Chase Huante
Shaochen Ren
Dream Team

Written Project 1 Report

Project brief description:

# Description
For project 1 is a demonstortion sorting algorithm in the following way.
# is_initialized()
  Return true when this disk_state is in alternating format. That means that the first disk at index 0 is dark, the second disk at index 1 is light, and so on for the entire row of disks. bool is_initialized() const {

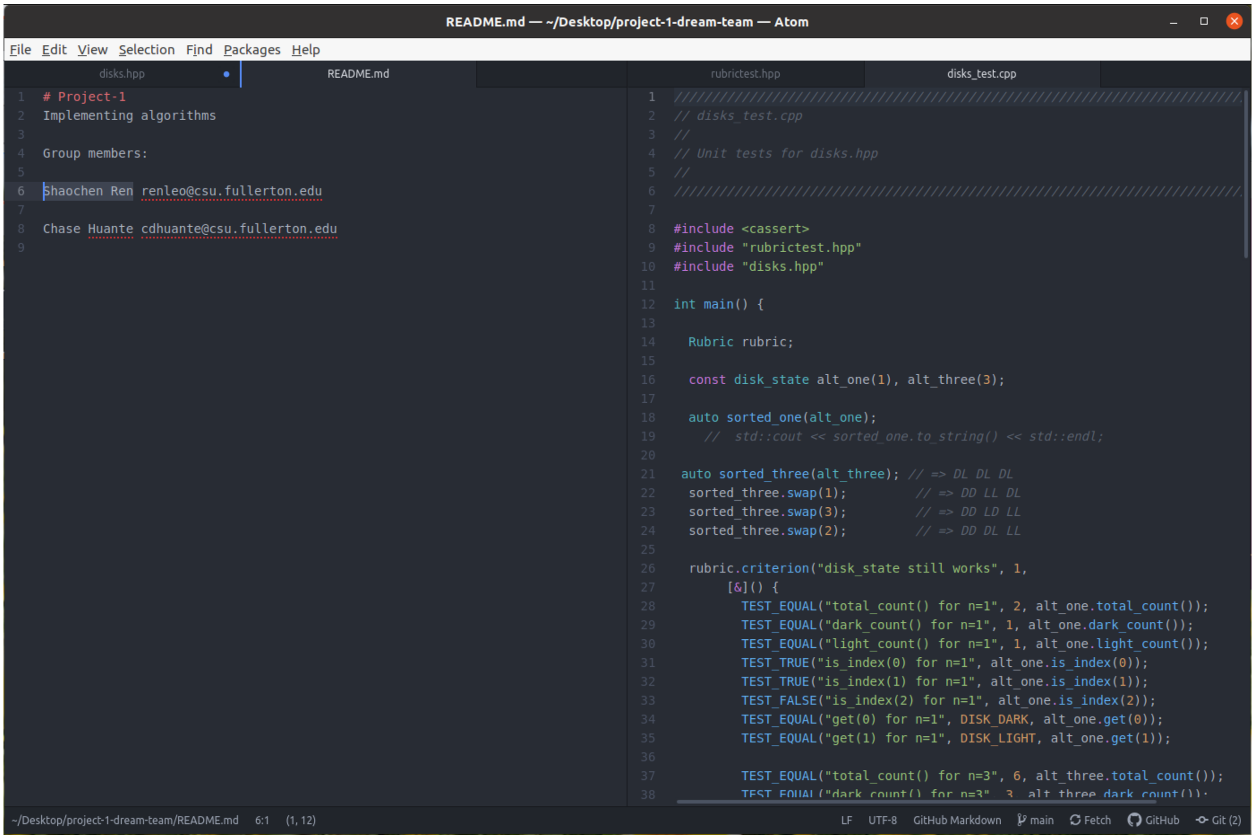
# is_sorted()
  Return true when this disk_state is fully sorted, with all dark disks on the left (low indices) and all light disks on the right (high indices).
# sort_alternate()
Algorithm that sorts disks using the alternate algorithm.
# sort lawnmower
Algorithm that sorts disks using the lawnmower algorithm.

