

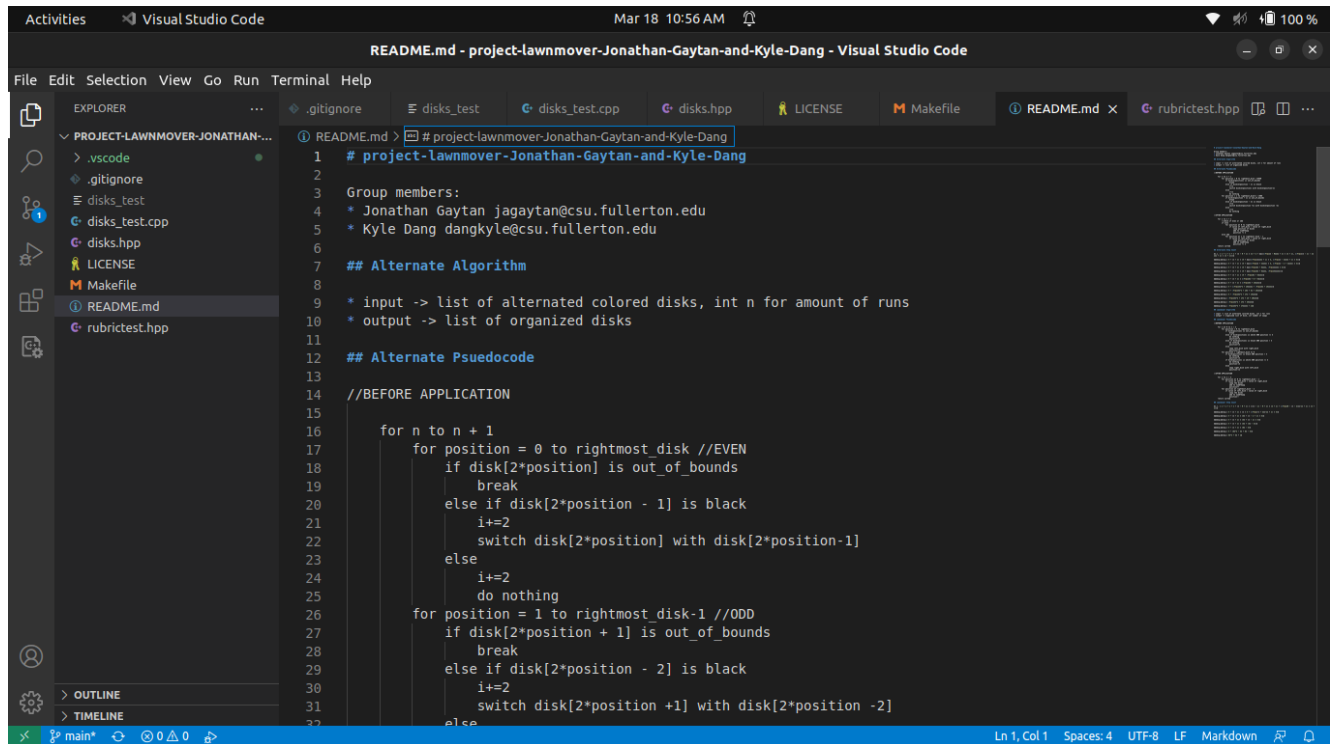
Submission for Project 1

Group members:

Jonathan Gaytan jagaytan@csu.fullerton.edu

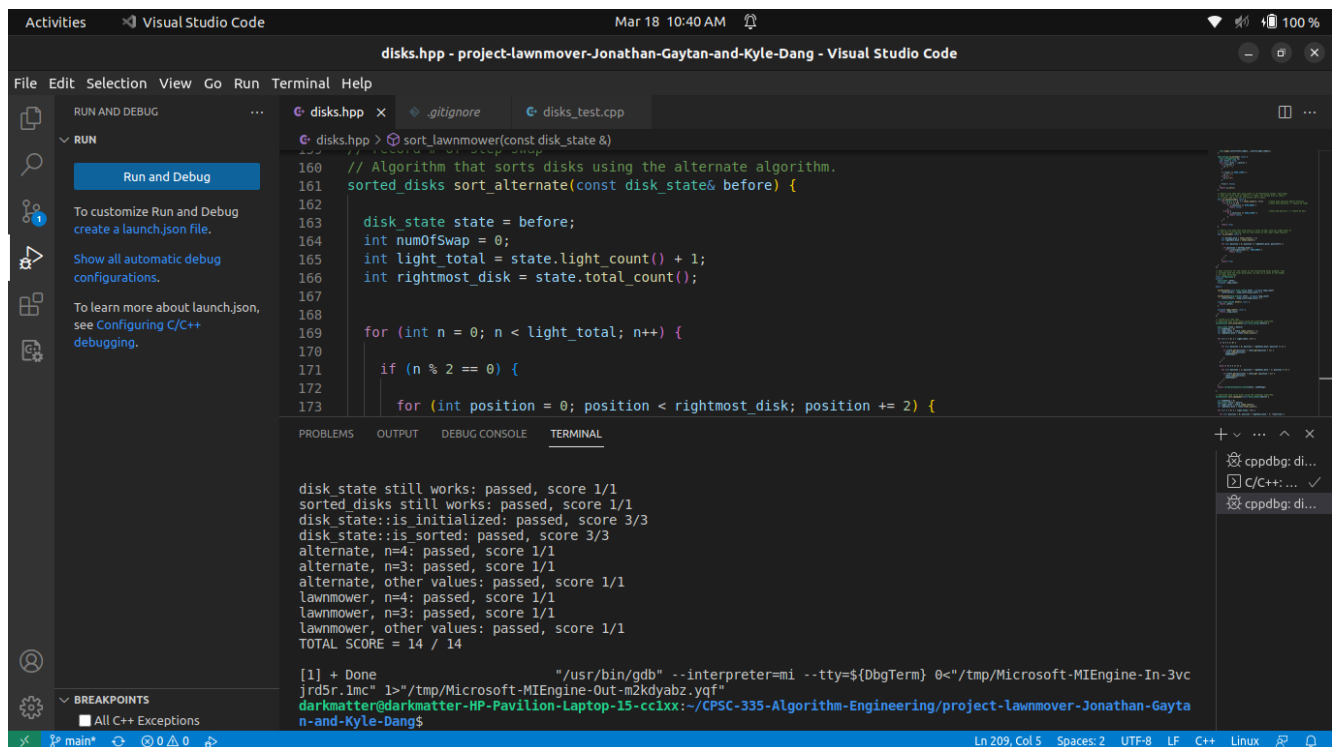
Kyle Dang dangkyle@csu.fullerton.edu

Screenshots of Editor and Code Compiling and Executing:



Visual Studio Code interface showing the README.md file for the project-lawnmower-Jonathan-Gaytan-and-Kyle-Dang. The file content is as follows:

```
1 # project-lawnmower-Jonathan-Gaytan-and-Kyle-Dang
2
3 Group members:
4 * Jonathan Gaytan jagaytan@csu.fullerton.edu
5 * Kyle Dang dangkyle@csu.fullerton.edu
6
7 ## Alternate Algorithm
8
9 * input -> list of alternated colored disks, int n for amount of runs
10 * output -> list of organized disks
11
12 ## Alternate Psuedocode
13
14 //BEFORE APPLICATION
15
16 for n to n + 1
17     for position = 0 to rightmost disk //EVEN
18         if disk[2*position] is out_of_bounds
19             break
20         else if disk[2*position - 1] is black
21             i+=2
22             switch disk[2*position] with disk[2*position-1]
23         else
24             i+=2
25             do nothing
26     for position = 1 to rightmost disk-1 //ODD
27         if disk[2*position + 1] is out_of_bounds
28             break
29         else if disk[2*position - 2] is black
30             i+=2
31             switch disk[2*position +1] with disk[2*position -2]
32         else
```



Visual Studio Code interface showing the disks.hpp file for the project-lawnmower-Jonathan-Gaytan-and-Kyle-Dang. The file content is as follows:

```
160 // Algorithm that sorts disks using the alternate algorithm.
161 sorted_disks sort_alternate(const disk_state& before) {
162
163     disk state state = before;
164     int numOfSwap = 0;
165     int light_total = state.light_count() + 1;
166     int rightmost_disk = state.total_count();
167
168     for (int n = 0; n < light_total; n++) {
169
170         if (n % 2 == 0) {
171             for (int position = 0; position < rightmost_disk; position += 2) {
```

