**CPSC 6109:** [**Advanced**](https://colstate.view.usg.edu/d2l/lp/ouHome/home.d2l?ou=1218642) **Algorithms**

**Spring 2018**

**Assignment #2**

**Due: 11:59 PM Monday, Feb. 12**

**Student: Lu Lin**

**Date: 02/10/2018**

Do the following exercises/problems. Each problem is worth 20 points with a total of 100 points.

1. Using Figure 7.1 as a model, illustrate the operation of PARTITION on the array

A = <20, 5, 4, 22, 13, 2, 15, 1, 18, 7, 14, 10, 16, 3>.

Solution:

As the operation of PARTITION on an array in Figure 7.1, element 3 becomes the pivot element x. Steps are as following:

Original A = <20, 5, 4, 22, 13, 2, 15, 1, 18, 7, 14, 10, 16, 3>

1. A = <20, 5, 4, 22, 13, 2, 15, 1, 18, 7, 14, 10, 16, 3>
2. A = <20, 5, 4, 22, 13, 2, 15, 1, 18, 7, 14, 10, 16, 3>
3. A = <20, 5, 4, 22, 13, 2, 15, 1, 18, 7, 14, 10, 16, 3>
4. A = <20, 5, 4, 22, 13, 2, 15, 1, 18, 7, 14, 10, 16, 3>
5. A = <20, 5, 4, 22, 13, 2, 15, 1, 18, 7, 14, 10, 16, 3>
6. A = <2, 5, 4, 22, 13, 20, 15, 1, 18, 7, 14, 10, 16, 3>
7. A = <2, 5, 4, 22, 13, 20, 15, 1, 18, 7, 14, 10, 16, 3>
8. A = <2,1, 4, 22, 13, 20, 15, 5, 18, 7, 14, 10, 16, 3>
9. A = <2,1, 4, 22, 13, 20, 15, 5, 18, 7, 14, 10, 16, 3>
10. A = <2,1, 4, 22, 13, 20, 15, 5, 18, 7, 14, 10, 16, 3>
11. A = <2,1, 4, 22, 13, 20, 15, 5, 18, 7, 14, 10, 16, 3>
12. A = <2,1, 4, 22, 13, 20, 15, 5, 18, 7, 14, 10, 16, 3>
13. A = <2,1, 4, 22, 13, 20, 15, 5, 18, 7, 14, 10, 16, 3>
14. A = <2,1, 3, 22, 13, 20, 15, 5, 18, 7, 14, 10, 16, 4>
15. Exercise 7.2-3 on page 178

Solution:

To prove that the running time of QUICKSORT is Θ(n2) when the array A contains distinct elements and is sorted in decreasing order, I need to prove Exercise 7.2-1 first and then use its conclusion.

By definition of Θ, if we want to prove T(n) = Θ(n2) the substitution method requires to show T(n) ≤ cn2 for some constant c > 0. If T(n) ≤ cn2 is correct then for n -1 which is < n, we have T(n - 1) ≤ c(n – 1)2. Substituting into the recurrence yields

T(n) ≤ c(n – 1)2 + Θ(n) ≤ c(n – 1)2 + mn = cn2 – 2cn + c + mn ≤ cn2

Where the last step holds for n ≥ 1 and 0 ≤ m ≤ c. Thus it turns out T(n) = Θ(n2).

In this array all elements are greater than pivot. The partition step takes Θ(n) time and gives an empty partition and a subproblem of size (n-1). This yields recurrence T(n) = T(n - 1) + Θ(n). As proved previously, we get the solution the worst case running time of QUICKSORT is Θ(n2).

1. Exercise 8.2-1 on page 196. Change the array A to

A = <6, 0, 5, 4, 0, 1, 3, 2, 0, 2, 4, 1, 3, 5, 0, 5, 1, 2>.

Solution:

Let C[0..k] be a new array and for i = 0 to k C[i] = 0; for j = 1 to A.length C[A[j]] = C[A[j]] + 1.

