

Project 1

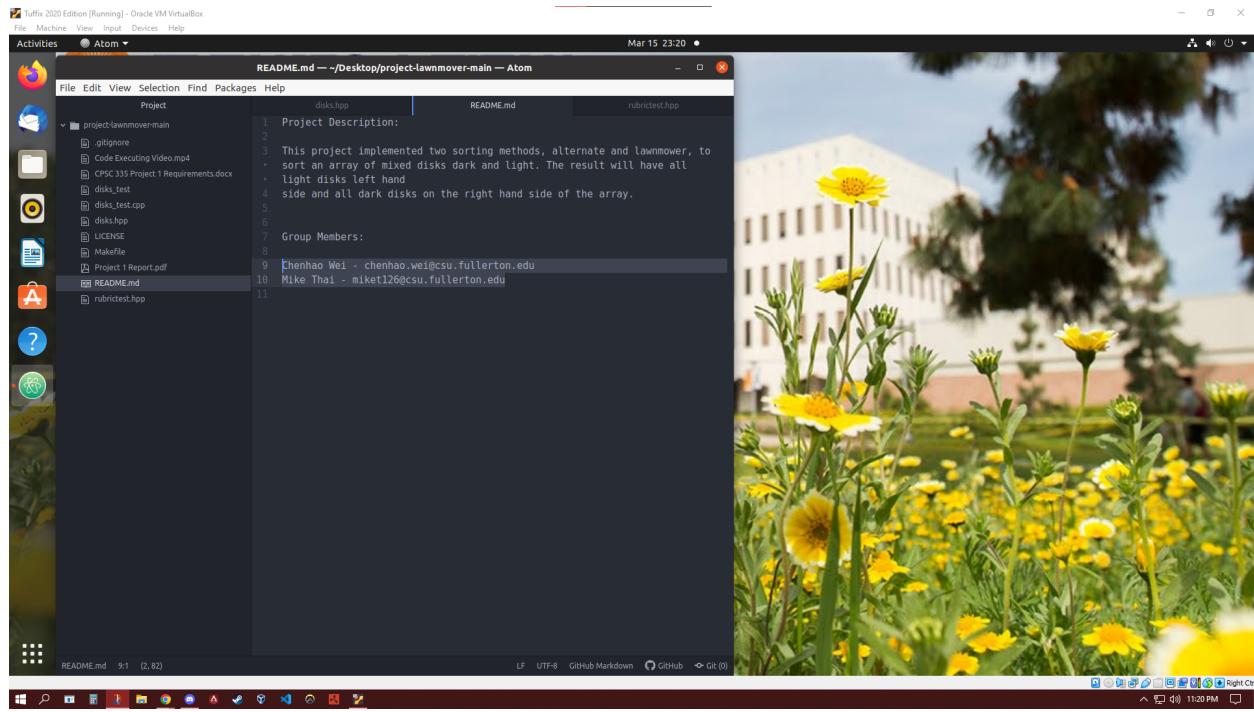
Mike Thai - miket126@csu.fullerton.edu

Chenhao Wei - chenhao.wei@csu.fullerton.edu

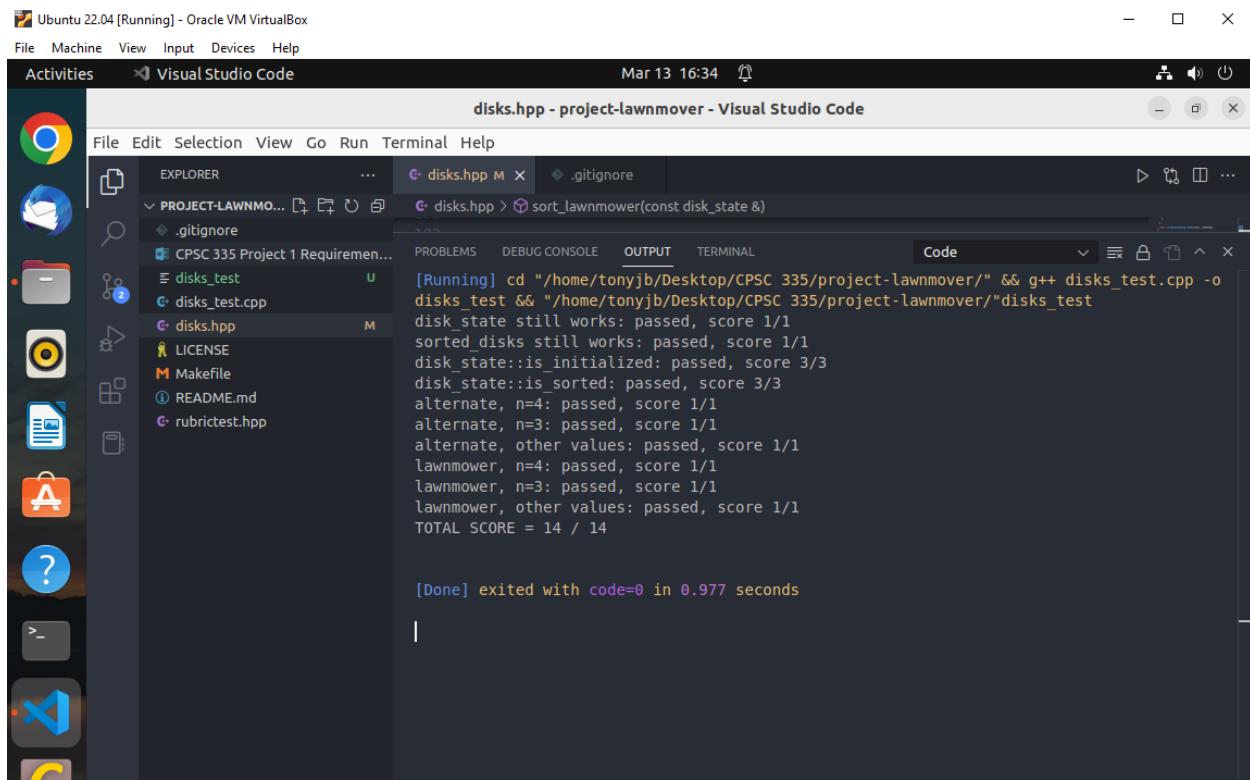
Due: March 19, 2023

README Screenshot

```
Project Description:  
This project implemented two sorting methods, alternate and lawnmower, to sort an array of mixed disks dark and light. The result will have all light disks left hand side and all dark disks on the right hand side of the array.  
Group Members:  
Chenhao Wei - chenhao.wei@csu.fullerton.edu  
Mike Thai - miket126@csu.fullerton.edu
```



Code Compiling and Executing Screenshot



Lawnmower Algorithm

The screenshot shows a code editor window with the file `disks.hpp` open. The code implements the Lawnmower algorithm for sorting disk states. The algorithm iterates through the disks, performing swaps between adjacent elements based on their values (DISK_DARK or DISK_LIGHT) until the array is sorted. The code includes comments explaining the logic and a call to a helper function `sorted_disks`.

```
183 // Algorithm that sorts disks using the lawnmower algorithm.
184 sorted_disks sort_lawnmower(const disk_state& before) {
185     int numOfSwap = 0;
186     disk_state state = before;
187     size_t size = state.total_count();
188
189     for (size_t n = 0; n < size / 2; n++) {
190         //Go Left
191         for (size_t left = 0; left < size - 1; left++) {
