Assignment Six

Question 1

Wooden Block Toys color sort

1. Consider an array of “Wooden blocks toys”. One of the features of these toys is that they are painted either Blue or Red. Devise an algorithm to keep all the Blue toys together at one end of the array and all Red toys together at the other end of the array. Is your algorithm in place? If not, what is the space complexity? What is the time complexity?

Assume blue = 0, red = 1

Algorithm colorSort1(A, start, stop)

i <- start

j <- stop – 1

while (i <= j) do

while (i < stop & A[i] == blue) i++

while (j >= start & A[j] == red) j—

If (i < j)

swap(A, i, j)

i++

j—

The algorithm is in place and time complexity is O(n)

—---------—-----------------------——----------------—-----------------------—-------

b) Solve it for three different colors: Blue, Red and Green. Is your algorithm in place? If not, what is the space complexity? What is the time complexity? Remember we are more concerned about the time complexity

Assume color is quantitated as blue = 0, red = 1, green = 2

Algorithm colorSort1(A, start, stop, color, max\_color)

// if 3 color max\_color = 2

i <- start

j <- stop – 1

while (i <= j) do

while (i < stop & A[i] == color) i++

while (j >= start & A[j] != color) j—

If (i < j)

swap(A, i, j)

i++

j—

if(color < max\_color)

colorSort1(A, i, stop, color + 1, max)

The algorithm is in place and time complexity is O(2n)

—---------—-----------------------——----------------—-----------------------—-------

c) Solve it for four different colors: Blue, Red, Green and Yellow. Is your algorithm in place? If not, what is the space complexity? What is the time complexity? Remember we are more concerned about the time complexity.

Assume color is quantitated as blue = 0, red = 1, green = 2, yellow = 3

Algorithm colorSort1(A, start, stop, color, max\_color)

// if 3 color max\_color = 3

i <- start

j <- stop – 1

while (i <= j) do

while (i < stop & A[i] == color) i++

while (j >= start & A[j] != color) j—

If (i < j)

swap(A, i, j)

i++

j—

if(color < max\_color)

colorSort1(A, i, stop, color + 1, max)

The algorithm is in place and time complexity is O(2n)

—---------—-----------------------——----------------—-----------------------—-------

Question 2

1. {1,2,3,4,5,6,7,8,9}

1 2 3 4 5 6 7 8 9 pivot is 5, swap

1 2 3 4 9 6 7 8 5

j i swap(A[i],pivot)

1 2 3 4 5 6 7 8 9

1 3 4 2 5 6 8 9 7 pivot is 2 in L,7 in G

, j i , j i

1 2 3 4 5 6 7 8 9 swap(A[i],pivot) on both

—---------—-----------------------——----------------—-----------------------—-------

1. {8,7,6,5,4,3,2,1,9}

8 7 6 5 9 3 2 1 4 pivot is 4

swap

1 7 6 5 9 3 2 8 4

swap

1 2 6 5 9 3 7 8 4

1 2 6 5 9 3 7 8 4

swap

1 2 3 5 9 6 7 8 4

1 2 3 5 9 6 7 8 4

1 2 3 5 9 6 7 8 4

swap (A[i], pivot)

1 2 3 4 9 6 7 8 5

1 3 2 [4] 9 6 5 8 7

1 3 2 [4] 9 6 5 8 7

swap swap (A[i], A[j])

1 [2] 3 [4] 5 6 9 8 7

1 [2] 3 [4] 5 6 9 8 7

swap(A[i], pivot)

[1] [2] [3] [4] 6 5 [7] 9 8

1 2 3 4 5 6 7 8 9

—---------—-----------------------——----------------—-----------------------—-------

1. {9,1,8,2,7,3,6,4,5}

9 1 8 2 5 3 6 4 7

swap (A[i], A[j])

4 1 8 2 5 3 6 9 7

swap (A[i], A[j])

4 1 6 2 5 3 8 9 7

4 1 6 2 5 3 [7] 9 8

3 1 6 2 5 4 [7] 8 9

swap

3 1 2 6 5 4 [7] [8] [9]

swap (A[i], pivot)

3 1 2 [4] 5 6 [7] [8] [9]

3 1 2 4 5 6 [7] [8] [9]

swap(A[i], A(j])