Finally, produce a brief written project report ***in PDF format***. Submit your PDF by committing it to your GitHub repository along with your code. Your report should include the following:

1. Your names, CSUF-supplied email address(es), and an indication that the submission is for project 1.

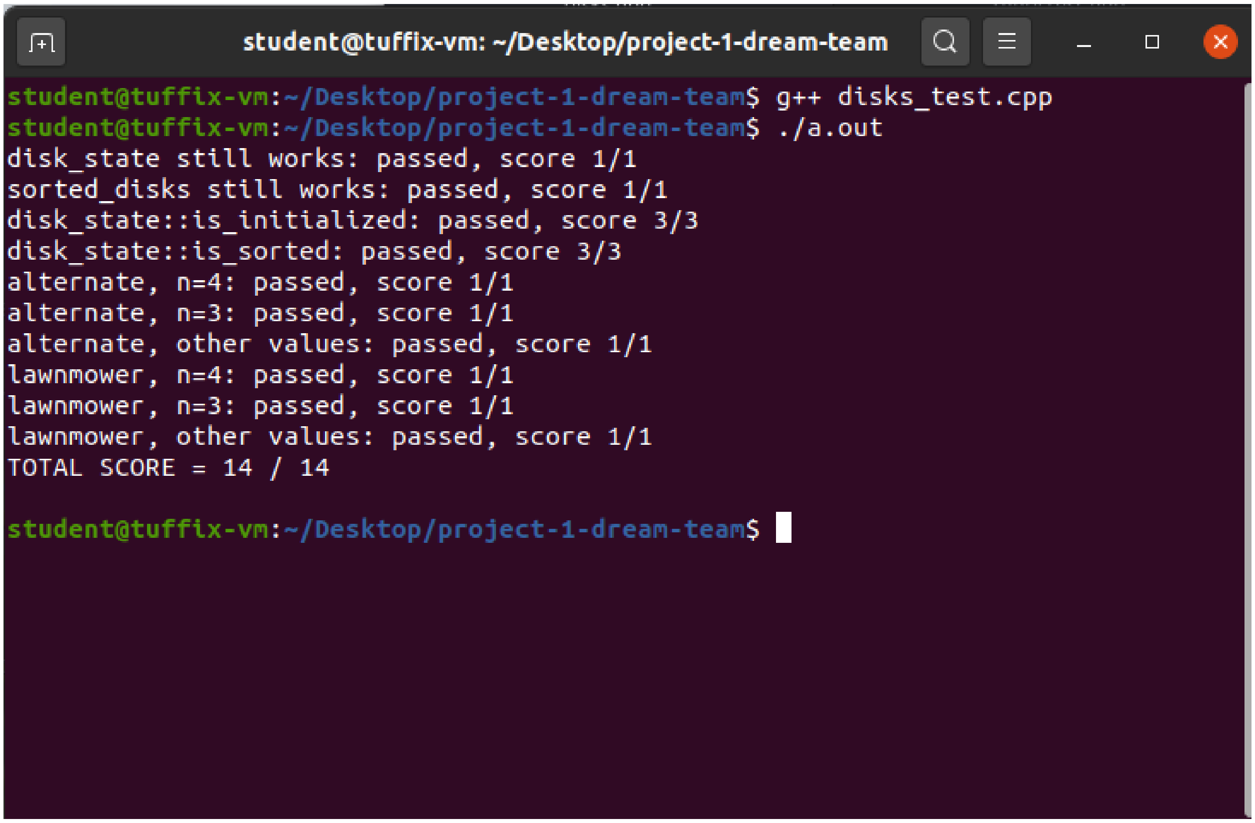
Chase Huante	cdhuante@csu.fullerton.edu
Shaochen Ren	renleo@csu.fullerton.edu

2. Two full-screen screenshots: one inside Tuffix, showing the Atom editor, with your group member names inside Atom. One way to make your names appear in Atom is to simply open your README.md. The second screenshot is with your code executing the command make.


README.md — ~/Desktop/project-1-dream-team — Atom
File Edit View Selection Find Packages Help
disks.hpp
README.md
rubrictest.hpp
disks_test.cpp
~/Desktop/project-1-dream-team/README.md 6:1 (1, 12)
LF UTF-8 GitHub Markdown main Fetch GitHub Git (2)

```
# Project-1
Implementing algorithms

Group members:

Shaochen Ren renleo@csu.fullerton.edu

Chase Huante cdhuante@csu.fullerton.edu

