

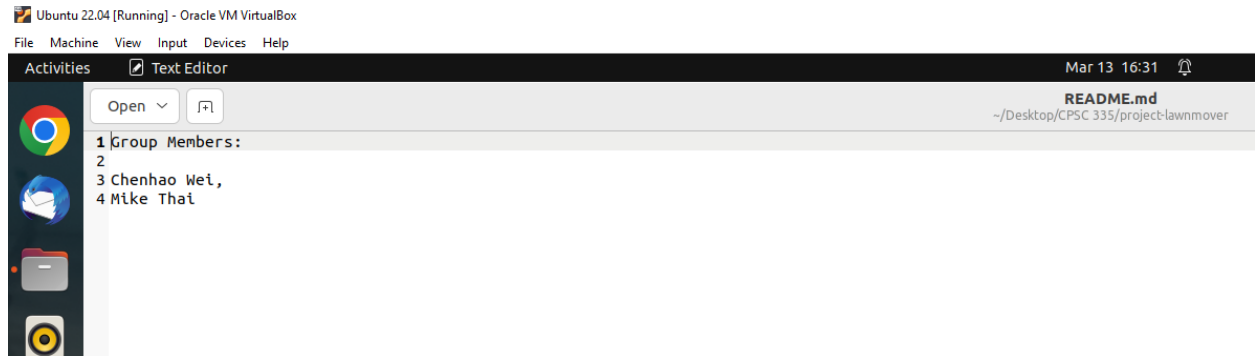
Project 1

Mike Thai - miket126@csu.fullerton.edu

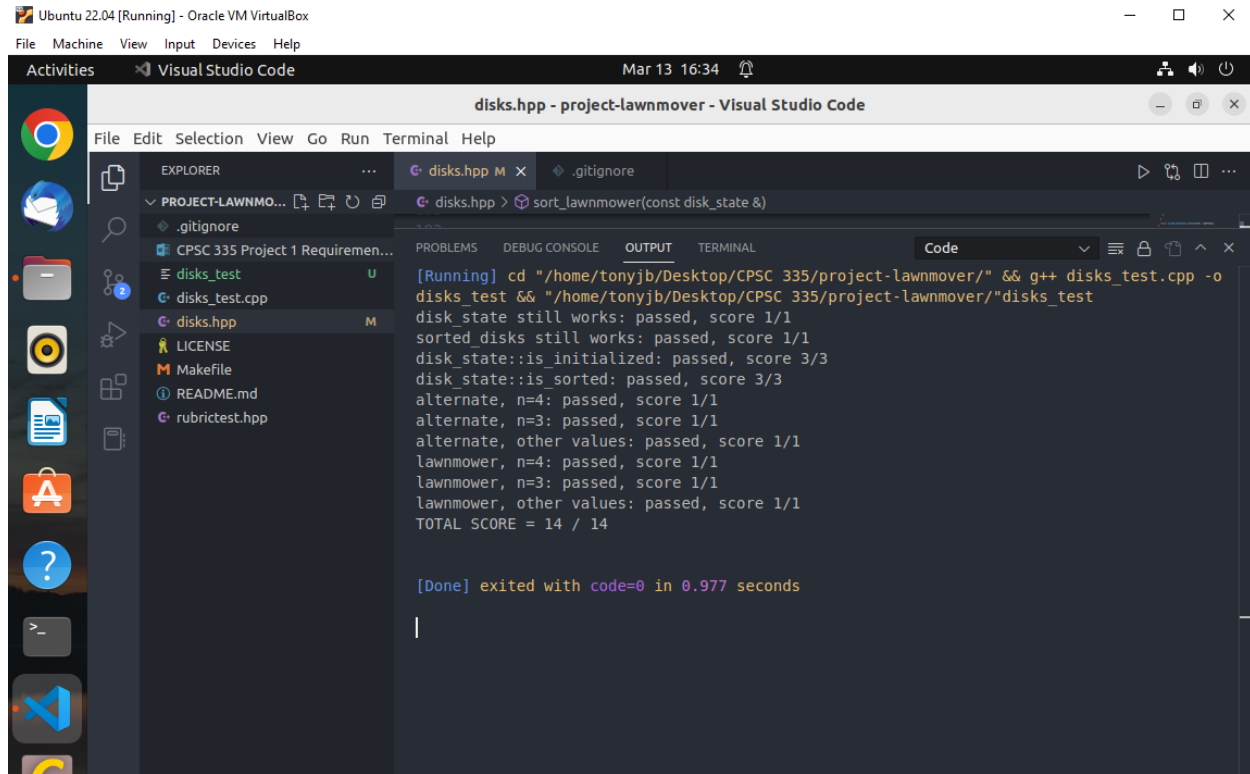
Chenhao Wei - chenhao.wei@csu.fullerton.edu

Due: March 19, 2023

README Screenshot



Code Compiling and Executing Screenshot



The screenshot shows the Visual Studio Code interface within an Ubuntu 22.04 virtual machine. The Explorer sidebar on the left displays the project structure for 'PROJECT-LAWNMOWER', including files like `.gitignore`, `CPSC 335 Project 1 Requirements.txt`, `disks_test`, `disks_test.cpp`, `disks.hpp`, `LICENSE`, `Makefile`, `README.md`, and `rubrictest.hpp`. The main editor area shows the `disks.hpp` file. The Output window at the bottom right displays the execution results of the `disks_test` program, showing various test cases passing with scores.

```
[Running] cd "/home/tonyjb/Desktop/CPSC 335/project-lawnmover/" && g++ disks_test.cpp -o disks_test && "/home/tonyjb/Desktop/CPSC 335/project-lawnmover/"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: 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

[Done] exited with code=0 in 0.977 seconds
```

Lawnmower Algorithm

```
disks.hpp M X .gitignore
disks.hpp > sort_lawnmower(const disk_state &)

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

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:

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

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

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

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

$$4 \rightarrow 4$$

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

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

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

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

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     for (size_t i = 0; i < size/2; i++) {
150         if (i % 2 == 0) {
151             for (size_t j = 0; j < size; j += 2) {
152                 if (state.get(j) == DISK_DARK && state.get(j+1) == DISK_LIGHT) {
153                     state.swap(j);
154                     numOfSwap++;
155                 }
156             }
157         }
158         else {
159             for (size_t j = 1; j < size-1; j += 2) {
160                 if (state.get(j) == DISK_DARK && state.get(j+1) == DISK_LIGHT) {
161                     //state[j] = DISK_LIGHT;
162                     //state[j+1] = DISK_DARK;
163                     state.swap(j);
164                     numOfSwap++;
165                 }
166             }
167         }
168     }
169     return sorted_disks(disk_state(state), numOfSwap);
170 }
171 }
172
173
174
175
```

```
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$
```

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 ($L[j] == \text{dark} \text{ \& \& } L[j+1] == \text{light}$) — 4 tu

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

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

count ++

) 4 tu

endif
endfor

else

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

if ($L[j] == \text{dark} \text{ \& \& } L[j+1] == \text{light}$) — 4 tu

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

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

count ++

) 4 tu

endif
endfor

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

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

By def:

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

$$\left(\begin{array}{l} \text{Choose: } C = 8 + 14 + 7 = 29 \\ n_0 = 1 \end{array} \right)$$

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

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

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

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

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