COP 3502 10/15/21

SORTING

(1) Bubble Sort

(2) Insertion Sort

(3) Selection Sort

(4) Overall Ranking - UCF Prog Team Quals 21

Bubble Sort

King ~~Hill~~ of the Hill

8, 2, 9, 3, 5, 1, 6, 7

1 iter: 2 8 3 5 1 6 7 | 9 (at end of 1st iteration)

2 3 5 1 6 7 8 9 — end iter 2nd

2 3 1 5 6 7 8 9 — end 3rd iter

2 1 3 5 6 7 8 9 = 4th iter

1 2 3 5 6 7 8 9 = 5th iter

1 2 3 5 6 7 8 9 — 6th iter

1 2 3 5 6 7 8 9 — 7th

→ no swaps (on wiki => they break ends a/s)

Runtime: 1st iter: n steps; 2nd iter: $n-1$ steps; 3rd: $n-2$ steps; ⋮

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

Sort (n items)	Best	Avg	Worst
Bubble	$O(n^2)$	$O(n^2)$	$O(n^2)$
Insertion	$O(n)$ sorted	$O(n^2)$	$O(n^2)$
Selection	$O(n^2)$	$O(n^2)$	$O(n^2)$

Insertion Sort

8 | 2, 9, 3, 5, 1, 6, 7

2 8 | 9, 3, 5, 1, 6, 7 end 1st iter

2 8 9 | 3, 5, 1, 6, 7, end 2nd iter

2 3 8 9 | 5, 1, 6, 7 end 3rd iter

2 3 5 8 9 | 1, 6, 7 = 4^{th} =

1 2 3 5 8 9 | 6, 7 = 5^{th} =

1 2 3 5 6 8 9 | 7 6^{th} iter

1 2 3 5 6 7 8 9 7^{th} iter

Best = sorted = $1+1+1+\cdots 1 = O(n)$

Worst = reverse = $1+2+3+\cdots n-1 = \frac{(n-1)n}{2} = O(n^2)$

Avg Case = $\frac{1}{2} + \frac{2}{2} + \frac{3}{2} + \cdots \frac{n-1}{2} = \frac{(n-1)n}{4} = O(n^2)$

↓ expected # insertions

Selection Sort

8, 2, 9, 3, 5, 1, 6, 7 — maxI = 0 2

8, 2, 7, 3, 5, 1, 6, 9 — end 1st iter (swap 7, 9)

6, 2, 7, 3, 5, 1, 8, 9 — end 2nd iter (swap 8, 6)

6, 2, 1, 3, 5, 7, 8, 9 — = 3rd iter (swap 7, 1)

5, 2, 1, 3, 6, 7, 8, 9 — = 4th iter (swap 6, 5)

3, 2, 1, 5, 6, 7, 8, 9 — = 5th iter (swap 3, 5)

1, 2, 3 5 6, 7, 8, 9 — = 6th iter

1, 2, 3, 5, 6, 7, 8, 9 — = 7th iter (swap 2, 2)

$n + (n-1) + (n-2) \cdots + 1 = \frac{n(n+1)}{2} = O(n^2)$

UCF Local Contest (Qualifying Round) — September 4, 2021

Overall Ranking

filename: `ranking`
Difficulty Level: `Easy-Medium`
Time Limit: `5 seconds`

The Regional Competitions of ICPC (International Collegiate Programming Contest) allow universities to enter more than one team in the contest. The scoresheet for the contest will list the ranking for each team. For example, if UCF has three teams, GT two teams, and Auburn four teams, the final team ranking may look like:

```
1. Auburn
2. GT
3. UCF
4. UCF
5. UCF
6. Auburn
7. Auburn
8. GT
9. Auburn
```

The regional contest does not show the "overall performance" of each university which is the average of the team rankings for the university. For the above scoresheet, Auburn has the overall performance of "(1 + 6 + 7 + 9) / 4 = 5.75"; UCF has the overall performance of "(3 + 4 + 5) / 3 = 4.0"; and GT has the overall performance of "(2 + 8) / 2 = 5.0". So, the overall ranking of the universities will be:

```
1. UCF
2. GT
3. Auburn
```

Note that lower average indicates better performance, hence higher university ranking.

Assume that the universities will not have the same overall performance, i.e., no need for tiebreaker.

The Problem:

Given the scoresheet for a contest with each university having one or more teams, find the university ranking based on the overall performances.

The Input:

The first input line contains an integer, n ($2 \le n \le 100$), indicating the number of teams in the regional contest. The teams are listed on the following n input lines, one per line, in the order of

team ranking at the regional. Each team name will be 1 to 20 letters (uppercase and/or lowercase). Assume there will be at least two universities in the input.

The Output:

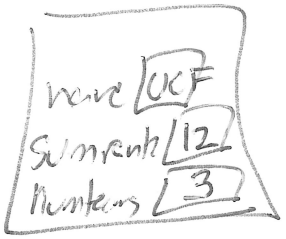
Print the universities, one per line, in the order of overall performance.

Sample Input **Sample Output**

Sample Input	Sample Output
9 Auburn GT UCF UCF UCF Auburn Auburn GT Auburn	UCF GT Auburn
7 UofX UofA UofC UofB UofA UofB UofY	UofX UofC UofA UofB UofY
8 UA UB UA UA UA ua Ua uA	UB UA ua Ua uA

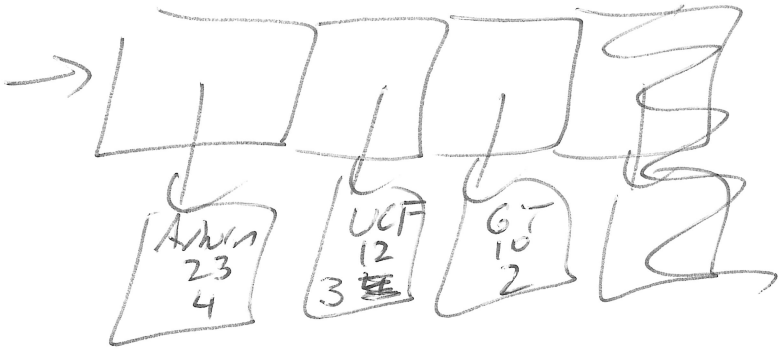
Ranking Strategy

char** rawdata



struct School**

convert(char** rawdata, int n, int* ptrNewSize)



sort(School** list, int size)

cmp(School* itemA, School* itemB)

$$\frac{S1\rightarrow sumR}{S1\rightarrow numT} < \frac{S2\rightarrow sumR}{S2\rightarrow numT}$$

$$S1\rightarrow sumR * S2\rightarrow numT < S1\rightarrow numT * S2\rightarrow sumR$$

if true iff $S1\rightarrow sumR * S2\rightarrow numT - S1\rightarrow numT * S2 sumR < 0$