

## BLG335E – Assignment1 Report

Name: Ömer Malik Kalembaşı

Student ID: 150180112

### a. How2Run

directory:     cd assignment1  
compile:       g++ -g assignment1.cpp -o assignment1  
run:           ./assignment1

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

[1] + Done      "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-uufh3avp.r0p" 1>"/tmp/Micro
soft-MIEngine-Out-hpa3tcz2.vsg"
test@blg335e:~/hostvolume$ cd assignment1
test@blg335e:~/hostvolume/assignment1$ g++ -g assignment1.cpp -o assignment1
test@blg335e:~/hostvolume/assignment1$ ./assignment1
```

```
32  int partition(vector<float> &arr, vector<float> &id, int low, int high, int &numberOfPartitions){
33
34      numberOfPartitions++;
35
36      if(high > low ){
37          int i = low;
38          int j = low;
39          float pivot = arr[high];
40          float temp = 0;
41
42          while(arr[i] <= pivot && arr[j] <= pivot ){
43
44              if(i < high && j < high){
45                  j++;
46                  i++;
47              }
48              else{
49                  break;
50              }
51          }
52
53          while(arr[j] > pivot ){
54              if(j < high){
55                  j++;
56              }
57              else{
58                  break;
59              }
60
61              temp = arr[i]; //swap avg rating
62              arr[i] = arr[j];
63              arr[j] = temp;
64
65              temp = id[i]; //swap id
66              id[i] = id[j];
67              id[j] = temp;
68
69              if(arr[i] <= pivot){
70                  if(i < high){
71                      i++;
72                  }
73                  else{
74                      break;
75                  }

```

```

75         }
76     }
77 }
78
79     temp = 0;
80
81     if(j == high){
82         temp = arr[i]; //swap avg rating
83         arr[i] = arr[high];
84         arr[high] = temp;
85
86         temp = id[i]; //swap id
87         id[i] = id[j];
88         id[j] = temp;
89     }
90     return i;
91 }
92 }
93
94 void quickSort(vector<float> &arr, vector<float> &id, int low, int high, int &numberOfPartitions, int &numberOfSwaps){
95
96     numberOfSwaps++;
97
98     if(high > low ){
99
100         int p = partition(arr, id, low, high, numberOfPartitions);
101         quickSort(arr, id, low, p-1, numberOfPartitions, numberOfSwaps);
102         quickSort(arr, id, p, high, numberOfPartitions, numberOfSwaps);
103     }
104 }

```

b. Sorted results in .txt

assignment1 > sorted\_books.txt

```

1 799,0
2 1302,0
3 1537,0
4 1549,0
5 2442,0
6 3351,0
7 3479,0
8 3852,0
9 5720,0
10 5934,0
11 6625,0
12 7848,0
13 9337,0
14 9338,0
15 10200,0
16 10536,0
17 12712,0
18 15186,0
19 16806,0
20 17054,0
21 19257,0
22 19668,0
23 19858,0
24 21534,0
25 21536,0
26 21538,0

```

c. Time elapsed (microseconds), number of partitions, number of swaps

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Time elapsed (microseconds): 3005
Number of partitions: 11126
Number of swaps: 22253
[1] + Done      "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-2jobz14o.dcr" 1>"/tmp/Micro
soft-MIEngine-Out-cxywubkq.gum"
test@blg335e:~/hostvolume$
```

Time elapsed (microseconds), number of partitions, number of swaps (half)

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Time elapsed (microseconds): 1215
Number of partitions: 5562
Number of swaps: 11125
[1] + Done      "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-0hfmzk1k.sh2" 1>"/tmp/Micro
soft-MIEngine-Out-3h1hwqb0.do2"
test@blg335e:~/hostvolume$
```

Time elapsed (microseconds), number of partitions, number of swaps (quarter)

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Time elapsed (microseconds): 480
Number of partitions: 2780
Number of swaps: 5561
[1] + Done      "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-b4ph0gy4.btr" 1>"/tmp/Micro
soft-MIEngine-Out-k50nhkoz.3r5"
test@blg335e:~/hostvolume$
```

d.

### Worst case:

This happens when the array is already sorted in reverse order and we choose the pivot as the largest or smallest element. With each partition move, we split the array in half; one element and the rest, until two elements remain.

$$n + (n-1) + (n-2) + (n-3) + (n-4) + \dots + 2 = n*(n+1)/2 + 1 = o(n^2)$$

$$T(n) = o(n^2)$$

**Best case:**

Best case occurs when we select mean as pivot. In this case recursion occurs like  $\log n$ . Array will be divided in branches of equal size. Thus, height of the tree will be minimum. In each branch, we will traverse to the all elements, where  $n$  comes.

$$T(n) = n * \log n$$

**Average case:**

Average case calculated by taking average of all complexities, when occurs all elements divided properly, not nearly descending or ascending.

$$T(n) = n * \log n$$

- e. The worst case happens when the elements of the array are almost sorted. To avoid the worst case, we choose the element farthest from the largest or smallest value as the pivot. It can be mean, median, or just random selection.