$$

$$f(n) \geq n^2 \quad ? \quad n^{\log_2^2} = n^1$$

$$f(n) \geq n^2 \in \Omega(n^{\log_2^2 + \varepsilon}), \quad \leftarrow \text{الخطوات}$$

$$\in \Omega(n^{1+\varepsilon})$$

$$\rightarrow 1+\varepsilon < 2 \rightarrow \boxed{\varepsilon < 1}$$

$$af\left(\frac{n}{b}\right) \rightarrow 2f\left(\frac{n}{2}\right) = 2\left(\frac{n}{2}\right)^2 = \frac{n^2}{2} \stackrel{?}{\in} C.f(n), cn^2$$

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

$$\Rightarrow T(n) \in \Theta(f(n)) \rightarrow T(n) \in \Theta(n^2)$$

$$T(n) \leq 3T\left(\frac{n}{3}\right) + \Theta(n) : 3\downarrow n$$

$$f(n) \in \Theta(n) ? n^{\log_b^a} = n^{\log_3^3}, n^1$$

$$f(n) \in \Theta(n) \in \Theta(n^{\log_b^a}) \in \Theta(n^{\log_3^3}) \in \Theta(n') \leftarrow \text{Case 3}$$

$$T(n) \in \Theta(n^{\log_b^a} \cdot \log n) = \Theta(n^1 \cdot \log n) = \Theta(n \log n)$$

$$T(n) \leq 2T\left(\frac{n}{2}\right) + n \log n : 4\downarrow n$$

$$a=2, b=2, f(n), n \log n$$

$$f(n), n \log n ? n^{\log_b^a}, n^{\log_2^2}, n^1$$

$$f(n), n \log n \in \Omega(n^{\log_b^a + \varepsilon}) \in \Omega(n^{\log_2^2 + \varepsilon}) \in \Omega(n^{1+\varepsilon}) \leftarrow \text{Case 2}$$

$$\exists \varepsilon > 0 \rightarrow \log n \in \Omega(n^\varepsilon) \xrightarrow{\text{根据 } \varepsilon, \text{ 有 } \log n \rightarrow \infty}$$

! 2N! 2N! 2N! 2N! 2N! 2N! 2N! 2N! 2N! 2N!

لهم ياخذكم في حالت فتح وفتح

$f(n) \in n^{\log_b^a}$: دار :

$f(n) \in \Theta(n^{\log_b^a - \varepsilon})$ $\exists \varepsilon > 0$ كل $n > n_0$

$T(n) \in \Theta(n^{\log_b^a})$ دار

$T(n) \in \Theta(n^{\log_b^a} \cdot (\log n)^k)$ دار

$\exists c < C < 1$, $f(n) \in \Omega(n^{\log_b^a + \varepsilon})$ $\exists \varepsilon > 0$ كل $n > n_0$

$T(n) \in \Theta(f(n))$ دار $af\left(\frac{n}{b}\right) \leq C \cdot f(n)$

$T(n) = 2T\left(\frac{n}{2}\right) + n \log n$: كل $n \geq 1$

$f(n) = n \log n$? $n^{\log_b^a} \cdot (\log n)^k$

$f(n) = n \log n \in \Theta(n^{\log_b^a} \cdot (\log n)^1) \rightarrow k=1$

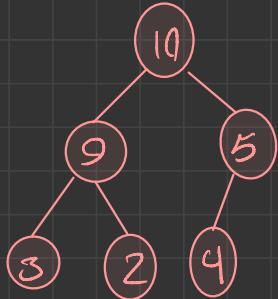
كل $n \geq 1$ دار

$T(n) \in \Theta(n^{\log_b^a} \cdot (\log n)^{k+1}) \in \Theta(n^{\log_b^2} \cdot (\log n)^{1+1}) = \Theta(n \log n)^2$

Quick sort *

堆排序 * : تحليل الالгорتم ها في مختصر سلسلة

الدورة الأولى من الالغورتم هي الـ Quick sort ، صيغة الـ Quick sort هي الترتيب من الأكبر إلى الأصغر

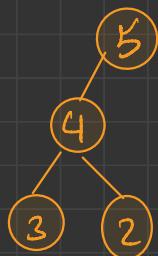
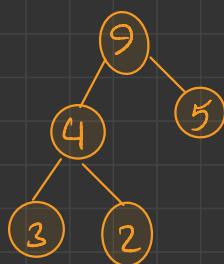
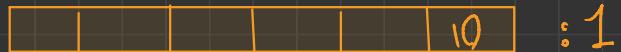
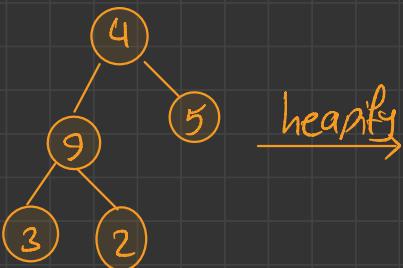


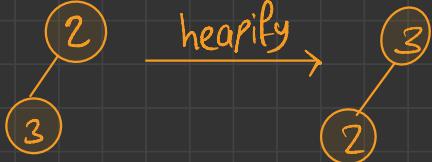
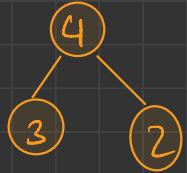
الدورة الثانية

الدورة الثانية من الـ Quick sort هي الـ heap sort

الدورة الثالثة

الدورة الثالثة من الـ Quick sort هي الـ heapify





(2)

