

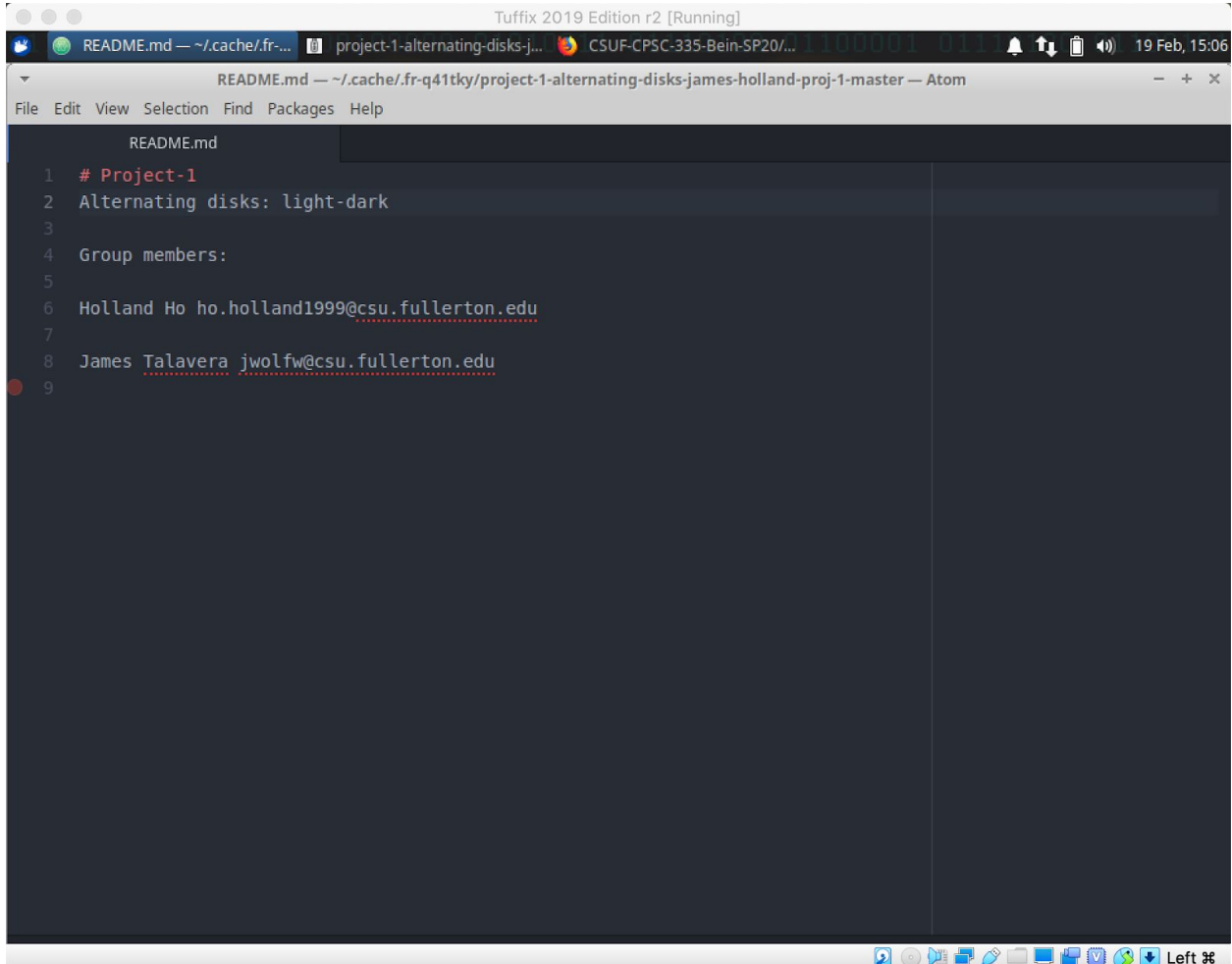
Project 1 Brief Written Report

Group Member Names:

Holland Ho ho.holland1999@csu.fullerton.edu

James Talavera jwolfw@csu.fullerton.edu

Screenshot:



```
README.md
1 # Project-1
2 Alternating disks: light-dark
3
4 Group members:
5
6 Holland Ho ho.holland1999@csu.fullerton.edu
7
8 James Talavera jwolfw@csu.fullerton.edu
9
```

Pseudocode listings:

1. sort_left_to_right algorithm:

```
# sort_left_to_right algorithm pseudocode

def sort_left_to_right(disk_state before):
    assert(before.is_alternating())

    disk_state temp = before
    swap_count = 0

    for i in range(temp.light_count()):
        for j in range(temp.total_count - 1):
            if temp.get(j) is DISK_DARK and temp.get(j+1) is DISK_LIGHT:
                temp.swap(j)
                swap_count += 1

    return sorted_disks(temp, swap_count)
```