1. Original A = <6, 0, 5, 4, 0, 1, 3, 2, 0, 2, 4, 1, 3, 5, 0, 5, 1, 2>

C[i] contains the number of elements equal to i:

C = <4, 3, 3, 2, 2, 3, 1>

1. For i = 1 to k: C[i] = C[i] + C[i-1] we got

C = <4, 7, 10, 12, 14, 17, 18>

1. For j = 18 down towards 1: B[C[A[j]]] = A[j], and C[A[j]] = C[A[j]] – 1

B = < , , , , , , , , , 2, , , , , , , , > and C = <4, 7, 9, 12, 14, 17, 18>

B = < , , , , , , 1, , , 2, , , , , , , , > and C = <4, 6, 9, 12, 14, 17, 18>

B = < , , , , , , 1, , , 2, , , , , , , 5, > and C = <4, 6, 9, 12, 14, 16, 18>

B = < , , , 0, , , 1, , , 2, , , , , , , 5, > and C = <3, 6, 9, 12, 14, 16, 18>

B = < , , , 0, , , 1, , , 2, , , , , , 5, 5, > and C = <3, 6, 9, 12, 14, 15, 18>

B = < , , , 0, , , 1, , , 2, , 3, , , , 5, 5, > and C = <3, 6, 9, 11, 14, 15, 18>

B = < , , , 0, , 1, 1, , , 2, , 3, , , , 5, 5, > and C = <3, 5, 9, 11, 14, 15, 18>

B = < , , , 0, , 1, 1, , , 2, , 3, , 4, , 5, 5, > and C = <3, 5, 9, 11, 13, 15, 18>

B = < , , , 0, , 1, 1, , 2, 2, , 3, , 4, , 5, 5, > and C = <3, 5, 8, 11, 13, 15, 18>

B = < , , 0, 0, , 1, 1, , 2, 2, , 3, , 4, , 5, 5, > and C = <2, 5, 8, 11, 13, 15, 18>

B = < , , 0, 0, , 1, 1, 2, 2, 2, , 3, , 4, , 5, 5, > and C = <2, 5, 7, 11, 13, 15, 18>

B = < , , 0, 0, , 1, 1, 2, 2, 2, 3, 3, , 4, , 5, 5, > and C = <2, 5, 7, 10, 13, 15, 18>

B = < , , 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, , 4, , 5, 5, > and C = <2, 4, 7, 10, 13, 15, 18>

B = < , 0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, , 4, , 5, 5, > and C = <1, 4, 7, 10, 13, 15, 18>

B = < , 0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 4, 4, , 5, 5, > and C = <1, 4, 7, 10, 12, 15, 18>

B = < , 0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 4, 4, 5, 5, 5, > and C = <1, 4, 7, 10, 12, 14, 18>

B = < 0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 4, 4, 5, 5, 5, > and C = <0, 4, 7, 10, 12, 14, 18>

B = < 0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 4, 4, 5, 5, 5, 6> and C = <0, 4, 7, 10, 12, 14, 17>

1. Finally B[18] = < 0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 4, 4, 5, 5, 5, 6>
2. Exercise 10.1-2 on page 235

Solution:

The first stack (s1) starts at 1 and grows up towards n, while the second (s2) starts at n and grows downwards 1. Stack overflow happens when an element is pushed when the two stack pointers are adjacent.

int n = 100;

int[] stack = new int[n];

int left = -1; // head of left stack

int right = n; // head of right stack

Pseudo-codes PUSH (int x, boolean pushLeft)

if (left + 1 == right) then

error “overflow”;

end if;

if (pushLeft) then

left++;

stack[left] = x;

else

right--;

stack[right] = x;

end if;

Pseudo-codes POP (Boolean popLeft)

if (popLeft) then

if (left == -1) then error “underflow”

rtn = stack[left];

stack[left] = 0;

left--;

return rtn;

end if;

else

if (right == n) then error “underflow”;

rtn = stack[right];

stack[right] = 0;

right++;

return rtn;

end if

end if

1. Exercise 12.1-2 on page 289.

Solution:

As for binary search tree (BST), all its(root’s) left child nodes are smaller than root and all its right child nodes are larger than root. As for min-heap, its property both left and right children are larger than root, but the child nodes are not in order. Thus the min-heap can’t be used to print out the keys in sorted order in linear time because the subtrees are not in order. In BST the INORDER-TREE-WALK can be used to print out keys in order. For min-heap, it needs to call MAX-HEAPIFY to maintain its property so it costs O() time, and BUILD-MAX-HEAP makes O(n) such calls. Thus the running time is O(n).