1 3 2 4 5 6 [7] [8] [9]

swap(A[i], A[j])

1 2 3 4 5 6 7 8 9

—---------—-----------------------——----------------—-----------------------—-------

1. {5,1,4,2,3,9,7,6,8}

8 1 4 2 3 9 7 6 5

8 1 4 2 3 9 7 6 5

swap (A[i], A[j])

3 1 4 2 8 9 7 6 5

3 1 4 2 8 9 7 6 5

swap (A[i], A[j])

3 1 4 2 [5] 9 7 6 8

swap (A[i], A[j])

3 1 4 2 [5] 6 7 9 8

swap

1 3 4 2 [5] 6 7 9 8

swap swap(A[i], A[j])

1 2 4 3 [5] 6 7 8 9

1 2 3 4 [5] 7 6 8 9

swap(A[i], A[j])

1 2 3 4 [5] 6 7 8 9

—---------—-----------------------——----------------—-----------------------—-------

Question 3

1. {1,2,3,4,5,6,7,8,9} k=5

1 2 3 4 5 6 7 8 9 pivot is 5, swap

1 2 3 4 9 6 7 8 5

j i swap(A[i],pivot)

1 2 3 4 5 6 7 8 9

1 3 4 2 5 6 8 9 7 pivot is 2 in L,7 in G

, j i , j i

1 2 3 4 5 6 7 8 9 swap(A[i],pivot) on both

The 5th element is 5.

—---------—-----------------------——----------------—-----------------------—-------

1. {8,7,6,5,4,3,2,1,9} k=3

8 7 6 5 9 3 2 1 4 pivot is 4

swap

1 7 6 5 9 3 2 8 4

swap

1 2 6 5 9 3 7 8 4

1 2 6 5 9 3 7 8 4

swap

1 2 3 5 9 6 7 8 4

1 2 3 5 9 6 7 8 4

1 2 3 5 9 6 7 8 4

swap (A[i], pivot)

1 2 3 4 9 6 7 8 5

1 3 2 [4] 9 6 5 8 7

1 3 2 [4] 9 6 5 8 7

swap swap (A[i], A[j])

1 [2] 3 [4] 5 6 9 8 7

1 [2] 3 [4] 5 6 9 8 7

swap(A[i], pivot)

[1] [2] [3] [4] 6 5 [7] 9 8

1 2 3 4 5 6 7 8 9

The 3rd element is 3.

—---------—-----------------------——----------------—-----------------------—-------

1. {9,1,8,2,7,3,6,4,5} k=8

9 1 8 2 5 3 6 4 7

swap (A[i], A[j])

4 1 8 2 5 3 6 9 7

swap (A[i], A[j])

4 1 6 2 5 3 8 9 7

4 1 6 2 5 3 [7] 9 8

3 1 6 2 5 4 [7] 8 9

swap

3 1 2 6 5 4 [7] [8] [9]

swap (A[i], pivot)

3 1 2 [4] 5 6 [7] [8] [9]

3 1 2 4 5 6 [7] [8] [9]

swap(A[i], A(j])

1 3 2 4 5 6 [7] [8] [9]

swap(A[i], A[j])

1 2 3 4 5 6 7 8 9

The 8th element is 8.

—---------—-----------------------——----------------—-----------------------—-------

1. {5,1,4,2,3,9,7,6,8} k=5

8 1 4 2 3 9 7 6 5

8 1 4 2 3 9 7 6 5

swap (A[i], A[j])

3 1 4 2 8 9 7 6 5

3 1 4 2 8 9 7 6 5

swap (A[i], A[j])

3 1 4 2 [5] 9 7 6 8

swap (A[i], A[j])

3 1 4 2 [5] 6 7 9 8

swap

1 3 4 2 [5] 6 7 9 8

swap swap(A[i], A[j])

1 2 4 3 [5] 6 7 8 9

1 2 3 4 [5] 7 6 8 9

swap(A[i], A[j])

1 2 3 4 [5] 6 7 8 9

[k=1, k=4] k=5 [k=6 up to k=9]

The 5th element is 5.

—---------—-----------------------——----------------—-----------------------—-------

Question 4

(a)

This length is

At each level of the recursion tree, total processing time is . Therefore, the total running time is the good case is

(b) Yes, we are able to derive the same results in Slides 16 and 17.