```

```cpp
/////////////////////////////////////////////////////////////////////////////
// disks_test.cpp
//
// Unit tests for disks.hpp
//
/////////////////////////////////////////////////////////////////////////////

#include <cassert>
#include "rubrictest.hpp"
#include "disks.hpp"

int main() {

  Rubric rubric;

  const disk_state alt_one(1), alt_three(3);

  auto sorted_one(alt_one);
    //  std::cout << sorted_one.to_string() << std::endl;

 auto sorted_three(alt_three); // => DL DL DL
  sorted_three.swap(1);         // => DD LL DL
  sorted_three.swap(3);         // => DD LD LL
  sorted_three.swap(2);         // => DD DL LL

  rubric.criterion("disk_state still works", 1,
       [&]() {
         TEST_EQUAL("total_count() for n=1", 2, alt_one.total_count());
         TEST_EQUAL("dark_count() for n=1", 1, alt_one.dark_count());
         TEST_EQUAL("light_count() for n=1", 1, alt_one.light_count());
         TEST_TRUE("is_index(0) for n=1", alt_one.is_index(0));
         TEST_TRUE("is_index(1) for n=1", alt_one.is_index(1));
         TEST_FALSE("is_index(2) for n=1", alt_one.is_index(2));
         TEST_EQUAL("get(0) for n=1", DISK_DARK, alt_one.get(0));
         TEST_EQUAL("get(1) for n=1", DISK_LIGHT, alt_one.get(1));

         TEST_EQUAL("total_count() for n=3", 6, alt_three.total_count());
         TEST_EQUAL("dark_count() for n=3", 3, alt_three.dark_count());
```



```
student@tuffix-vm:~/Desktop/project-1-dream-team$ g++ disks_test.cpp
student@tuffix-vm:~/Desktop/project-1-dream-team$ ./a.out
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

student@tuffix-vm:~/Desktop/project-1-dream-team$ 
```

3. Two pseudocode listings, for the two algorithms.

Alternating algorithm:

```
sorted_disks sort_alternate(const disk_state &before) {
Index = 0;    SC:1
disk_state after = disk_state(before.light_count()); SC: n
For 0 to 2n do  SC: (2n-0+1) = 2n+1
  if(i%2 ==1) then   SC: 2
     For 1 to 2n-1 skips 2 do   SC:(2n-1-1+1)/2 = (2n-1)/2
       swap = after.get(j);   SC:1
       swap1 = after.get(j + 1); SC:1
       if (swap == DISK_LIGHT && swap1 == DISK_DARK) SC: 3
         swap(j);  SC: 1
         index++; SC:1
  Else
       For 1 to 2n skips 2 do   SC:(2n-1+1)/2 = n
         swap = after.get(j);   SC:1
         swap1 = after.get(j + 1); SC:1
         if (swap == DISK_LIGHT && swap1 == DISK_DARK) SC: 3
           swap(j);  SC: 1
           index++; SC:1
```

The first (if)nest if and for SC are (2n-1)/2 *1 *1*(3+max(2,0))) = 5n-5/2
The else part nest if and for SC are n*1*1*(3+max(2,0)) = 5n
TOTAL SC = 1 + n + (2n+1)*(2+max(5n-5/2,5n) = 1+n+(2n+1)*(5n+2)
= 10n^2+8n+4

Proof:

10n^2+8n+4 belongs to O(n^2)

