

Aayush Shah

19BCE245

28 July 2021

# Design and Analysis of Algorithms

## Practical 2

### • Code :

```
1. #include <stdio.h>
2. #include <time.h>
3. #include <stdlib.h>
4.
5.
6. void swap(int *xp, int *yp){
7.     int temp = *xp;
8.     *xp = *yp;
9.     *yp = temp;
10.}
11.
12.int partition (int arr[], int low, int high)
13.{
14.    int pivot = arr[high];
15.    int i = (low - 1);
16.    for (int j = low; j <= high - 1; j++){
17.        if (arr[j] < pivot){
18.            i++;
19.            swap(&arr[i], &arr[j]);
20.        }
21.    }
22.    swap(&arr[i + 1], &arr[high]);
23.    return (i + 1);
24.}
25.
26.void quickSort(int arr[], int low, int high){
27.    if (low < high){
28.        int pi = partition(arr, low, high);
29.        quickSort(arr, low, pi - 1);
```

```
30.         quickSort(arr, pi + 1, high);
31.     }
32.}
33.
34.void printArray(int arr[], int size){
35.    int i;
36.    for (i = 0; i < size; i++)
37.        printf("%d ",arr[i]);
38.}
39.
40.int main() {
41.    int lower = 0;
42.    int upper = 100000;
43.    int count = 1000000;
44.    int arr[count];
45.
46.    clock_t start, end;
47.    double cpu_time_used;
48.
49.    srand(time(0));
50.
51.    for (int i = 0; i < count; i++){
52.        int num = (rand() % (upper - lower + 1)) + lower;
53.        arr[i] = num;
54.    }
55.
56.//    int arr[] = {64, 34, 25, 12, 22, 11, 90};
57.    int n = sizeof(arr)/sizeof(arr[0]);
58.
59.    //QUICK SORT
60.    start = clock();
61.    quickSort(arr, 0, n-1);
62.    end = clock();
63.    cpu_time_used = ((double) (end - start)) /
        CLOCKS_PER_SEC;
64.    printf("Sorted array in %f seconds with QUICK sort:
        \n",cpu_time_used);
65.//    printArray(arr, n);
66.
67.
68.    return 0;
69.}
```

• **Execution time in seconds :**

Numbers	Time taken
500	0.000051
1000	0.000133
5000	0.000791
10000	0.001538
50000	0.007806
100000	0.016710
150000	0.023415
160000	0.025759
170000	0.028243
200000	0.031532
500000	0.082039
1000000	0.166498
1500000	0.259056
2000000	0.358336

The screenshot shows a code editor window titled 'algo\_quicksort.c'. The code includes `<stdio.h>`, `<time.h>`, and `<stdlib.h>`. It defines a `swap` function and implements a quicksort algorithm. The output window shows the message: 'Sorted array in 0.355541 seconds with QUICK sort:'. The status bar at the bottom indicates 'Run Succeeded', 'Time 406 ms', and 'Peak Memory 8.7M'.

```

1  #include <stdio.h>
2  #include <time.h>
3  #include <stdlib.h>
4
5
6  void swap(int *xp, int *yp){
7      int temp = *xp;
8      *xp = *yp;
9      *yp = temp;
10 }
11
12 int partition(int arr[], int low, int high){
13     int pivot = arr[high];
14     int i = low - 1;
15     for (int j = low; j < high; j++){
16         if (arr[j] < pivot){
17             i++;
18             swap(&arr[i], &arr[j]);
19         }
20     }
21     swap(&arr[i+1], &arr[high]);
22     return i+1;
23 }
24
25 void quicksort(int arr[], int low, int high){
26     if (low < high){
27         int pi = partition(arr, low, high);
28         quicksort(arr, low, pi-1);
29         quicksort(arr, pi+1, high);
30     }
31 }
32
33 int main(){
34     int n;
35     printf("Enter the number of elements: ");
36     scanf("%d", &n);
37     int arr[n];
38     printf("Enter the elements: ");
39     for (int i = 0; i < n; i++){
40         scanf("%d", &arr[i]);
41     }
42     time_t start, end;
43     start = clock();
44     quicksort(arr, 0, n-1);
45     end = clock();
46     printf("Sorted array in %f seconds with QUICK sort:", (end - start) / CLOCKS_PER_SEC);
47     return 0;
48 }

```

Sorted array in 0.355541 seconds with QUICK sort:

Run Succeeded | Time 406 ms | Peak Memory 8.7M | Symbol | Tabs: 4 | Line 1, Column 1