CS-205, Assignment –V

Assignment Date: 28/08/2017 **Submission Deadline: 03/09/2017**

A. Let n denote the size of an array A (of integers) which we want to sort.

The performance of quick sort is very sensitive to the initial distribution of A and also to the choice of the pivot. We need to work with random, sorted, and almost sorted arrays. Also, the following ways of choosing the pivot for partitioning is to be experimented with.

FIRST	Choose the first element A[0] as the pivot.
RANDOM	Choose A[r] as the pivot for a random r $\in \{0, 1, 2, \ldots, n-1\}$.
MEDIAN OF THREE	Choose r, s,t $\in \{0, 1, 2, \ldots, n-1\}$, and take the median of A[r], A[s], A[t] as the pivot. Use both the following choices for r, s,t. (1) r = 0, s = n/2, and t = n − 1. (2) r = n/4, s = n/2, and t = 3n/4.

Write a function quicksort(A, n, pivot type) to sort an array A with n (non-negative) integers. The third argument pivot type indicates how you choose the pivot for partitioning:

0 means FIRST,
1 means RANDOM,
2 means MEDIAN OF THREE (1), and
3 means MEDIAN OF THREE (2).

In the main() function

For $n = 10^k$, k = 4, 5, 6, 7, and for each choice of the pivot type, repeat the following:

Populate an array A with n random integers in the range 0 to 109 −1. The random choices of elements may lead to repetitions. You do not need to avoid repetitions. Sort A by calling quicksort. Now, A is sorted. Call quicksort again on this sorted array. Now, swap a few elements in A (take k = n/100, and swap k randomly chosen pairs of A). This gives you an almost sorted array. Run quicksort again on this array. Report the times taken by the three calls. Do not include the array-preparation times.

A random integer in the range [0, $10^9 - 1$] can be generated by the call (#include <stdlib.h>):

```
A[i] = rand() % 1000000000;
```

Different runs of your program should handle different random sequences. In order to achieve that, write the following line at the beginning of your main() function. You need to include <stdlib.h> and <time.h> .

```
srand((unsigned int)time(NULL));
```

Finally, this is how you can measure (calendar) times in C programs (include <time.h>).

```
clock_t c1, c2;

double runtime;

c1 = clock();

<- - - Code for which the running time is to be measured - - ->

c2 = clock();

runtime = (double)(c2 - c1) / (double)CLOCKS_PER_SEC;          /* Time in seconds */
```

SAMPLE OUTPUT : Present your output in the following format.

Performance of Quick Sort

n	Pivot type	Random	Sorted	Almost Sorted
10000	FIRST	0.001	0.002	0.002
10000	RANDOM	0.001	0.001	0.001
10000	MEDIAN OF THREE 1	0.001	0.002	0.001
10000	MEDIAN OF THREE 2	0.001	0.000	0.001
100000	FIRST	-	-	-
100000	RANDOM	-	-	-
100000	MEDIAN OF THREE 1	-	-	-
100000	MEDIAN OF THREE 2	-	-	-
1000000	FIRST	-	-	-
1000000	RANDOM	-	-	-
1000000	MEDIAN OF THREE 1	-	-	-
1000000	MEDIAN OF THREE 2	-	-	-
10000000	FIRST	-	-	-
10000000	RANDOM	-	-	-
10000000	MEDIAN OF THREE 1	-	-	-
10000000	MEDIAN OF THREE 2	-	-	-

Upload file Assign5A.c

B. You are given name of few test cricket players with their runs in international test cricket career. Sort the player in ascending order based on their score. If scores of more than one player is same then order those players in lexicographical order (when you try to sort name in lexicographical order , find out the name with minimum characters let's say it is ***m*** then consider first m characters of all the words of which you have to break the tie). Use linear sorting algorithm.

Dravid	13288
Gambhir	11953
Gavaskar	10122
Ponting	13378
jayawardene	11814
Lara	11953
Kallis	13289
Chanderpaul	11867
Tendulkar	15921
Gooch	13289
Cook	1586
Sangakkara	12400
Sarwan	11953
IRBell	13289

Upload file Assign5B.c