CPSC 335 Fall 2019 MW 2:30-3:45 pm Project 1 Report

Carla Jacobsen cjacobsen2016@csu.fullerton.edu

Left-To-Right Algorithm:

```
def sort_left_to_right(before)
{//sort_left_to_right

        check disks for alternating

        numSwaps = 0

        newDisks = before

        while the disks are being sorted do
                {//while
                for i = 0 to n -1 do
                        {//for
                        //if current disk is light and the next is dark then swap
                        if (disk[i] == light) && (disk[i+1] == dark) do
                                {//if
                                ++numSwaps

                                swap disk[i] and disk[i+1]
                                }//if
                        }//for
                }//while

        return a new disk object
}//sort_left_to_right
```

<u>Proof for Left-to-Right</u>:

sc of check disks for alternating = n
sc of numSwaps = 1
sc of newDisks = 1
sc of return = 1

//if statement block:
sc of swap = 1
sc of ++numSwaps = 1
sc of evaluating if condition = 2
sc of then branch = 1 + 1 = 2
sc of else branch = 0
sc of if block = sc of evaluating if condition + max(2,0) = 2 + 2 = 4

//for loop:
sc inside for loop = sc of if block = 4
sc of for loop duration = (n – 1) – 0 + 1 = n
sc of for loop block = (sc of for loop duration) * (sc inside for loop) = 4n

//while loop:
sc of while loop duration = n
sc of while loop block = (while loop duration) * (for loop block) = 4(n^2)

//the entire function:
sc of left_to_right =
(sc while block) + (sc check disks) + (sc numSwaps) + (sc newDisks) + (sc return) =
4(n^2) + n + 1 + 1 + 1 = 4(n^2) + n + 3 =
O(n^2)

Lawnmower Algorithm:

```
def sort_lawnmower(before)
{//sort_lawnmower

        check disks for alternating

        numSwaps = 0

        newDisks = before

        while the disks are being sorted do
                {//while
                for i = 0 to n - 1 do     //block a
                        {//for
                        //if current disk is light and the next is dark then swap
                        if (disk[i] == light) && (disk[i+1] == dark) do        //block c
                                {//if
                                ++numSwaps

                                swap disk[i] and disk[i+1]
                                }//if
                        }//for
                for i = n - 1 to 1 do     //block b
                        {//for
                        //if current disk is dark and the left is light then swap
                        if(disk[i] == dark) && (disk[i-1] == light) do         //block d
                                {//if
                                ++numSwaps

                                swap disk[i] and disk[i-1]
                                }//if
                        }//for
                }//while

        return a new disk object
}//sort_lawnmower
```

<u>Proof for Lawnmower</u>:

sc of check disks for alternating = n
sc of numSwaps = 1
sc of newDisks = 1
sc of return = 1

//if statement block c:
sc of swap = 1
sc of ++numSwaps = 1
sc of evaluating if condition = 2
sc of then branch = 1 + 1 = 2
sc of else branch = 0
sc of if block c = sc of evaluating if condition + max(2,0) = 2 + 2 = 4

//if statement block d:
sc of swap = 1
sc of ++numSwaps = 1
sc of evaluating if condition = 2
sc of then branch = 1 + 1 = 2
sc of else branch = 0
sc of if block d = sc of evaluating if condition + max(2,0) = 2 + 2 = 4

//for loop block a:
sc inside for loop = sc of if block c = 4
sc of for loop duration = (n – 1) – 0 + 1 = n
sc of for loop block a = (sc of for loop duration) * (sc inside for loop) = 4n

//for loop block b:
sc inside for loop = sc of if block d = 4
sc of for loop duration = (n – 1) - 1 + 1 = n - 1
sc of for loop block b = (sc of for loop duration) * (sc inside for loop) = 4(n - 1) = 4n - 4

//while loop:
sc of inside while loop = (sc of for loop a) + (sc of for loop b) = 4n + 4n – 4 = 8n – 4
sc of while loop duration = n / 2
sc of while loop block = (sc of while loop duration) * (sc of inside while loop) =
(n / 2) * (8n – 4) = 4(n^2) – 2n

//the entire function:
sc of lawnmower =
(sc while block) + (sc check disks) + (sc numSwaps) + (sc newDisks) + (sc return) =
4(n^2) – 2n + n +1 +1 +1 = 4(n^2) – n + 3 =
O(n^2)

Screenshot:

```
carla@scotchmallow: ~/Desktop/cs335 proj1
File  Edit  View  Search  Terminal  Help
carla@scotchmallow:~/Desktop/cs335 proj1$ sh proj1.sh
rm: cannot remove '*.o,': No such file or directory
rm: cannot remove '*.lis,': No such file or directory
rm: cannot remove '*.out': No such file or directory
Compile the C++ file disks_test.cpp
run the executable
disk_state still works: passed, score 1/1
sorted_disks still works: passed, score 1/1
disk_state::is_alternating: passed, score 3/3
disk_state::is_sorted: passed, score 3/3
left-to-right, n=4: passed, score 1/1
left-to-right, n=3: passed, score 1/1
left-to-right, other values: passed, score 1/1
lawnmower, n=4: passed, score 1/1
lawnmower, n=3: passed, score 1/1
lawnmower, other values: passed, score 1/1
TOTAL SCORE = 14 / 14

carla@scotchmallow:~/Desktop/cs335 proj1$
```