192             if (state.get(left) == DISK_DARK && state.get(left+1) == DISK_LIGHT) {
193                 state.swap(left);
194                 numOfSwap++;
195             }
196         }
197         //Go Right
198         for (size_t right = size - 1; right > 0; right--) {
199             if (state.get(right) == DISK_LIGHT && state.get(right-1) == DISK_DARK) {
200                 state.swap(right-1);
201                 numOfSwap++;
202             }
203         }
204     }
205
206     return sorted_disks(disk_state(state), numOfSwap);
207 }
208
```

The screenshot shows a desktop environment with multiple windows. On the left is a file manager window showing project files like `disks.hpp`, `README.md`, and `rubrictest.hpp`. In the center is a code editor window for `disks.hpp` in Atom, displaying the same lawnmower algorithm code as the previous screenshot. To the right is a terminal window showing the command `make disks_test` being run, followed by the output of the test program which shows various disk configurations and their scores.

```
student@tuffix-vm:~/Desktop/project-lawnmover-main$ make disks_test
g++ -std=c++11 -Wall disk_test.cpp -o disks_test
disks still works: passed, score 1/1
sorted disks still works: passed, score 1/1
disk_state::is_sorted: passed, score 3/3
disk_state::is_initialized: passed, score 3/3
alternate, n=0: passed, score 1/1
alternate, n=1: passed, score 1/1
alternate, n=2: 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
```

