CPSC 335 Project 1

Justin Jhern
justinjhern@csu.fullerton.edu

Algo 1 Lawnmower Pseudocode

```
# Input: a positive integer n and a list of 2n disks of alternating colors light-dark, starting with light
# Output: a list of 2n disks, the first n disks are light, the next n disks are dark, and an integer m representing the number of swaps to move the dark ones after the light ones.
#
# for i = 0 to ciel(n/2) times do
#       for int j = 0 to 2n do
#           if L[j] is dark and L[j+1] is black:
#               swap L[j] with L[j+1]
#        for int j = 2n-1 to 0 do step -j
#           if L[j] is white and L[j-1] is black:
#               swap L[j-1] with L[j]
#   if !hasswap break;
```

Algo 2 Alternate Pseudocode

```
# Input: a positive integer n and a list of 2n disks of alternating colors light-dark, starting with light
# Output: a list of 2n disks, the first n disks are light, the next n disks are dark, and an integer m representing the number of swaps to move the dark ones after the light ones.
#
# for i=0 to n + 1
#   if i%2 == 0
#       for j = 0 to 2n step 2
#           if L[j] is dark and L[j+1] is light
#               swap L[j] and L[j+1]
#   else
#       for j = 1 to 2n-2 step 2
#           if L[j] is dark and L[j+1] is light
#               swap L[j] and L[j+1]
```

Lawnmower algorithm step count

```
sorted_disks sort_lawnmower(const disk_state &before) {
   int numOfSwap = 0;  // record # of step swap                                  1 tu
   disk_state after = before;                                                    1 tu
   size_t n = after.total_count() / 2;                                           2 tu
   for (int i = 0; i < std::ceil((int)n / 2); ++i) {                   n/2-0 + 1
      bool hasSwap = false;                                                      1 tu
      for (size_t j = 0; j < 2 * n - 1; ++j) {                         2*n-1 -0 +1
         if (after.get(j) == DISK_DARK && after.get(j + 1) == DISK_LIGHT) {               4 tu
            after.swap(j);                                                       -
            hasSwap = true;                                                      1
            numOfSwap++;                                                         1
         }
      }
      for (size_t j = 2 * n - 1; j > 0; --j) {                         (0-(2n-1)/-1) + 1
         if (after.get(j) == DISK_LIGHT && after.get(j - 1) == DISK_DARK) {      4 tu
            after.swap(j - 1);                                         1 tu
            hasSwap = true;                                                      1 tu
            numOfSwap++;                                                         1 tu
         }
      }
      if (!hasSwap) {                                                            don't know if !
counts as one time unit - won't count
         break;
      }
   }
   return sorted_disks(disk_state(after), numOfSwap);
}
```

Inner two loops

```
for (size_t j = 0; j < 2 * n - 1; ++j) {                              2*n-1 -0 +1
         if (after.get(j) == DISK_DARK && after.get(j + 1) == DISK_LIGHT) {      4 tu
         after.swap(j);                                                -
         hasSwap = true;                                                         1
         numOfSwap++;                                                            1
         }
}
```

(2 * n - 1 - 0 +1) * (4 + 2)

2n*6

12n

```
for (size_t j = 2 * n - 1; j > 0; --j) {                                    (0-(2n-1)/-1) + 1
        if (after.get(j) == DISK_LIGHT && after.get(j - 1) == DISK_DARK) {          4 tu
                after.swap(j - 1);                                                  1 tu
                hasSwap = true;                                                            1 tu
           numOfSwap++;                                                             1 tu
     }
}
```

((0-(2n-1)/-1) + 1) * 7
((-2n-1)/-1 + 1) * 7
(2n+2)*7
14n+14

all together
14n+14 + 12n
=26n + 14

Inner loops and outer loop

```
for (int i = 0; i < std::ceil((int)n / 2); ++i) ciel(n/2) - 0 + 1
```

((n+1)/2 + 1) (26n + 14) changing ciel to n+1/2 since its effectively the same thing in code
(n/2+3/2)(26n+14)

=13n^2 + 46n + 21

Total

13n^2 + 46n + 21 + 4

Step Count = 13n^2 + 46n + 25

Proof by limit theorem
----
f(n) exists in O(n^2)
13n^2 + 46n + 25 exists in O(n^2)
lim n->inf (13n^2 + 46n + 25)/n^2
lim n->inf 13 + 46/n + 25/n^2
as n -> inf f(n) approaches 0
exists

Proof by definition

$f(n) \le cn^2$
$13n^2 + 46n + 25 \le c\,n^2$
$c = 13+46+25 = 84$
$n=2$
$13n^2 + 46n + 25 \le 84 * n^2$
$169 \le 336$