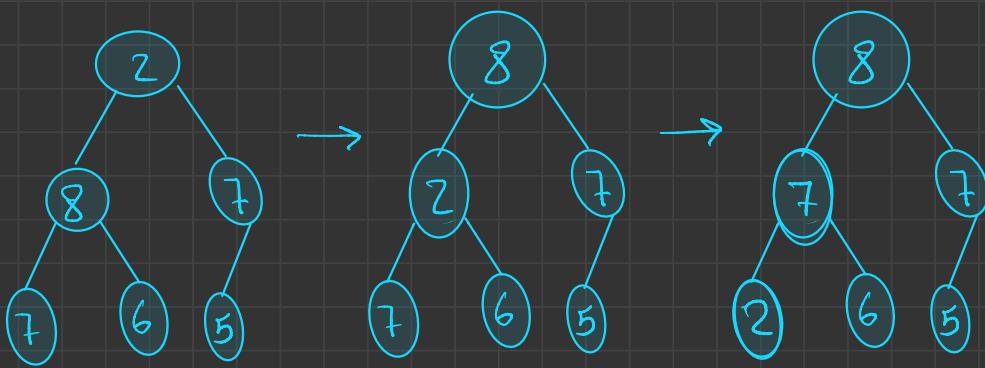


Heap_Sort(A, n):

```

1 Build_Max_Heap( $A, n$ )
2 for ( $i = n$  to 2) do:
3     Swap( $A[1], A[i]$ )
4     heap_size --
5     Max_Heapify( $A, i$ )

```

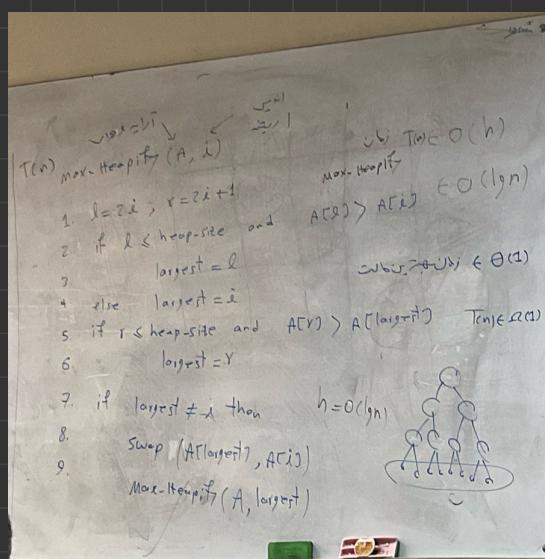
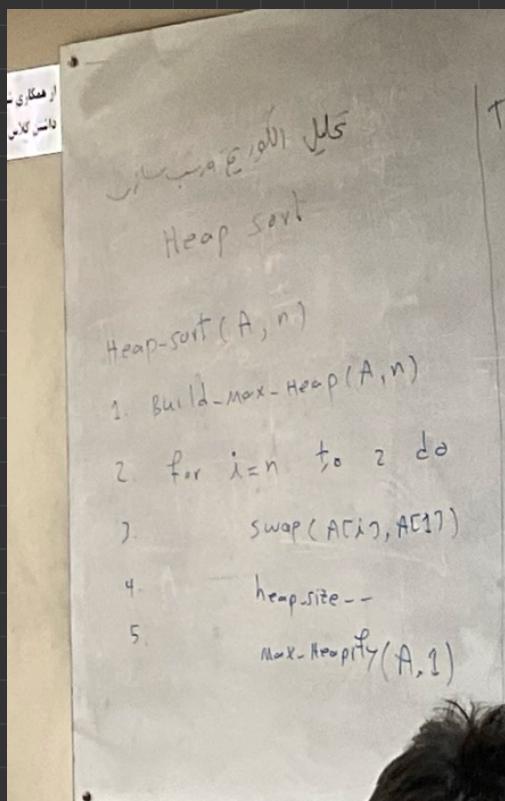


heap اولیه : heap-size = 6

نیزی اولیه : Length = 7

8	7	7	2	6	5	1
---	---	---	---	---	---	---

(١٤٠١، ١٢، ٢٢) : *permild*



Build-Max-Heap(A, n)

1. heap-size = n
2. for $i = \lfloor \frac{n}{2} \rfloor$ to 1 do
 - Max-Heapify(A, i)

Max-Heapify $\in \Theta(1)$

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

$a=1, b=2, f(n)=C$

$f(n)=C ? \frac{n^{\frac{a}{b}}}{n^{\frac{1}{2}}} = \frac{n^{\frac{1}{2}}}{n^{\frac{1}{2}}} = n^0 = 1$

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

Subject : _____
 Year : _____ Month : _____ Date : _____

- 5. if $r \leq \text{heap_size}$ and $A[r] > A[\text{largest}]$
- 6. $\text{largest} = r$
- 7. if $\text{largest} \neq i$ then
- 8. max-heapify ($A, \text{largest}$)
- 9. swap ($A[i]$, $A[\text{largest}]$)

Build - Max - Heap (A, n)

- 1. $\text{heap_size} = n$
- 2. for $i = \lfloor \frac{n}{2} \rfloor$ to 1 do
- 3. max-heapify (A, i)

$T(n) \in O(n) \in O(\lg n)$, Max-Heapify

$$T(n) = C + T\left(\frac{n}{3}\right) \rightarrow \text{بعضی ماتحتانه اینجا بحث برداشته شد}$$

$$a=1, b=3, f(n)=C$$

$$f(n) = C ? n^{\log_b a} = n^{\log_3 1} = n^0 = 1 \quad \text{حالات بد نظر نمایم}$$

$$T(n) \in O(\lg n)$$

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

$$m = \frac{n-2}{3}$$

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

$$\text{as } 1, b = \frac{3}{2}, f(n) = C$$

$$f(n) = C ? n^{\log_{3/2} 1} = n^0 = 1$$

$$\text{old method} \rightarrow T(n) \in \Theta(\lg n)$$

Build-Max-Heap ترتيب الهرم

build-max-heap(A, n)