Lawnmover

```
def sort_lawnmover ( disk_state ):  
    number of swap = 0 + 1  
    n = disk_state.size() + 1  
    for i=1 to  $\frac{n}{2}$  :  
        for left=1 to n-1 :  
            if (left color == Dark && right color == light) + 3  
                swap  
                number of swap++ + 1  
            end if  
        end for  
  
        for right=n-1 to 1 :  
            if (left color == Dark && right color == light) + 3  
                swap  
                number of swap++ + 1  
            end if  
        end for  
  
    end for  
return;
```

Step Count:

$$\begin{aligned} & 2 + \sum_{i=1}^{\frac{n}{2}} \left(\sum_{j=1}^{n-1} 4 + \sum_{j=1}^{n-1} 4 \right) \\ &= 2 + \sum_{i=1}^{\frac{n}{2}} (4(n-1-1+1) + 4(h-1-1+1)) \\ &= 2 + \sum_{i=1}^{\frac{n}{2}} (8n-8) \\ &= 2 + (8n-8)(\frac{n}{2}-1+1) \\ &= 4h^2 - 4h + 2 \end{aligned}$$

Prove $4n^2 - 4n + 2 \in O(n^2)$

$$\text{take } \frac{4n^2 - 4n + 2}{n^2} = 4 - \frac{4}{n} + \frac{2}{n^2}$$

$$\text{as } n \rightarrow \infty$$

$$4 \rightarrow 4$$

$$-\frac{4}{n} \rightarrow 0$$

$$\frac{2}{n^2} \rightarrow 0$$

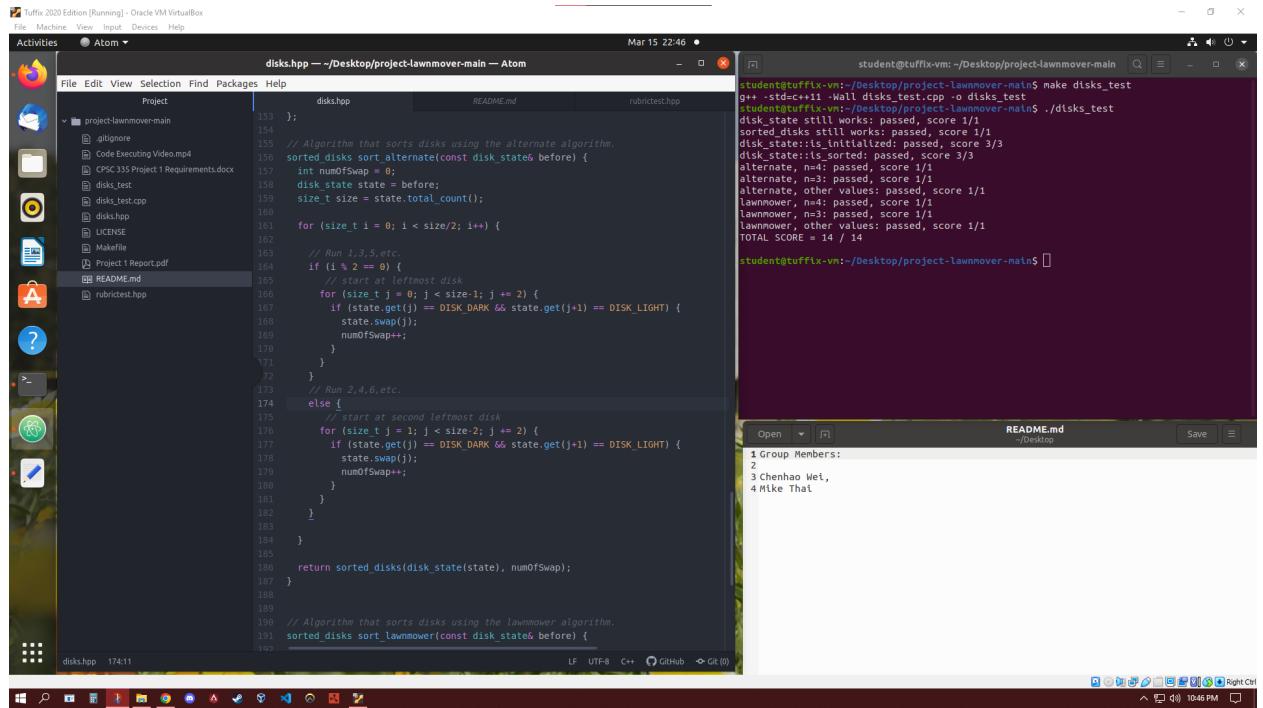
$$\text{hence } \frac{4n^2 - 4n + 2}{n^2} \rightarrow 4 \geq 0$$