Alternate sort step count

```
   int numOfSwap = 0;
1 tu
   disk_state after = before;                                                  1 tu
   size_t n = after.total_count() / 2;                                         2 tu
   if (n == 0) return sorted_disks(disk_state(after), numOfSwap);              1 tu
   for (int i = 0; i < (int)n + 1; ++i) {                                  n+1-0+1
      if (i % 2 == 0) {                                                    2 tu + max(
         for (size_t j = 0; j < n * 2; j = j + 2) {                        (2n-0)/2+1
            if (after.get(j) == DISK_DARK && after.get(j + 1) == DISK_LIGHT) {     4 tu
               after.swap(j);
               numOfSwap++;                                                1 tu
            }
         }
      } else {
         for (size_t j = 1; j < n * 2 - 2; j = j + 2) {                    (2n-2)/2-1+1
            if (after.get(j) == DISK_DARK && after.get(j + 1) == DISK_LIGHT) {     4 tu
               after.swap(j);
               numOfSwap++;                                                1 tu
            }
         }
      }
   }
   return sorted_disks(disk_state(after), numOfSwap);
}
```

Inner loops

```
for (size_t j = 0; j < n * 2; j = j + 2) {                          (2n-0)/2+1
         if (after.get(j) == DISK_DARK && after.get(j + 1) == DISK_LIGHT) {      4 tu
         after.swap(j);
            numOfSwap++;                                            1 tu
         }
}

for (size_t j = 1; j < n * 2 - 2; j = j + 2) {                      (2n-2)/2-1+1
         if (after.get(j) == DISK_DARK && after.get(j + 1) == DISK_LIGHT) {      4 tu
         after.swap(j);
                  numOfSwap++;                                               1 tu
         }
}
```

(2n-0)/2+1 * (5)
n+1(5)
5n+5

(2n-2)/2-1+1 * (5)
n-1 * (5)
5n-5

If statement
----
if (i % 2 == 0) 2 tu
2 tu + max(5n+5, 5n-5)
2 + 5n+5
5n+7 tu

Outer Loop
----
for (int i = 0; i < (int)n + 1; ++i) { n+1-0+1
(n+2)(5n+7)
5n^2 + 17n + 14

Total

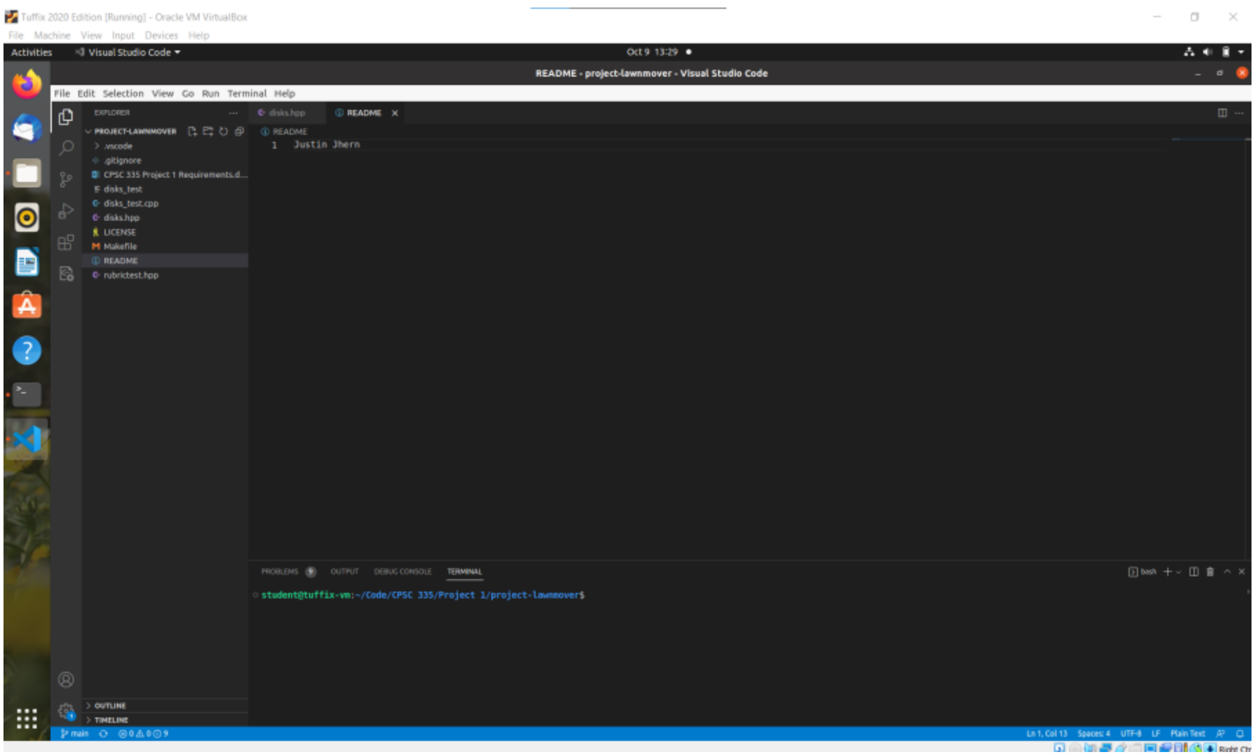
5n^2 + 17n + 14 + 5
Step Count = 5n^2 + 17n + 19