1. heap-size = n

2. for $i = \lfloor \frac{n}{2} \rfloor$ to 1 do

3. max-heapify(A, i)

$$\frac{n}{2} \times O(\lg n) = O(n \lg n)$$

Build-max-heap $\in O(n \lg n)$

$\in O(n)$

max-heapify ترتيب الهرم
ن ارجع ويفسح مساحة ويفيد
و ارجع ويفسح مساحة ويفيد

ـ $O(n \lg n)$ ترتيب الهرم

Heap sort

Heap-sort(A, n)

1. Build-Max-Heap(A, n) $\rightarrow \Theta(n)$

$\text{C} \log n$

2. for $i=n$ to 2 do $\rightarrow \Theta(n)$

Swap($A[i], A[1]$)

heap-size--

Max-Heapify(A, i)

Time $\in O(n \log n)$

$\text{C} \log n$

$\text{C} \log n \in O(n \log n)$

$\text{C} \log n$

$\text{C} \log n$

$\Theta(n \log n) \leftarrow \text{Build Max-Heap}$

$\text{C} \log n$

وقت
آخر



الحالات الممكنة لـ " $i \leftarrow i + 1$ " : loop invariant

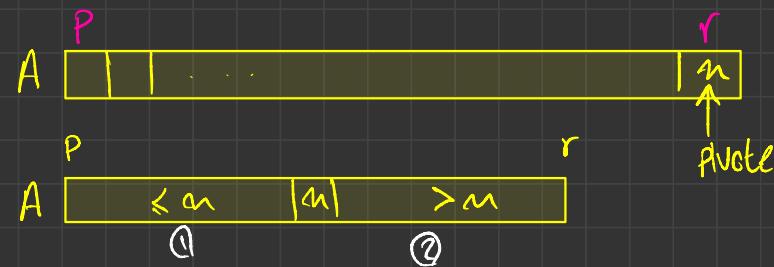
الحالات الممكنة لـ " $i \leftarrow n$ " : $i \leftarrow 1$

Max-Heap

جلسه (١٤) : (١٤٠٢، ١، ١٤)

تحليل الورقة Quick Sort

اسس الورقة divided & conquer و Merge Sort في Quick Sort *



الخطوة الثانية Partition (ج): ١٦

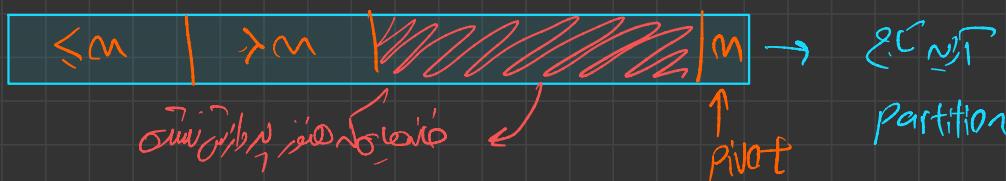
٢-١. عراحتنا هي حل معياري ، حل معياري
٣-٤. $E_k \leftarrow S_6$

```
Quick_sort (A,p,r):  
    if(p<r) : → ٤  
        q=partition(A,p,r) → Θ(n)  
        T(n) ← Quick_sort(A,p,q-1) ] → ٥  
        T(n) ← Quick_sort(A,q+1,r) ] → ٦
```

$$n_1 = q-p$$

$$n_2 = n - n_1 - 1$$

٥-٦. ابر و سلسلة ابر اكبر



Partition(A, p, r):

x = A[r]
i = j = p

for j = p to r-1 :

if (A[j] <= x):
 swap(A[i], A[j])
 i++

swap(A[r], A[i])
return i

: partition \rightarrow يُعيد ترتيب

إذا (x) if \leq A[j] \rightarrow أكبر بـ x \rightarrow يمكن تبديل

أكبر بـ x \rightarrow يمكن تبديل \rightarrow يمكن تبديل

C₂(n-1)

إلا (x) \rightarrow يمكن تبديل \rightarrow يمكن تبديل

C₃ \rightarrow يمكن تبديل \rightarrow يمكن تبديل

T(n) = C₁ + C₂(n-1) + C₃ $\in \Theta(n)$

End of proof \rightarrow T(n) $\in \Omega(n)$

الآن \rightarrow نريد إثبات \rightarrow نريد إثبات \rightarrow نريد إثبات

T(n) = C₁ + C₂'(n-1) + C₃ $\in \Theta(n)$ (C₂' > C₂)