2. sort_lawnmower algorithm:

```
# sort_lawnmower pseudocode

def sort_lawnmower(disk_state before):
    assert(before.is_alternating())

    disk_state after = before
    swap_count = 0

    for i in range(after.light_count()/2):
        for j in range(after.total_count()-1):
            if after.get(j) is DISK_DARK and after.get(j+1) is DISK_LIGHT:
                after.swap(j)
                swap_count +=1

        for k in range(after.total_count()-2,0,-1):
            if after.get(k) is DISK_DARK and after.get(k+1) is DISK_LIGHT:
                after.swap(k)
                swap_count += 1

    return sorted_disks(after, swap_count)
```

Brief Proof Argument for time complexity:

Proof of Step Count for `is_alternating()`:

```
def is_alternating():
```

```
    if _colors[0] is DISK_DARK:  $\leftarrow 2tu$ 
```

```
        return false  $\leftarrow 1tu$ 
```

$$\left. \begin{array}{l} 2 + \max(1, 0) \\ = 2 + 1 = 3 \end{array} \right\}$$

```
    for i in range(_colors.size()-1):  $\leftarrow 2n$ 
```

```
        if _colors[i] is DISK_LIGHT and _colors[i+1] is DISK_LIGHT:  $\leftarrow 6tu$ 
```

```
            return false  $\leftarrow 1tu$ 
```

$$\left. \begin{array}{l} 2n(7+7) \\ 2n(14) \\ = 28n \end{array} \right\}$$

$$\left. \begin{array}{l} 6 + \max(1, 0) \\ = 6 + 1 = 7 \end{array} \right\}$$

```
        else if _colors[i] is DISK_DARK and _colors[i+1] is DISK_DARK:  $\leftarrow 6tu$ 
```

```
            return false  $\leftarrow 1tu$ 
```

$$\left. \begin{array}{l} 6 + \max(1, 0) \\ 6 + 1 = 7 \end{array} \right\}$$

```
    return true  $\leftarrow 1tu$ 
```

total SC:

$$3 + 1 + 28n = 28n + 4$$

Left to Right Algorithm Time Complexity Proof:

def sort_left_to_right(disk_state before):

assert(before.is_alternating()) $\leftarrow 28n+4$

disk_state temp = before $\leftarrow 1tn$

swap_count = 0 $\leftarrow 1tn$

for i in range(temp.light_count()): $\leftarrow n-1$

for j in range(temp.total_count - 1): $\leftarrow 2n-1$

if temp.get(j) is DISK_DARK and temp.get(j+1) is DISK_LIGHT: $\leftarrow 6tn$

temp.swap(j) $\leftarrow 1tn$

swap_count += 1 $\leftarrow 1tn$

return sorted_disks(temp, swap_count) $\leftarrow 1tn$

$$\begin{aligned} n-1(16n-8) \\ = 16n^2 - 8n + 16n - 8 \\ = 16n^2 - 24n + 8 \end{aligned}$$

$$2n-1(8) = 16n-8$$

$$\begin{aligned} 6 + \max(2, 0) \\ = 8 \end{aligned}$$

total sc:

$$28n+6 + 16n^2 - 24n + 8 + 1$$

$$= 16n^2 + 4n + 15$$

Proof:

$$\lim_{n \rightarrow \infty} \frac{16n^2 + 4n + 15}{n^2} \in \mathcal{O}(n^2)$$

$$16 \geq 0$$

therefore the algorithm is $\mathcal{O}(n^2)$

Lawnmower Algorithm Time Complexity Proof:

```
def sort_lawnmower (disk_state before):
```

```
    assert(before.is_alternating())  $\leftarrow 28n+4$ 
```

```
    disk_state after = before  $\leftarrow 1n$ 
```

```
    swap_count = 0  $\leftarrow 1n$ 
```

```
    for i in range(after.light_count()/2):  $\leftarrow \frac{n}{2}$ 
```

```
        for j in range(after.total_count()-1):  $\leftarrow 2n-1$ 
```

```
            if after.get(j) is DISK_DARK and after.get(j+1) is DISK_LIGHT:  $\leftarrow 6n$ 
```

```
                after.swap(j)  $\leftarrow 1n$ 
```

```
                swap_count += 1  $\leftarrow 1n$ 
```

```
    for k in range(after.total_count()-2, 0, -1):  $\leftarrow 2n-2$ 
```

```
        if after.get(k) is DISK_DARK and after.get(k+1) is DISK_LIGHT:  $\leftarrow 6n$ 
```

```
            after.swap(k)  $\leftarrow 1n$ 
```

```
            swap_count += 1  $\leftarrow 1n$ 
```

```
    return sorted_disks(after, swap_count)  $\leftarrow 1n$ 
```

$$\frac{n}{2} (32n - 24)$$

$$16n^2 - 12n$$

$$2n - 1(8) = 16n - 8$$

$$6 \times \max(2n, 0) = 8$$

$$6 \times \max(2n, 0) = 8$$

$$2n - 2(8) = 16n - 16$$

$$16n - 8 + 16n - 16 = 32n - 24$$

total sc:

$$28n + 6 + 16n^2 - 12n + 7$$

$$= 16n^2 + 16n + 7$$

Proof:

$$\lim_{n \rightarrow \infty} \frac{16n^2 + 16n + 7}{n^2} \in O(n^2)$$

$$16 \geq 0$$

therefore the algorithm is $O(n^2)$