مهرناز خوش‌دوست آزاد 99521253

پویا رحیمی 99521289

Quiz①: The main charasteristic of the sorting Algorithm used in Radix sort is Stable Sort.

radix sort از هر رقم به عنوان key استفاده می‌کند و از یک Stable Sort (مانند Counting Sort یا bucket_Sort) در هر مرتبه استفاده می‌کند. در بخش دوم
هر سؤال هم، Quick Sort ، O(n logn) و در حالت worst_case
$O(n^2)$ می‌باشد در حالی است که Counting Sort از مرتبهٔ O(n+k) می‌باشد
پس Counting Sort نسبت به Quick Sort بهینه‌تر می‌باشد

Subject : .... Date :

Quiz ②: Radix Sort

Best: زمانی این حالت رخ می‌دهد که آرایه sort شده باشد

Average: زمانی رخ می‌دهد که آرایه نه به صورت شده و نه inversly_sort باشد $T(n)=O(nk)$

Worst: عناصر به صورت inversly_sort می‌باشند $T(n)=O(nk)$

Counting Sort

Best: آرایه sort شده می‌باشد $T(n)=\Omega(n+k)$

Average: عناصر نه sort شده و نه inversly-Sort می‌باشند $T(n)=O(n+k)$

Worst: آرایه inversly-Sort می‌باشد $T(n)=O(n+k)$

Bucket_Sort

Best: آرایه sort شده می‌باشد و در این حالت time-complexity کمترین می‌باشد
و اگر از insertion Sort برای عناصر داخل bucket ها استفاده کنیم، اردر خطی
خواهد بود و $T(n)=O(n+k)$ می‌باشد که در آن $O(n)$ برای ساخت bucket ها
و $O(k)$ برای Sort bucket ها.

Average: زمانی اتفاق می‌افتد که عناصر آرایه به هم ریخته باشند (نه حالت صعودی و
نزولی به هم ریخته باشند) در این حالت در bucket Sort leaner time رخ می‌دهد
حتی اگر عناصر به طور یکنواخت توزیع شده باشند پس $T(n)=O(n+k)$

Worst: زمانی رخ می‌دهد که عناصر از فاصله‌ی نزدیک در آرایه باشند، به همین دلیل در یک
bucket قرار می‌گیرند و وقتی که این عناصر ترتیب هم به صورت inversly Sort
باشند time complexity افزایش پیدا می‌کند $T(n)=O(n^2)$

Subject : ....   Date :   .   .

Quiz ③: باید از روش DAC به صورت هوشمندانه استفاده کنیم، یعنی باید از آن در جاهایی کاهش time complexity استفاده کرد و جاهای افزایش کمی نمی‌توان این کار را کرد.

گاهی در مسئله با divide کردن، مسئله پیچیده‌تر می‌شود time complexity آن افزایش پیدا می‌کند، پس متوقف کردن عملیات در زمان مناسب می‌تواند بسیار مفید باشد چون از افزایش time complexity جلوگیری می‌کند. علاوه بر آن در ادامه این روش نیز space و time بیشتری نیاز داریم (نسبت به حل مسئله با Brute Force) که این باعث می‌شود که بهینه نباشد.

برای مثال برای اینکه احتمال worst case را کاهش دهیم، در quick sort از DAC استفاده می‌شود و در حالت worst case نیز داریم $T(n) = O(n^2)$ ولی با اعمال تغییرات می‌توان تعداد دفعات divide کردن را کاهش داد