\rightarrow T(n) $\in \Theta(n)$

Quick sort \rightarrow الوقت \rightarrow الوقت \rightarrow الوقت

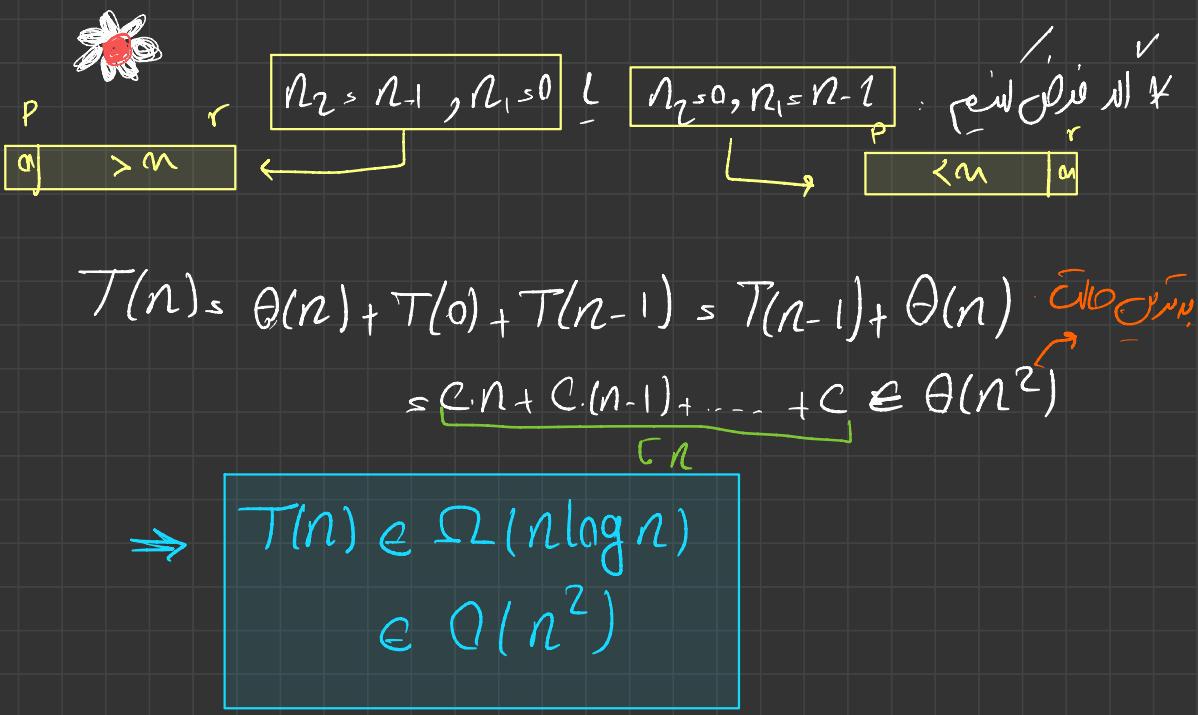
Quick sort \rightarrow الوقت

T(n) = C₁ + Θ(n) + T(n₁) + T(n₂)

T(n) = 2T(n/2) + Θ(n) $\in \Theta(n \log n)$

End of proof

n₁ = n₂ = n/2 \rightarrow الوقت \times الوقت \rightarrow merge sort \rightarrow الوقت



اسکن درسی الوریخ
 partition
 در خط ابتداء دسته
 سامان طبقه ← راسایی (اصلی) ←
 زیرگروه $A[i, r-1]$ نشان کن ، $>n$ مقدار $A[i, r-1]$
 \leftarrow $i \leq j \leq r$ ، $n = A[r],$ و هر چیزی

- این قسم نمودارها را با نام $i=j=p$ نیز می‌نامند
 $\leftarrow i=j=p : n \leq m$

حال نوار: $m \geq f(A[j]) \geq f(A[i])$ که می‌تواند مجموع دو گروه است
 $f(A[j]) \leq m$ باید شرطی داشت

(1402, 1, 19) : ملحوظات

ارقام تحليل Quick sort

$$T(n) = T(k) + T(n-k-1) + \Theta(n)$$

الآن $k < n$ و $n-k-1 < n$

نفترض $n-k-1 < k$ (مثلاً $n=10, k=5$)

الآن $T(n) = T(k) + T(n-k-1) + \Theta(n)$

نفترض $n-k-1 < k$ (مثلاً $n=10, k=5$)

الآن $T(n) = T(k) + T(n-k-1) + \Theta(n)$

(Randomized) : Quick sort مع التعديل

ـ اختيارpivot عشوائياً

```
Randomized_partition(A, p, r):  
    i = RANDOM(p, r)  
    swap(A[i], A[r])  
    return partition(A, p, r)
```

```
randomized_Quick_sort(A, p, r):  
    if(p < r):  
        q = Raandomized_partition(A, p, r)  
        randomized_Quick_sort(A, p, q-1)  
        randomized_Quick_sort(A, q+1, r)
```

ـ Randomized Quick Sort صحيح

$T(n) \leq O(n + X)$

ـ عدد مرات خروج

ـ عدد مرات دخول

$E(T(n)) \leq O(n + E(X))$

ـ متوسط

$z_1, z_2, z_3, \dots, z_n$ \leftarrow لیست اعداد اول \leftarrow لیست اعداد اول \leftarrow لیست اعداد اول

$Z_{i:j} = \{Z_i, Z_{i+1}, \dots, Z_j\} \rightarrow Z_i$ مجموعه اعداد میانگین

$$X = \sum_{i=1}^{n-1} \sum_{j=i+1}^n X_{i,j}$$

$$E(X) \leq \sum_{i=1}^{n-1} \sum_{j=i+1}^n E(X_{j,j}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^n \overbrace{P(Z_{ij} \geq 0)}^{\text{Probability}} Z_{ij}$$

$$k=j-i \rightarrow E[X] = \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} \leq \sum_{i=1}^{n-1} \left\{ \sum_{k=1}^n \frac{2}{k} \right\} \leq \sum_{i=1}^{n-1} O(n) \leq O(n \log n)$$

$\hookrightarrow \left\{ \frac{dm}{m} \text{ sum of } O(1) \right\} \leq \ln n$

وَهُوَ مَعَنِي، دَارِي إِنْ هُوَ مَدِينَةُ اسْتَنْدَانَ نَبَّهَ ازَانَ إِنْ تَرَسِيَ الْمَلَكَ،

نَسْبَتْ إِلَيْهِ

* دَعْيَا بِهِ هَرَبَّوْنَ نَعْلَمُ أَنَّهُمْ لَمْ يَأْتُوا بِالْحُجَّةِ فَلَمَّا رَأَوْنَا مُهَاجِرَاتِهِنَّا

$$\log(n!) \in \Theta(n \log n)$$

* الخوارزميات (Algorithms) البيانات (Data Structures)

٦٧

الطباطبائي

Counting Sort algorithm *

الله ربكم جميع درجات و مراتبكم جائزة

بـ الـ نـ وـ كـ سـ اـ حـ

اگر α ایک ممکنہ مقدار کے لئے $\beta = \sqrt{1 - \alpha^2}$ تو

نیازی می باشد که در اینجا تایپ (اچ اس) نموده و در پایان آن را باز کرده و متن مورد نظر را برای انتقال به سایر فایل ها استفاده کنید.