Find c>0 and n0>n st 10n^2+8n+4

Choose c = 10+8+4 = 22

10n^2+8n+4 <= 22n^2

22n^2 - 10n^2 -8n -4 >=0

12n^2 -8n-4 >=0

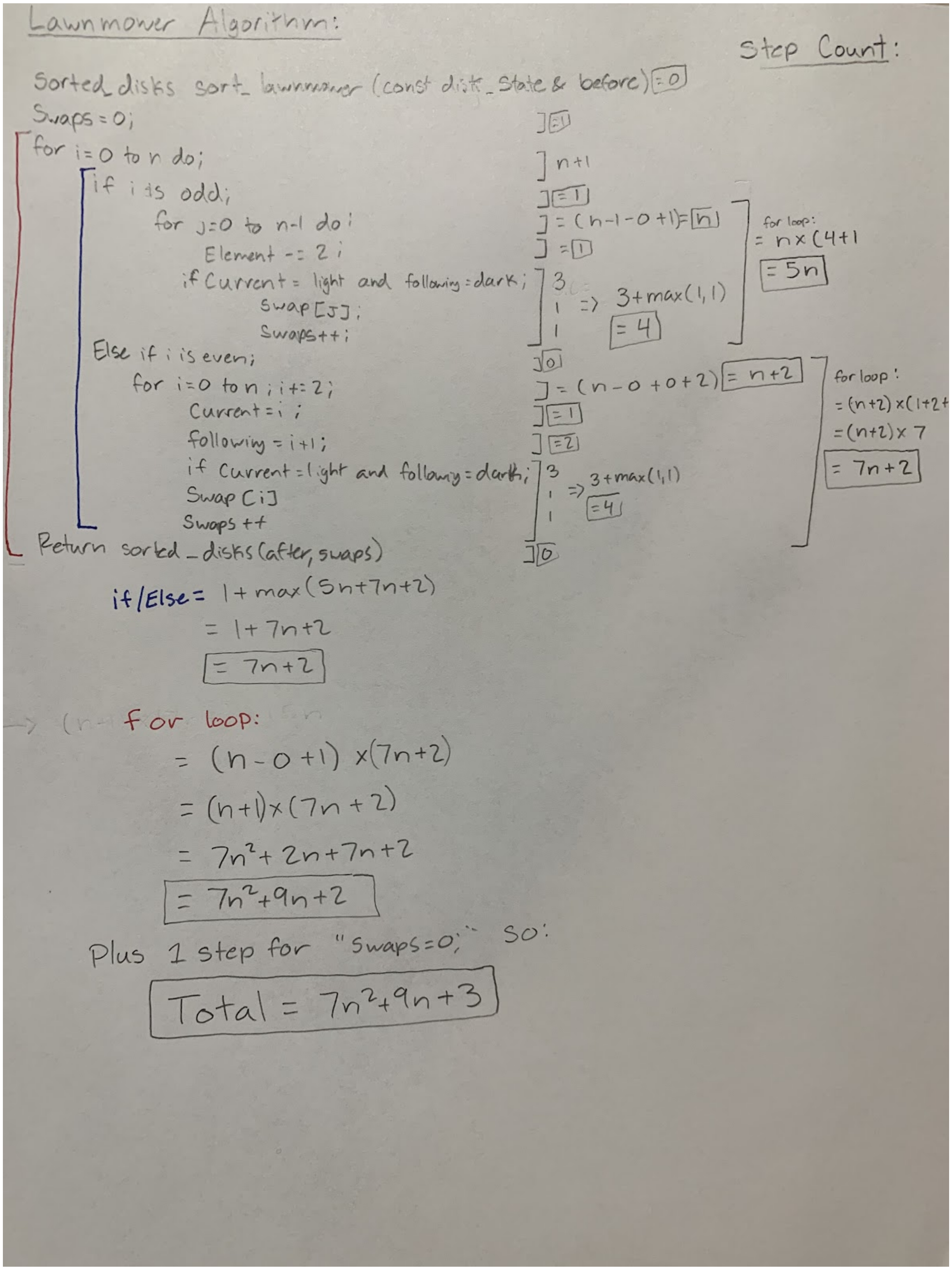
8n^2 - 8n+4n^2-4>=0 V n>=0 n0 = 1

By definition that 10n^2+8n+4 belongs to O(n^2)

Lawnmower algorithm:

```
Sorted_disks sort_lawnmower()
Swaps = 0; SC: 1
For i = 0 to n do: (n-0+1) = n+1
    If i is odd:
        For j = 0 to n-1 do: (n-1-0+1) = n
            Element -= 2
            Current = element
            Following = element+1
            If current = light and following = dark
                Swap [ j ]
                Swaps++
    Else if i is even:
        For i = 0 to n; i += 2;
            Current = i
            Following = i+1
            If current = light and following = dark
                Swap [ i ]
            Swaps++
Return sorted_disks(after, swaps)
```

Lawnmower algorithm:



Lawnmower Algorithm:

Step Count:

```
Sorted_disks sort_lawnmower (const disk_State & before)    ]=0
Swaps = 0;                                                   ]=1
for i = 0 to n do;                                           ] n+1
    if i is odd;                                             ]=1
        for j = 0 to n-1 do;                                 ] = (n-1-0+1) = n      for loop:
            Element -= 2;                                    ] = 1                  = n x (4+1
            if Current = light and following = dark;         ] 3                    = 5n
                Swap[J];                                     | 1 => 3+max(1,1)
                Swaps++;                                     | 1    = 4
    Else if i is even;                                       ]0
        for i = 0 to n; i+=2;                                ] = (n-0+0+2) = n+2    for loop:
            Current = i;                                     ]=1                    = (n+2) x (1+2+
            following = i+1;                                 ]=2                    = (n+2) x 7
            if Current = light and following = dark;         ] 3                    = 7n+2
            Swap [i]                                         | 1 => 3+max(1,1)
            Swaps ++                                         | 1    = 4
Return sorted_disks(after, swaps)                            ]0
```

if/Else $= 1 + \max(5n+7n+2)$

$= 1 + 7n + 2$

$\boxed{= 7n+2}$

for loop:

$= (n-0+1) \times (7n+2)$

$= (n+1) \times (7n+2)$

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

$\boxed{= 7n^2 + 9n + 2}$

Plus 1 step for "Swaps=0;" so:

$\boxed{\text{Total} = 7n^2 + 9n + 3}$

Proof:
7n^2 + 9n+3 belongs to O(n^2)

Find c>0 and n0>n st 7n^2 + 9n+3

Choose $c = 7+ 9 + 3 = 19$

$7n^2 + 9n+3 <= 19*n^2$

$19n^2-7n^2-9n-3 >=0$

$12n^2-9n-3>=0$

$9n^2 - 9n +3n^2 + 3 =>0$ V $n>=0$ $n0 = 1$

By definition that $7n^2 + 9n+3$ belongs to $O(n^2)$