The terminal output shows the results of the compilation and execution:

```
disk state still works: passed, score 1/1
sorted disks still works: passed, score 1/1
disk state::is initialized: passed, score 3/3
disk state::is sorted: passed, score 3/3
alternate, n=4: passed, score 1/1
alternate, n=3: passed, score 1/1
alternate, 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
```

Alternate Algorithm:

Alternate Psuedocode

```
for n to n + 1

    //check if even or odd
    if even

        for position at 0 to rightmost_disk

            if value of left_disk > value of right_disk
                swap position of disks
                add to numOfSwap
                position += 2

    else odd

        for position at 1 to rightmost disk - 1

            if value of left_disk > value of right_disk
                swap position of disks
                add to numOfSwap
                position += 2

return sorted
```

Alternate Step Count

$$\begin{aligned} s.c. &= 1 + 1 + 2 + 1 + (n - 0 + 1) * (2 + 2 + \max((\frac{n-0}{2} + 1) * (2 + 1), (\frac{(n-1)-1}{2} + 1) * (2 + 1))) \\ &= 5 + (n + 1) * (4 + \max((\frac{n}{2} + 1) * 3, (\frac{n-2}{2} + 1) * 3)) \\ &= 5 + (n + 1) * (4 + \max((\frac{n+2}{2}) * 3, (\frac{n-2+2}{2}) * 3)) \\ &= 5 + (n + 1) * (4 + \max(\frac{3n+6}{2}, \frac{n}{2} * 3)) \\ &= 5 + (n + 1) * (4 + \max(\frac{3n+6}{2}, \frac{3n}{2})) \\ &= 5 + (n + 1) * (4 + \frac{3n+6}{2}) \\ &= 5 + (n + 1) * (\frac{3n+6+8}{2}) \\ &= 5 + (n + 1) * (\frac{3n+14}{2}) \\ &= 5 + (\frac{3n^2+14n}{2} + \frac{3n+14}{2}) \\ &= 5 + \frac{3n^2+14n+3n+14}{2} \\ &= 5 + \frac{3n^2+17n+14}{2} \\ &= \frac{3n^2+17n+14+10}{2} \\ &= \frac{3n^2+17n+24}{2} \\ &= \frac{3n^2+17n}{2} + 12 \end{aligned}$$

Alternate Algorithm Efficiency Class with Limit Theorem

$$\lim_{n \rightarrow \infty} \frac{\frac{3}{2}n^2 + \frac{17}{2}n + 12}{n^2} = \frac{3}{2}$$

Due to $\frac{3}{2} \geq 0$ and $\frac{3}{2}$ being a constant, the Limit Theorem states that $\frac{3n^2+17n}{2} + 12 \in O(n^2)$

That means this algorithm has a time complexity of $O(n^2)$

Lawnmower Algorithm:

Lawnmower Psuedocode

```
for n to n / 2

    for position at 0 to rightmost_disk - 1

        if value of left_disk > value of right_disk
            swap the disks
            add to numOfSwap
            position++

    for position at rightmost_disk - 2

        if value of left_disk > value of right_disk
            swap the disks
            add to numOfSwap
            position--

return sorted
```

Lawnmower Step Count

$$\begin{aligned}s.c. &= 1 + 1 + 1 + 1 + (n - 0 + 1) * (((n - 1) - 0 + 1) * (2 + 1) + (\frac{0 - (n - 2)}{-1} + 1) * (2 + 1)) \\&= 4 + (n + 1) * (n * 3 + (\frac{-n + 2}{-1} + 1) * 3) \\&= 4 + (n + 1) * (3n + (n - 2 + 1) * 3) \\&= 4 + (n + 1) * (3n + (n - 1) * 3) \\&= 4 + (n + 1) * (3n + (3n - 3)) \\&= 4 + (n + 1) * (6n - 3) \\&= 4 + (6n^2 - 3n + 6n - 3) \\&= 6n^2 + 3n + 1\end{aligned}$$

Lawnmower Algorithm Efficiency Class with Limit Theorem

$$\lim_{n \rightarrow \infty} \frac{6n^2 + 3n + 1}{n^2} = 6$$

Due to $6 \geq 0$ and 6 being a constant, the Limit Theorem states that $6n^2 + 3n + 1 \in O(n^2)$

That means this algorithm has a time complexity of $O(n^2)$