• १०५

* الورق Counting - Sort

Counting_sort(A,n,k):

$\Theta(k) \leftarrow \begin{cases} \text{for } (i=0 \text{ to } k) : \\ \quad C[i] = 0 \end{cases} \rightarrow$ الخطوة الأولى (أي الخطوة الأولى) لتحقيق

$\Theta(n) \leftarrow \left[\begin{array}{l} \text{for } (i=1 \text{ to } n): \\ \quad C[A[i]]++ \end{array} \right]$

```
θ(k) ← [for (i=1 to K):  
          C[i] = C[i]+C[i-1]
```

$$\Theta(n) \leftarrow \begin{cases} \text{for } (i=n \text{ down to } 1): \\ \quad B[C[A[i]]] = A[i] \\ \quad C[A[i]--] \end{cases}$$

1

۷

stable المُعْدِلُ

stable المُعْرِفِي

4 51 12 13

B 亂世の

$$\underline{m} = \Theta(n+k)$$

Counting Sort \rightarrow $\Theta(n+k)$

$$\Theta(n+k) = \Theta(n)$$

$\Theta(n^2)$ \rightarrow $\Theta(n+k)$

Radix Sort \rightarrow $\Theta(n+k)$

: Radix sort \rightarrow $\Theta(n+k)$

K'th digit \rightarrow n^{th} digit

\downarrow \downarrow \downarrow

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

\rightarrow \downarrow
 \downarrow

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

: Radix sort \rightarrow $\Theta(n+k)$

Counting sort \rightarrow $\Theta(n+k)$

Stable \rightarrow $\Theta(n+k)$

Will Stable Counting per \rightarrow will count \leftarrow will be well *



Radix_sort(A, d, n, k):

for (i=1 down to d):

sort A on i_th digit using a stable sorting algorithm (like counting sort)

$O(Kn)$ time, $O(1)$ space

switches $\Rightarrow \Theta(d(n+K))$

: Counting sort $\Theta(n^2)$ time, $O(n^2)$ space, $O(n)$ switches

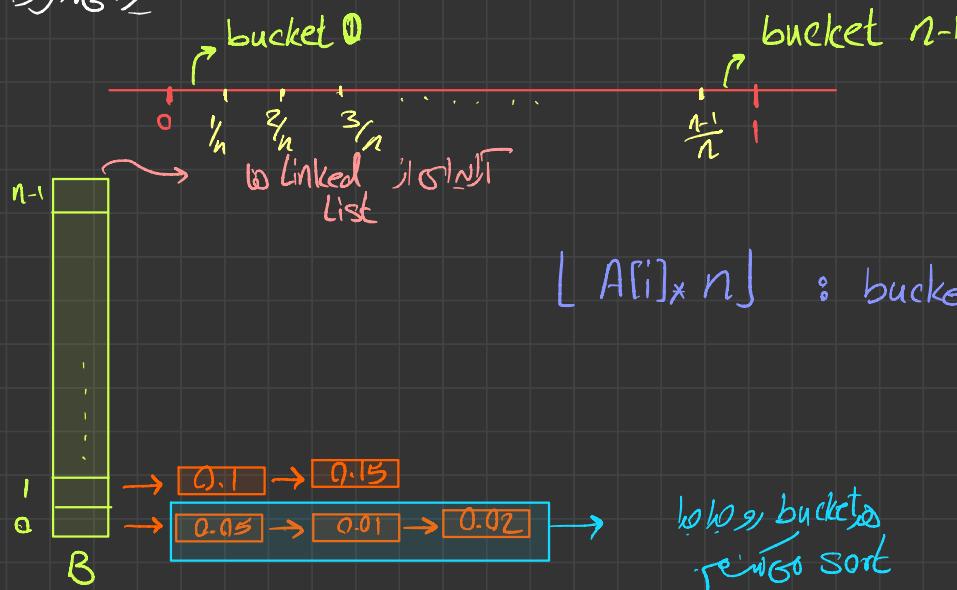
Counting sort $\Theta(n+n^2)$

edition 2 - 83.4 page

(1402, 1, 26) : العنوان

Bucket Sort ✓

الgoritam ✓



$[A[i] * n]$: bucket ($A[i]$) ✓

```
Bucket_sort(A, n):
    for(i=1 to n):
        insert A[i] into Bucket B[(A((i)*n))]
    for (i=0 to n-1):
        insertion sort Bucket B[i]
    concatenate Bucket B[0] ... B[n-1]
```

$O(n)$

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

$$T(n) \leq O(n) \quad \leftarrow \quad n=1 : \text{Call } C_{\text{start}}$$

$$T(n) \leq O(n^2) \quad \leftarrow \quad \begin{aligned} n_i &\leq 0 & \text{for } i \neq k \\ n_k &\leq n & \\ 0 &\leq k < n \end{aligned} \quad : \text{End Call}$$

