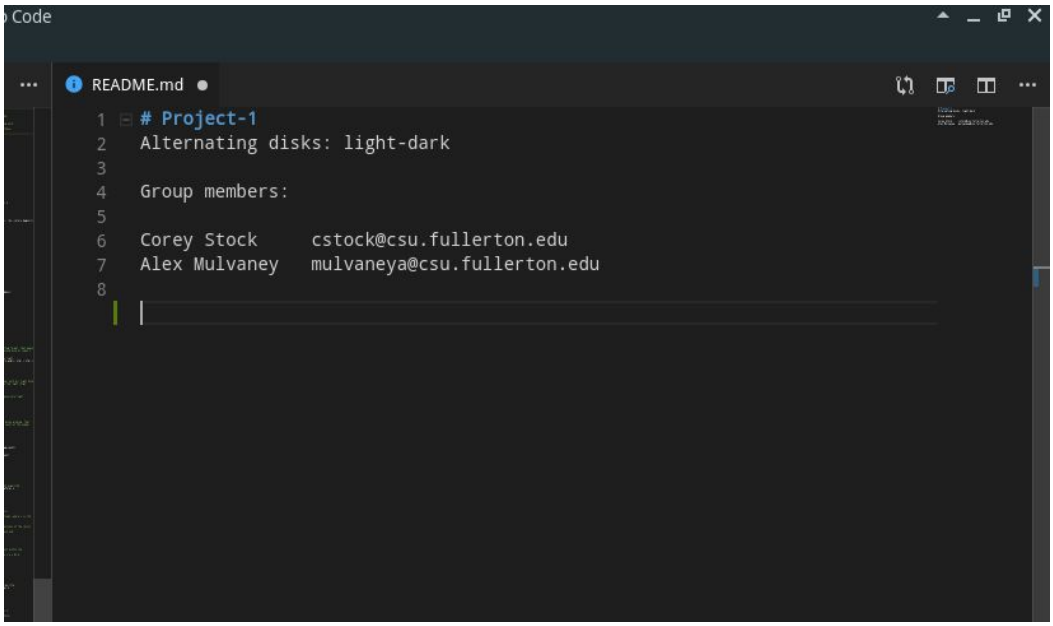
CPSC 335 Project 1
Alex Mulvaney - mulvaneya@csu.fullerton.edu
Corey Stock - cstock@csu.fullerton.edu



PSEUDOCODE:
(1)

```
Sorted_disks sort_left_to_right(disk_state before)
        If (before.total_count()) == 2
                Return sorted_disks(before, 0) // only 2 disks no swaps
        disk_state after = disk_stake(before)
        max_iterations = after.light_count() //max iterations is equal to the # of light disks
        //set sorted portions as the first and last element
        left_sorted = 1
        right_sorted = after.total_count() - 1

        for( i = 0; i < max_iterations; i++)
                //disk swaps until reaching end of sorted portion of the vector
                for( j = left_sorted; j < right_sorted; j+=2)
                        after.swap(j)
                        swap_count++
                //close in the sorted sections
                left_sorted++
                right_sorted--
        Return sorted_disks(after, swap_count)
```

(2)

```
sorted_disks sort_lawnmower( disk_state before )
        If (before.total_count()) == 2)
                Return sorted_disks(before, 0) // only 2 disks no swaps
        If (before.total_count()) == 4)
                disk_state after = disk_stake(before)
                after.swap(1)
                Return sorted_disks(before, 0) // only 4 disks 1 swaps

        disk_state after = disk_stake(before)
        //max iterations is equal to half the # of light disks with 1 iteration being a
          sweep right then left
        max_iterations = after.light_count()/2
        //set sorted portions as the first and last element
        left_sorted = 1
        right_sorted = after.total_count() - 1

        for(i = 0; i < max_iterations; i++)
                //disk swaps until reaching end of sorted portion of the vector
                for( j = left_sorted; j < right_sorted; j+=2)
                        after.swap(j)
                        swap_count++
                //close in the sorted sections
                left_sorted++
                right_sorted--
                //reverse the previous loop
                for(k = right_sorted; k >= left_sorted; k = k - 2)
                        after.swap(k)
                        swap_count++
                //close in the sorted sections
                left_sorted++
                right_sorted--
        return sorted_disks(after, swap_count)
```

