COP 3502 - 10/20/21

0. Reminders - Rec Can > F2 (but RP3 → on your own)
   Study Group Turn In - **UP TODAY**
1. Quick Sort
2. Quiz Details

Last Time: Partition

low — high

← 10, 12, 3, 6, 15, 99, 8, ~~8~~ 7 →

pivot — low — high

10 7, 3, 6, 15, 99, 8, 12 (swap 12, 7)

10 7, 3, 6, 8, 99, 15, 12 (swap 15, 8)

low — high

low, high crossed over
swap indexes high and pivot

8, 7, 3, 6, $\boxed{10}$, 99, 15, 12

$\underbrace{8, 7, 3, 6}_{< 10}$ needs to be sorted — $\underbrace{99, 15, 12}_{> 10}$ needs to be sorted

```
void quicksort(int* array, int low, int high) {
    if (low >= high) return;
    int partIndex = partition(array, low, high);
    quicksort(array, low, partIndex-1);
    quicksort(array, partIndex+1, high);
}
```

Quick Sort vs Merge Sort

ADV QSORT: It is in place (no extra mallocing of temp arrays)

ADV MERGE: worst case $O(n \lg n)$ BECAUSE OF even split always!

In practice, quick sort is faster because it's in place and makes up for less than perfect splits.

Best Case QSORT = split in 1/2 every time

$$T(n) = \underbrace{T(\tfrac{n}{2})}_{QSL} + \underbrace{T(\tfrac{n}{2})}_{QSR} + \underbrace{O(n)}_{\text{partition}}$$

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

Worst Case QSORT = Every split is 0 / n-1

$$T(n) = \underbrace{T(n-1)}_{\text{QS BigSide}} + \underbrace{O(n)}_{\text{Partition}}$$

$$= O(n^2)$$

Really bad case for quick sort on ints:

5, 5, 5, 5, ... 5

Usually after partition is chosen all values $\leq$ partition go to 1 side $\Rightarrow$ ensures WORST CASE SPLIT!

How to fix?

B.C.: if (isSorted(array, low, high)) return;

In general, choice of the partition element is IMPORTANT!

Strategies

1. always pick item on the left
2. pick a random element in between index low + high and swap it into index low.
3. Median of 3 or 5 strategy pick 3 or 5 rnd elems then choose the median of them.

smaller

Do this if array is large

more time choose

better split

Let $T(n)$ = avg case run time of quick sort.

Left	Right	Prob
0	n-1	1/n
1	n-2	1/n
2	n-3	1/n
⋮		
n-1	0	1/n

$$T(n) = \frac{1}{n}\left(T(n-1) + T(0) + O(n)\right) + \frac{1}{n}\left(T(n-2) + T(1) + O(n)\right) + \frac{1}{n}\left(T(n-3) + T(2) + O(n)\right) + \vdots + \frac{1}{n}\left(T(0) + T(n-1) + O(n)\right)$$

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

$$T(n) = \frac{1}{n}\left[2T(0) + 2T(1) + 2T(2) + \dots 2T(n-1)\right] + n$$

$$nT(n) = \left[\cancel{2T(0)} + \cancel{2T(1)} + \dots + 2T(n-1)\right] + n^2$$

$$-(n-1)T(n-1) = \left[\cancel{2T(0)} + \cancel{2T(1)} + \dots + \cancel{2T(n-2)}\right] + (n-1)^2$$

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

$$nT(n) - \underline{(n-1)T(n-1)} = \underline{2T(n-1)} + 2n - 1$$

$$nT(n) = (n-1+2)\,T(n-1) + (2n-1)$$

$$\frac{nT(n)}{n(n+1)} = \frac{(n+1)T(n-1)}{n(n+1)} + \frac{(2n-1)}{n(n+1)}$$

$$\frac{T(n)}{n+1} = \boxed{\frac{T(n-1)}{n}} + \left(\frac{-1}{n} + \frac{3}{n+1}\right)$$

Let $S(n) = \dfrac{T(n)}{n+1}$

$$S(n) = S(n-1) + \left(-\frac{1}{n} + \frac{3}{n+1}\right), \quad S(0) = 0$$

$$= \sum_{i=1}^{n}\left(-\frac{1}{n} + \frac{3}{n+1}\right)$$

$$\sim -\ln n + 3\ln(n+1)$$

$$= O(\lg n)$$

$$O(\lg n) = \frac{T(n)}{n+1} \Rightarrow \underline{(n+1)\,O(\lg n)} = T(n)$$

$$\Rightarrow \boxed{T(n) = O(n \lg n)}$$

$\sum_{i=1}^{n} \frac{1}{i} = H_n$

$\sim \ln n$

QUIZ 3

Topics: Algorithm Analysis
- Run Time Problems ($O(n^2)$ takes 500 ms $n=500$ ...
- Code Seg Analysis
- Summations
- Recurrence Relations

Sorting
- Bubble Sort
- Insertion Sort
- Selection Sort (select MAX)
- Merge Sort
- Quick Sort

$\boxed{\text{NO NOTES}}$ $\boxed{\text{NO CALCULATOR}}$

Given summation formulas