: Call \tilde{C}_{start}

$$E(T(n)) = O(n) + \sum_{i=0}^{n-1} O(E(n_i^2)) \leq O(n) + \sum_{i=0}^{n-1} O(2 - \frac{1}{n}) = O(n) + O(n) \cdot O(1)$$

$$\leq O(n)$$

$$: E(n_i^2) \leq 2 - \frac{1}{n} \quad \tilde{C}_{\text{start}}$$

$\sum_{j=1}^n (i \in A[j]) \leq n$

now,

$$X_{ij} = \left\{ \begin{array}{l} \text{Call } A[i] \\ \text{if } i \in A[j] \end{array} \right\} = \begin{cases} 1 & \\ 0 & \end{cases}$$

$$n_i = \sum_{j=1}^n X_{ij} \rightarrow (n_i)^2 = \left(\sum_{j=1}^n X_{ij} \right)^2 = \sum_{j=1}^n \sum_{k=1}^n X_{ij} \cdot X_{ik} \leq$$

$$\sum_{j=1}^n (X_{ij})^2 + \sum_{j=1}^n \sum_{k=1}^n X_{ij} \cdot X_{ik} \quad \begin{array}{l} \text{if } j \neq k \\ \text{otherwise} \end{array}$$

$$= E[X_{ij}] = \frac{1}{n} \quad \boxed{= E[X_{ij}] \cdot E[X_{ik}]}$$

$$E[n_i^2] \leq \sum_{j=1}^n \boxed{E[X_{ij}^2]} + \sum_{j=1}^n \boxed{E[X_{ij} \cdot X_{ik}]}$$

$$E[n_i^2] = \underbrace{\sum_{j=1}^n 1/n}_{\frac{n^2-n}{n^2}} + \underbrace{\sum_{j=1}^n \sum_{k=1}^n 1/n \cdot 1/n}_{1/n} = 2 - 1/n$$

$$E[n_i] = \sum_{j=1}^n E[X_{ij}] = \sum_{j=1}^n 1/n = 1 \longrightarrow \text{bucket, } \text{bucket, } \text{bucket, } \text{bucket, } \text{bucket}$$

الحل 1 : Selection Sort

الحل 2 : Selection Sort

إذا $k \ll n$ \rightarrow فيتم ترتيب n عن طريق K مرات

الحل 1 : Selection Sort : $\Omega(n \lg n)$

$O(n)$ \leftarrow Minimization $\leftarrow K=1$ \leftarrow حل 1

$O(n)$ \leftarrow Maximization $\leftarrow K=n$

الحل 2 : عد المقارنات $\leftarrow O(Kn)$

الحل 3 : $O(n) + O(n \lg n)$ \leftarrow $O(n \lg n)$ \leftarrow $O(n) + O(n \lg n)$ \leftarrow $O(n \lg n)$

الحل 4 : ساقط درجات وحده دوسي افروز

$O(n \lg n) + O(\lg n)$

\downarrow

\downarrow $O(n \lg k)$ \leftarrow $O(n \lg k)$

الحل 5 : با $A[1:k]$: $O(n \lg k)$ \leftarrow $O(n \lg k)$

الحل 5 : $O(n \lg k)$ \leftarrow $O(n \lg k)$ \leftarrow $O(n \lg k)$

الحل 5 : $O(n \lg k)$ \leftarrow $O(n \lg k)$ \leftarrow $O(n \lg k)$

$O(k \lg n) + O(n \lg k) \leq O(n \lg k)$



: الـ Quick Selection

$$\text{مقدمة } i = l = q - p + 1$$

```
selection(A, p, r, k):
    if (p==r):
        return A[p] ] → O(1)
    q = partition(A, p, r) → O(n)
    i = q-p+1
    if (k==i):
        return A[q]
    if (k < i):
        return selection(A, p, q-1, k) → O(1)
    else:
        return selection(A, q+1, r, k-i) → O(n)
```

جـ نـ جـ الـ Quick Sort

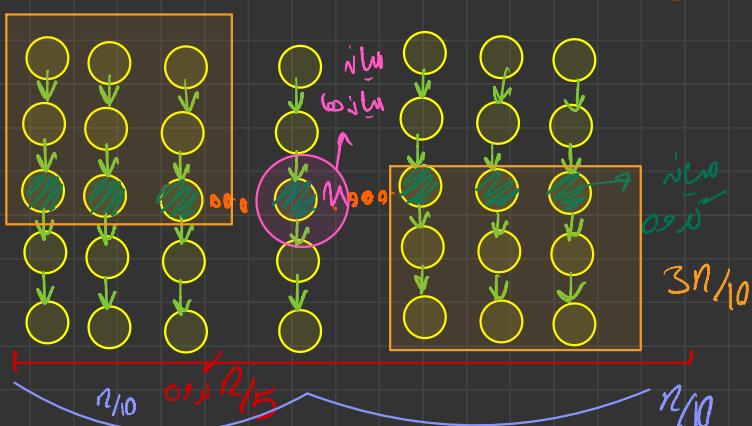
$$T(n) = T(n-1) + O(n) = O(n^2)$$

$M = n/4$: عدد المـ

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

$M = n/2$: عدد المـ

$$3n/10$$



الـ Selection الـ Quick

مـ اـ جـ 3n/10 دـ حـ *

مـ اـ جـ 7n/10 دـ حـ *

مـ اـ جـ 3n/10 دـ حـ *

مـ اـ جـ 7n/10 دـ حـ *

(1402, 2, 4) ~~possible~~ ^{possible}

میں اللعرن : ✓

العزم : $O(n)$
١- اعداد n بوقت $O(n)$ كـ $\sum_{i=1}^n$ بعض i و $\prod_{i=1}^n$ و $\max_{i=1}^n$ a_i $\min_{i=1}^n$ a_i

