**Sam Lazrak**

**CS 303 Algorithms and Data Structures**

**Homework Assignment 5**

**2/13/18**

1. Work the following Exercise from Chapter 8 of the text:
   1. (4 points) Exercise 8.2-1: Using Figure 8.2 as a model, illustrate the operation of COUNTING-SORT on the array A = {6, 0, 2, 0, 1, 3, 4, 6, 1, 3, 2}

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 4 | 6 | 8 | 9 | 9 | 11 |
| 6 | 0 | 2 | 0 | 1 | 3 | 4 | 6 | 1 | 3 | 2 |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 4 | 5 | 8 | 9 | 9 | 11 |
|  |  |  |  |  | 2 |  |  |  |  |  |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 4 | 5 | 7 | 9 | 9 | 11 |
|  |  |  |  |  | 2 |  | 3 |  |  |  |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 3 | 5 | 7 | 9 | 9 | 11 |
|  |  |  | 1 |  | 2 |  | 3 |  |  |  |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 3 | 5 | 7 | 9 | 9 | 10 |
|  |  |  | 1 |  | 2 |  | 3 |  |  | 6 |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 3 | 5 | 7 | 8 | 9 | 10 |
|  |  |  | 1 |  | 2 |  | 3 | 4 |  | 6 |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 3 | 5 | 6 | 8 | 9 | 10 |
|  |  |  | 1 |  | 2 | 3 | 3 | 4 |  | 6 |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 2 | 2 | 5 | 6 | 8 | 9 | 10 |
|  |  | 1 | 1 |  | 2 | 3 | 3 | 4 |  | 6 |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 5 | 6 | 8 | 9 | 10 |
|  | 0 | 1 | 1 |  | 2 | 3 | 3 | 4 |  | 6 |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 4 | 6 | 8 | 9 | 10 |
|  | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 4 |  | 6 |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 2 | 4 | 6 | 8 | 9 | 10 |
| 0 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 4 |  | 6 |

A: 1 2 3 4 5 6 7 8 9 10 11 C: 0 1 2 3 4 5 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 2 | 4 | 6 | 8 | 9 | 9 |
| 0 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 6 | 6 |

* 1. (2 points) Exercise 8.2-2: Prove that COUNTING-SORT is stable.
* “Let's say that two elements at indices i1<i2 are equal to each other. In the sorted array, they take place at indices j1+1=j2. Since the COUNTING-SORT processes the input array in reverse order, A[i2] is put in B[j2] first and then A[i1] is put in A[j2]. Since the two elements preserve their order, the algorithm is stable” (Kanev).
  1. (5 points) Exercise 8.3-2: Which of the following sorting algorithms are stable: insertion sort, merge sort, heapsort, and quicksort? Give a simple scheme that makes any sorting algorithm stable. How much additional time and space does your scheme entail?
* Stable: Insertion sort, merge sort

Not stable: Heapsort, quicksort

We can make any algorithm stable by mapping the array to an array of pairs, where the first element in each pair is the original element and the second is its index. Then we sort lexicographically. This scheme takes additional Θ(n) space.

(Kanev)

* 1. (3 points) Exercise 8.3-4: Show how to sort n integers in the range 0 to n3−1 in O(n) time.
* “We use radix sort. In this case, we have 2-digit numbers in base n. This makes RADIX-SORT to be Θ(2(n+n))= Θ(4n)= Θ(n)” (Kanev).

1. Work the following Exercises from Chapter 9 of the text:
   1. (4 points) Exercise 9.2-2: Argue that the indicator random variable Xkand the value T(max(k−1,n−k)) are independent.

* “Picking the pivot in one partitioning does not affect the probabilities of the subproblem. That is, the call to RANDOM in RANDOMIZED-PARTITION produces a result, independent from the call in the next iteration” (Kanev).
  1. (4 points) Exercise 9.23: Write an iterative version of RANDOMIZED-SELECT. (Skanev)

#include <stdlib.h>  
  
static int tmp;  
#define EXCHANGE(a, b) { tmp = a; a = b; b = tmp; }  
  
int randomized\_partition(int \*A, int p, int r);  
  
int randomized\_select(int \*A, int p, int r, int i) {  
 while (p < r - 1) {  
 int q = randomized\_partition(A, p, r);  
 int k = q - p;  
  
 if (i == k) {  
 return A[q];  
 } else if (i < k) {  
 r = q;  
 } else {  
 p = q + 1;  
 i = i - k - 1;  
 }  
 }  
  
 return A[p];  
}  
  
int partition(int \*A, int p, int r) {  
 int x, i, j;  
  
 x = A[r - 1];  
 i = p;  
  
 for (j = p; j < r - 1; j++) {  
 if (A[j] < x) {  
 EXCHANGE(A[i], A[j]);  
 i++;  
 }  
 }  
  
 EXCHANGE(A[i], A[r - 1]);  
  
 return i;  
}  
  
int randomized\_partition(int \*A, int p, int r) {  
 int pivot = rand() % (r - p) + p;  
 EXCHANGE(A[pivot], A[r - 1]);  
 return partition(A, p, r);  
}

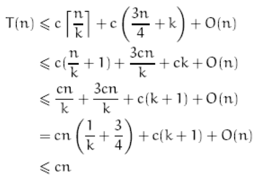
(Skanev)

* 1. (3 points) Exercise 9.3-1, page 223.

Assuming each group has k elements.

The number that are less(or greater) then median of median is at least n/4-k，In the worst case, next call to SELECT will recursive call 3n/4+k elements. So,

image

Assuming for all n, T(n) <= cn

So according to <= 1 we get k >= 4 (GZc)

Works Cited

Gzc. "C09-Medians-and-Order-Statistics." *GitHub*. N.p., n.d. Web.

Kanev, Stefan. *Introduction to Algorithms*. N.p., n.d.

Skanev. Exercise 9.2.3, ita.skanev.com/09/02/03.html.