Theorefore $4n^2 - 4n + 2 \in O(n^2)$

Alternate Algorithm

```
144 // Algorithm that sorts disks using the alternate algorithm.
145 sorted_disks sort_alternate(const disk_state& before) {
146     int numOfSwap = 0;
147     disk_state state = before;
148     size_t size = state.total_count();
149
150     for (size_t i = 0; i < size/2; i++) {
151
152         if (i % 2 == 0) {
153             for (size_t j = 0; j < size; j += 2) {
154                 if (state.get(j) == DISK_DARK && state.get(j+1) == DISK_LIGHT) {
155                     state.swap();
156                     numOfSwap++;
157                 }
158             }
159         } else {
160             for (size_t j = 1; j < size-1; j += 2) {
161                 if (state.get(j) == DISK_LIGHT && state.get(j+1) == DISK_DARK) {
162                     //state[j] = DISK_LIGHT;
163                     //state[j+1] = DISK_DARK;
164                     state.swap();
165                     numOfSwap++;
166                 }
167             }
168         }
169     }
170
171     return sorted_disks(disk_state(state), numOfSwap);
172 }
173
174 }
```

```
line 81 of file disks_test.cpp, message: is_sorted() for n=3
score 0/3
disks_test: disks.hpp:70: void disk_state::swap(size_t): Assertion `is_index(right_
Aborted (core dumped)
student@tuffix-vn:~/Desktop/project-lawnmover-main$ make disks_test
g++ -std=c++11 -Wall disks_test.cpp -o disks_test
student@tuffix-vn:~/Desktop/project-lawnmover-main$ ./disks_test
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:
    TEST FAILED:
        line 81 of file disks_test.cpp, message: is_sorted() for n=3
        score 0/3
alternate, n=4: passed, score 1/1
alternate, n=3: passed, score 1/1
alternate, other values: passed, score 1/1
lawnmower, n=4:
    TEST FAILED:
        line 123 of file disks_test.cpp, message: number of swaps must be 6
        score 0/1
lawnmower, n=3:
    TEST FAILED:
        line 130 of file disks_test.cpp, message: number of swaps must be 3
        score 0/1
lawnmower, other values:
    TEST FAILED:
        line 140 of file disks_test.cpp, message: n=10 gives 45 swaps
        score 0/1
TOTAL SCORE = 8 / 14
student@tuffix-vn:~/Desktop/project-lawnmover-main$
```



Tuffix 2020 Edition [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Activities Atom

disks.hpp -- ~/Desktop/project-lawnmover-main -- Atom

File Edit View Selection Find Packages Help

Project

disks.hpp README.md rubrictest.hpp

```
153 };
154 // Algorithm that sorts disks using the alternate algorithm.
155 sorted_disks sort_alternate(const disk_state& before) {
156     int numOfSwap = 0;
157     disk_state state = before;
158     size_t size = state.total_count();
159
160     for (size_t i = 0; i < size/2; i++) {
161
162         if (i % 2 == 0) {
163             for (size_t j = 0; j < size-1; j += 2) {
164                 if (state.get(j) == DISK_DARK && state.get(j+1) == DISK_LIGHT) {
165                     state.swap();
166                     numOfSwap++;
167                 }
168             }
169         } else {
170             for (size_t j = 1; j < size-1; j += 2) {
171                 if (state.get(j) == DISK_LIGHT && state.get(j+1) == DISK_DARK) {
172                     //state[j] = DISK_LIGHT;
173                     //state[j+1] = DISK_DARK;
174                     state.swap();
175                     numOfSwap++;
176                 }
177             }
178         }
179     }
180
181     return sorted_disks(disk_state(state), numOfSwap);
182 }
183
184 }
185
186
187
188
189
190 // Algorithm that sorts disks using the lawnmower algorithm.
191 sorted_disks sort_lawnmower(const disk_state& before) {
192 }
```