Ques. Explain Selection Sort, Insertion Sort with Example - 2

لـ 3 مـ 4 نـ 5 كـ 6 مـ 7 كـ 8 مـ 9

$$T(n) = \begin{cases} n & \text{if } n \leq 1 \\ T\left(\frac{n}{2}\right) + O(n) & \text{otherwise} \end{cases}$$

$$\text{EWD} \quad \text{using } \rightarrow T(n) = T\left(\frac{7n}{10}\right) + T\left(\frac{n}{5}\right) + O(n) = O(n)$$

الآن، بحسب $O(n)$ ، n هو T (لدينا n) *

↳ example : $\mathcal{ZT}(n/2) + O(n)$

$O(n \lg n)$

تحليل سعر سال (Amortized Analysis)

لـ $\sum_{i=1}^n c_i$ ، c_i ايـ $O(1)$ operation $\forall i \in \{1, \dots, n\}$

$$\left. \begin{array}{l} O(1) \times n \leftarrow \sum_{i=1}^n (n - c_i) \\ O(n) \times 1 \leftarrow \sum_{i=1}^n c_i \end{array} \right\}$$

تحليل:

$\sum_{i=1}^n c_i = O(n)$ \Rightarrow $n \times O(n) = O(n^2)$

$$\frac{\sum_{i=1}^n c_i}{n} = (n - c) \times O(1) + c \times O(n) = O(n)$$

تحليل متوسط

تحليل متوسط

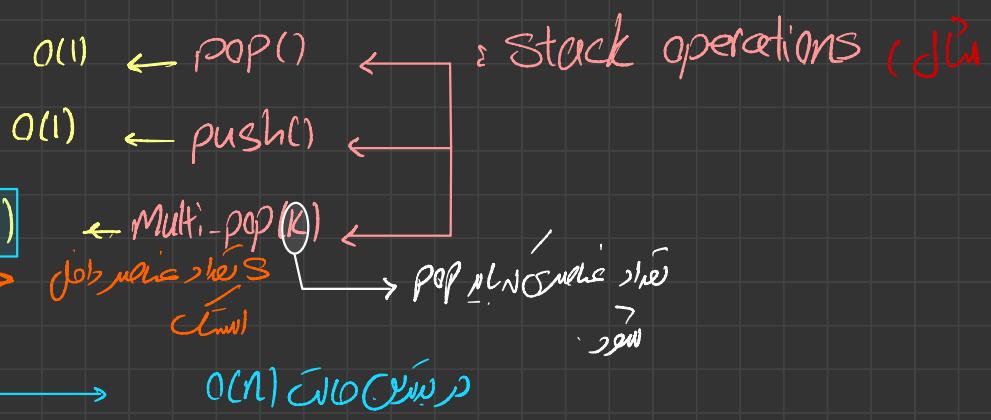
$$\frac{\sum_{i=1}^n c_i}{n} = O(n) \quad \text{aggregate analysis } ①$$

analysis method ②

$\sum_{i=1}^n c_i > \sum_{i=1}^n c_i$ \Rightarrow $\sum_{i=1}^n c_i = O(n)$ \Rightarrow $\sum_{i=1}^n c_i = O(n)$ \Rightarrow $\sum_{i=1}^n c_i = O(n)$

potentiated function ③

دی ③ فقط c_i $\forall i \in \{1, \dots, n\}$ \Rightarrow $c_i = O(1)$ $\forall i \in \{1, \dots, n\}$



→ pop object → push, stack, push (1st)
 $O(n) = \text{push } 1 \leftarrow \dots \leftarrow \text{push } n \rightarrow \text{push } 2 \leftarrow \dots \leftarrow \text{push } 1$

Info about 6 bits

(1402, 2, 9) = position needed

Binary Counter : Jlu

0	0	0	1	0	
0	0	0	1	1) ₂
0	0	1	1	0) ₁
0	0	1	1	1) ₃
0	1	1	0	0	

(log n) ← increment of info needed
↓
↓
↓
 $n \cdot O(\log n) \rightarrow O(n \log n)$

1	1	1	1	1	
0	0	0	1	0) ₄
					\rightarrow Info need
$n/8$	$n/4$	$n/2$	n		$\sum_{i=0}^{\log_2 n - 1} n/2^i \leq n \sum_{i=0}^{\infty} \frac{1}{2^i}$

$$\sum_{i=0}^{\log_2 n - 1} n/2^i \leq n \sum_{i=0}^{\infty} \frac{1}{2^i}$$

increment of info needed = $\frac{1}{\text{bits needed}} = \frac{2n}{n} = 2$ ✓

(وسن عالی طرایق الوریم :

- ① وسیع جستجوی خودش (Divide & Conquer)
- ② وسیع سیاست دینامیک (Dynamic programming)
- ③ وسیع جستجوی خودش (Greedy)
- ④ حلقه و خروجی های محدود (backtracking , branch & bound)