Mathematical Analysis:

(1)

```
Sorted_disks sort_left_to_right(disk_state before)
        If (before.total_count()) == 2                          -(1)
                Return sorted_disks(before, 0)                  -(1)
        disk_state after = disk_stake(before)                   -(1)
        max_iterations = after.light_count()                    -(1)
        left_sorted = 1                                         -(1)
        right_sorted = after.total_count() - 1                  -(2)

        for( i = 0; i < max_iterations; i++)                    -(n)

                for( j = left_sorted; j < right_sorted; j+=2)   -(n/2)
                        after.swap(j)                           -(1)
                        swap_count++                            -(1)

                left_sorted++                                   -(1)
                right_sorted--                                  -(1)
        return sorted_disks(after, swap_count)                  -(1)
```

Step-Count: 7 + (n * (n/2) * 2 * 2) + 1

: $((4n^2)/2) + 8$

$$\lim_{n\to\infty} \frac{((4n^2)/2) + 8}{n^2} = 4;$$ **4 $\geq$ 0, so $((4n^2)/2) + 8 \in O(n^2)$**

(2)

```
sorted_disks sort_lawnmower( disk_state before )
        If (before.total_count()) == 2                          1 + max(1, 0) = 2
                Return sorted_disks(before, 0)
        If (before.total_count()) == 4                          1 + max(4, 0) = 5
                disk_state after = disk_stake(before)
                after.swap(1)
                Return sorted_disks(before, 0)

        disk_state after = disk_stake(before)                   2
        max_iterations = after.light_count()/2                  2
        left_sorted = 1                                         1
```

```
        right_sorted = after.total_count() - 1                  2
                                                                For = (n/4) x (((n/2) x 2) + 2
+ ((n/2) x 2) + 2))
        for(i = 0; i < max_iterations; i++)                     n/4
                for( j = left_sorted; j < right_sorted; j+=2)   n/2
                        after.swap(j)                           1
                        Swap_count++                            1
                Left_sorted++                                   1
                Right_sorted--                                  1
                for(k = right_sorted; k >= left_sorted; k = k - 2)  n/2
                        after.swap(k)                           1
                        Swap_count++                            1
                Left_sorted++                                   1
                Right_sorted--                                  1
        return sorted_disks(after, swap_count)                  1
```

$$f(n) = max(1 + max(1,0),\ 1 + max(4,0),\ 8 + \tfrac{n}{4}((\tfrac{n}{2} \cdot 2) + 2 + (\tfrac{n}{2} \cdot 2) + 2) + 1)$$
$$f(n) = 8 + \tfrac{n}{4}((\tfrac{n}{2} \cdot 2) + 2 + (\tfrac{n}{2} \cdot 2) + 2) + 1 = \tfrac{1}{4}n^2 + \tfrac{1}{2}n + \tfrac{1}{4}n^2 + \tfrac{1}{2}n + 9 = \tfrac{1}{2}n^2 + n + 9$$

$$Show\ that: \tfrac{1}{2}n^2 + n + 9 = O(n^2)$$
$$\tfrac{1}{2}n^2 + n + 9 \le c \cdot n^2,\ n > n_0$$
$$Choose\ n_0 = 1\ and\ c = 11$$
$$\tfrac{1}{2}n^2 + n + 9 \le 11 \cdot n^2,\ n > 1$$
$$\tfrac{1}{2}(1)^2 + (1) + 9 \le 11(1)^2$$
$$\tfrac{1}{2} + 10 \le 11$$
$$\tfrac{1}{2} + 10 \le 11\ is\ true,\ so\ our\ function\ is\ O(n^2)$$