student@tuffix-vn:~/Desktop/project-lawnmover-main\$ make disks_test
g++ -std=c++11 -Wall disks_test.cpp -o disks_test
student@tuffix-vn:~/Desktop/project-lawnmover-main\$./disks_test
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:
 TEST FAILED:
 line 81 of file disks_test.cpp, message: is_sorted() for n=3
 score 0/3
alternate, n=4: passed, score 1/1
alternate, n=3: passed, score 1/1
alternate, other values: passed, score 1/1
lawnmower, n=4:
 TEST FAILED:
 line 123 of file disks_test.cpp, message: number of swaps must be 6
 score 0/1
lawnmower, n=3:
 TEST FAILED:
 line 130 of file disks_test.cpp, message: number of swaps must be 3
 score 0/1
lawnmower, other values:
 TEST FAILED:
 line 140 of file disks_test.cpp, message: n=10 gives 45 swaps
 score 0/1
TOTAL SCORE = 8 / 14
student@tuffix-vn:~/Desktop/project-lawnmover-main\$

student@tuffix-vn:~/Desktop/project-lawnmover-main\$

README.md

Open Save

1 Group Members:
2
3 Chenhao Wei,
4 Mike Thal

LF UTE-B C++ GitHub ⌂ Right Ctrl

Alternate Algo

List of disks, L

Size of list, 2n

count = 0 — 1 tu

for (i = 0 to n)

if ($i \% 2 == 0$) — 2 tu

for (j = 0 to 2n - 1) step 2

if ($L[j] == \text{dark}$ & & $L[j+1] == \text{light}$) — 1 tu

$L[j] = \text{light}$

$L[j+1] = \text{dark}$

count ++

endif

endfor

else

for (j = 1 to 2n - 2) step = 2

if ($L[j] == \text{dark}$ & & $L[j+1] == \text{light}$) — 1 tu

$L[j] = \text{light}$

$L[j+1] = \text{dark}$

count ++

endif

endfor

return count — 0 tu

Alternate Algo SC

$$SC_{\text{inner if 1}} = 4 + \max(4, 0) = 8$$

$$SC_{\text{inner if 2}} = 4 + \max(4, 0) = 8$$

$$\begin{aligned} SC_{\text{inner for 1}} &= \text{iteration} * SC_{\text{inner if 1}} \\ &= \left(\frac{2n-1-0}{2} + 1\right) * 8 = \left(n + \frac{1}{2}\right) * 8 = 8n + 4 \end{aligned}$$

$$\begin{aligned} SC_{\text{inner for 2}} &= \text{iteration} * SC_{\text{inner if 2}} \\ &= \left(\frac{2n-2-1}{2} + 1\right) * 8 = \left(n - \frac{1}{2}\right) * 8 = 8n - 4 \end{aligned}$$

$$\begin{aligned} SC_{\text{outer if}} &= 2 + \max(SC_{\text{inner for 1}}, SC_{\text{inner for 2}}) \\ &= 2 + \max(8n + 4, 8n - 4) = 6 + 8n \end{aligned}$$

$$\begin{aligned} SC_{\text{outer for}} &= \text{iteration} * SC_{\text{outer if}} \\ &= (n - 0 + 1) * (6 + 8n) \\ &= 6n + 6 + 8n^2 + 8n = 8n^2 + 14n + 6 \end{aligned}$$

$$\begin{aligned} SC_{\text{Alternate}} &= 1 + SC_{\text{outer for}} \\ &= 1 + 8n^2 + 14n + 6 \\ &= 8n^2 + 14n + 7 \end{aligned}$$

Alternate Algo Efficiency class

$$\text{prove: } 8n^2 + 14n + 7 \in O(n^2)$$

By def:

$$8n^2 + 14n + 7 \leq c \cdot n^2 \quad \forall n \geq n_0$$

$$\left(\text{choose: } c = 8 + 14 + 7 = 29 \right)$$

$$n_0 = 1$$

$$\rightarrow 8n^2 + 14n + 7 \leq 29n^2$$

$$8 \cdot 1 + 14 \cdot 1 + 7 \leq 29 \cdot 1$$

$$29 \leq 29 \rightarrow \text{True}$$

$$\rightarrow 8n^2 + 14n + 7 \in O(n^2)$$

\Rightarrow Alternate Algo Time Efficiency is $O(n^2)$