Quick Sort merge sort : divide & conquer

partition از $O(n)$ ل
اگر نیاز نداشتم اینجا
نمایش نمی شد
 $\hookrightarrow O(n)$

قسم آن را در دو قسم
دو قسم ایجاد کرد
 $\downarrow O(1)$ $n/2$ $O(1)$

حال ۱۶

QuickSort($A, P, Q-1$)
QuickSort($A, Q+1, R$)
 $\hookrightarrow T(k) + T(n-k-1)$

merge-Sort(A, P, Q)
merge-Sort($A, Q+1, R$)
 $\hookrightarrow T(n/2) + T(n/2)$

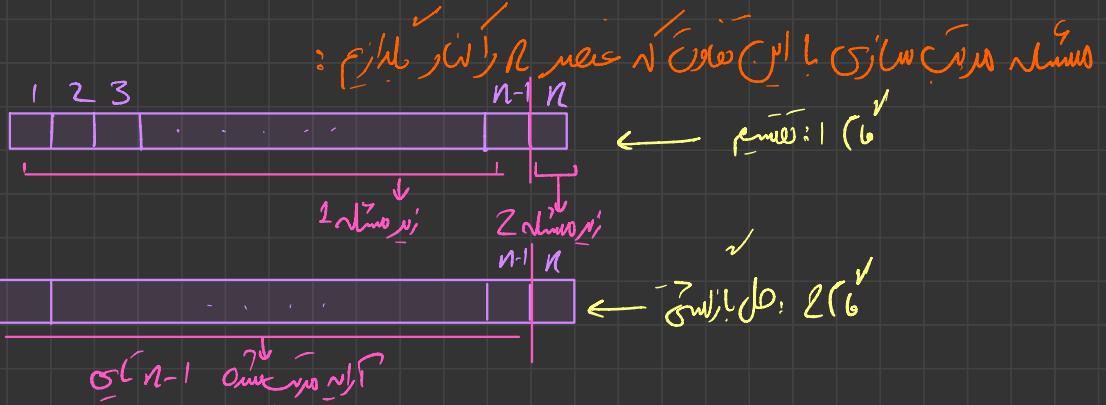
حال ۱۶

merge(A, P, Q, R)
 $\hookrightarrow O(n)$

حال ۱۶

$O(n \log n)$
 $O(n^2) \rightarrow \text{این} O(n^2)$

$O(n \log n)$



insertion-sort ပေါ်လေ့ရှိခဲ့သူ၏ အတွက်
မြစ်လေ့ရှိခဲ့သူ၏ အတွက်
မြစ်လေ့ရှိခဲ့သူ၏ အတွက်

```
Recursive_insertion_sort(A, n):
    if (n>1): → O(1)
        Recursive_insertion_sort(A, n-1) → T(n-1)
        key = A[n]
        i=n-1
        while (i>0 and key <A[i]): → O(n)
            A[i+1] = A[i]
            i=i-1
        A[i+1]= key
```

$T(n), T(n-1), O(n) = \underbrace{O(n^2)}$
- ငါးခါး အတွက်

(1402, 2, 16) : حل مجهود

مقدار مجهود

مقدار مجهود = $\frac{1}{2} n(n+1)$

$$\text{مقدار مجهود} \Rightarrow \frac{n(n-1)}{2}$$

مقدار مجهود = $\frac{1}{2} n(n-1)$

1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8
2	1	4	5	6	7	8	3
3	4	1	6	7	8	5	2
4	3	2	1	8	5	6	7
5		3	2	1			
6			3	2	1		
7				3	2	1	
8					3	2	1

روزنگاری فلسفی و معاشر:

	1	2	3	4	5	6	7
X 09:00	2	3	4	5	8	7	6
A 09:00	1	4	3	6	5	8	7
B 09:00	3	1	2	7	6	5	8
C 09:00	4	3	2	1	8	7	6
D 09:00	5	6	7	8	1	2	3
E 09:00	6	5	8	7	2	3	4
F 09:00	7	8	5	6	3	4	1
G 09:00	8	7	6	5	4	1	2

* الاربعاء عالى فرجو، نك تتم مبارى امارات، ونجد
ما نعم كلام بارى كلنا. درواع ونحب اهل سعادت

اللهم اذرا مسأتم سمع هاتون فرقكم

(1)

	1	2	3	4	5	
1	2	3	-			
2	1	-	3			
3	-	1	2			
4	5	6	-			
5	4	-	6			
6	-	4	5			

اللهم اذرا مسأتم سمع هاتون فرقكم

1	2	3	4
2	1	4	3
3	4	1	2
4	3	2	1

اللهم اذرا مسأتم سمع هاتون فرقكم

(2)

	1	2	3	4	5	
1	2	3	4	5	6	
2	1	5	3	6	4	
3	6	1	2	4	5	
4	5	6	1	3	2	1
5	4	2	6	1	3	2
6	3	4	5	2	1	3

وَهُنَّ الَّذِينَ لَمْ يَلْمِدُ

$$A = \begin{matrix} n \\ \boxed{} & \boxed{} & \boxed{} & \cdots & \boxed{} & \boxed{} \end{matrix}$$

اگر وسیع نہ رہے تو ممکن ہے کہ اس سے کوئی ایسا

$$\Theta(n^2) = n^2 + \frac{n(n-1)}{2} \quad \left\{ \begin{array}{l} \text{Gesamt } \sim n^2 \\ \text{Zusatz } \sim \frac{n(n-1)}{2} \end{array} \right.$$



الخطوة الأولى في حساب حاصل ضرب ماتريسي هي:

$$A = A_2 \times 2^{\frac{n}{2}} + A_1$$



$$B = B_2 \times 2^{\frac{n}{2}} + B_1$$

$$\begin{aligned}
 C &= A \times B = (A_2 \times 2^{\frac{n}{2}} + A_1) \times (B_2 \times 2^{\frac{n}{2}} + B_1) \\
 &= \underbrace{A_1 B_1}_{1} + \underbrace{A_2 B_2 \times 2^n}_{2} + \underbrace{(A_1 B_2 + B_1 A_2) 2^{\frac{n}{2}}}_{3} + \underbrace{(A_1 B_2 + B_1 A_2) 2^{\frac{n}{2}}}_{4}
 \end{aligned}$$

n نهاية $\frac{n}{2}$ نهاية

الخطوة الثانية في حساب حاصل ضرب ماتريسي هي:

$$T(n) = 4T\left(\frac{n}{2}\right) + O(1) + O(n) = \Theta(n^2)$$

↗ نهاية ↗ نهاية
 ↗ نهاية ↗ نهاية