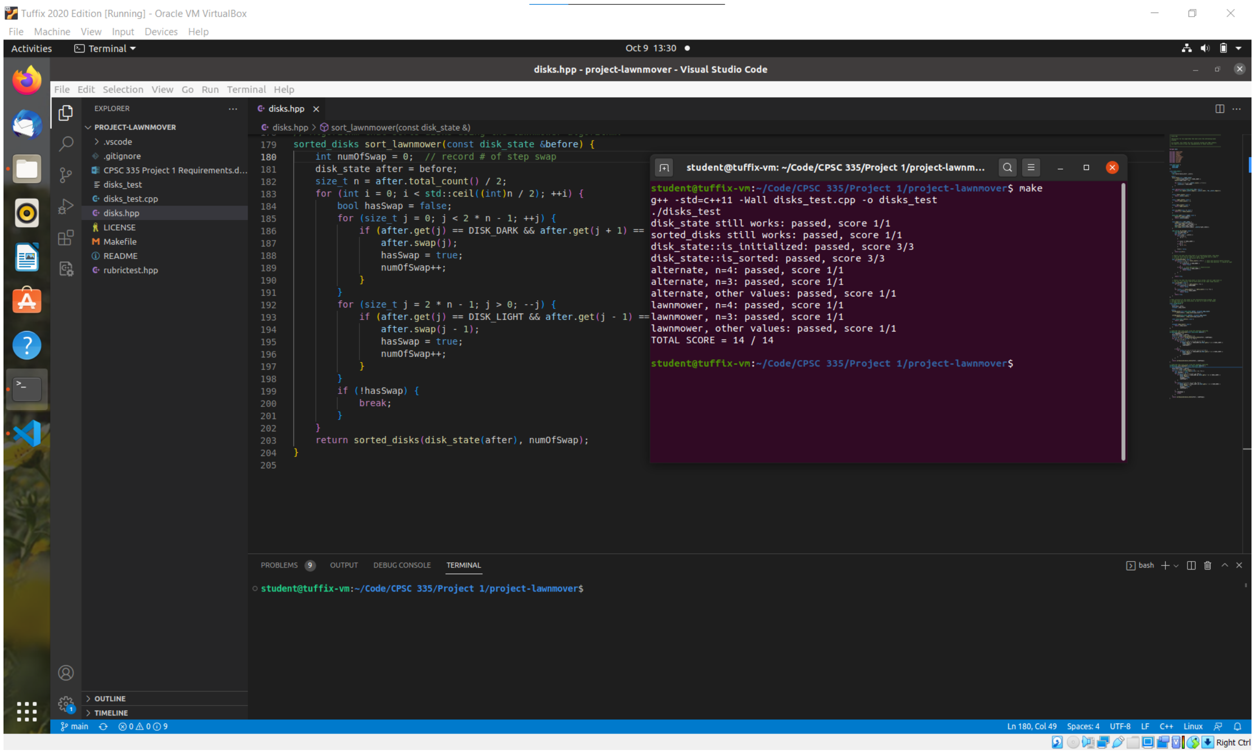
Proof by limit theorem
----
f(n) exists in O(n^2)
lim as n-> inf (5n^2 + 17n + 19)/n^2
lim as n-> inf 5+17/n + 19/n^2
as n -> inf f(n) goes to 0
exists

Proof by definition
----
f(n) <= cn^2
c = 5 + 17 + 19 n=2

5n^2 + 17n + 19 <= 41n^2
73 <= 164
works

Screenshots




Tuffix 2020 Edition [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Oct 9 13:30
disks.hpp - project-lawnmover - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
PROJECT-LAWNMOVER
.vscode
.gitignore
CPSC 335 Project 1 Requirements.d...
disks_test
disks_test.cpp
disks.hpp
LICENSE
Makefile
README
rubrictest.hpp
disks.hpp
disks.hpp > sort_lawnmower(const disk_state &)
sorted_disks sort_lawnmower(const disk_state &before) {
int numOfSwap = 0; // record # of step swap
disk_state after = before;
size_t n = after.total_count() / 2;
for (int i = 0; i < std::ceil((int)n / 2); ++i) {
bool hasSwap = false;
for (size_t j = 0; j < 2 * n - 1; ++j) {
if (after.get(j) == DISK_DARK && after.get(j + 1) ==
after.swap(j);
hasSwap = true;
numOfSwap++;
}
}
for (size_t j = 2 * n - 1; j > 0; --j) {
if (after.get(j) == DISK_LIGHT && after.get(j - 1) ==
after.swap(j - 1);
hasSwap = true;
numOfSwap++;
}
}
if (!hasSwap) {
break;
}
}
return sorted_disks(disk_state(after), numOfSwap);
}
student@tuffix-vm: ~/Code/CPSC 335/Project 1/project-lawnm...
student@tuffix-vm:~/Code/CPSC 335/Project 1/project-lawnmover$ make
g++ -std=c++11 -Wall disks_test.cpp -o disks_test
./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
student@tuffix-vm:~/Code/CPSC 335/Project 1/project-lawnmover$
PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL
bash
student@tuffix-vm:~/Code/CPSC 335/Project 1/project-lawnmover$
OUTLINE
TIMELINE
main
Ln 180, Col 49 Spaces: 4 UTF-8 LF C++ Linux
Right Ctrl