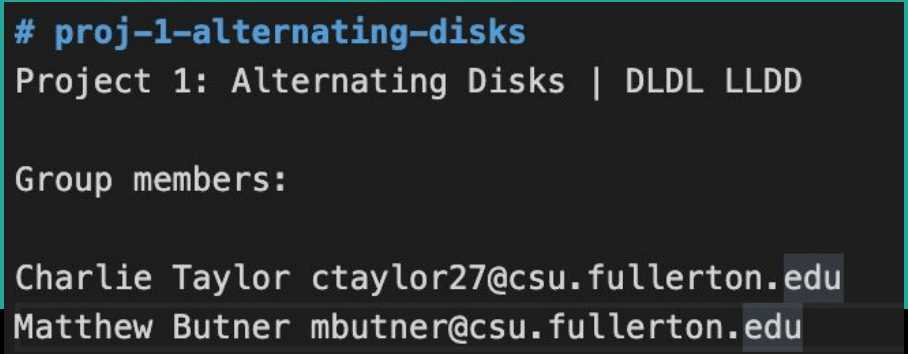
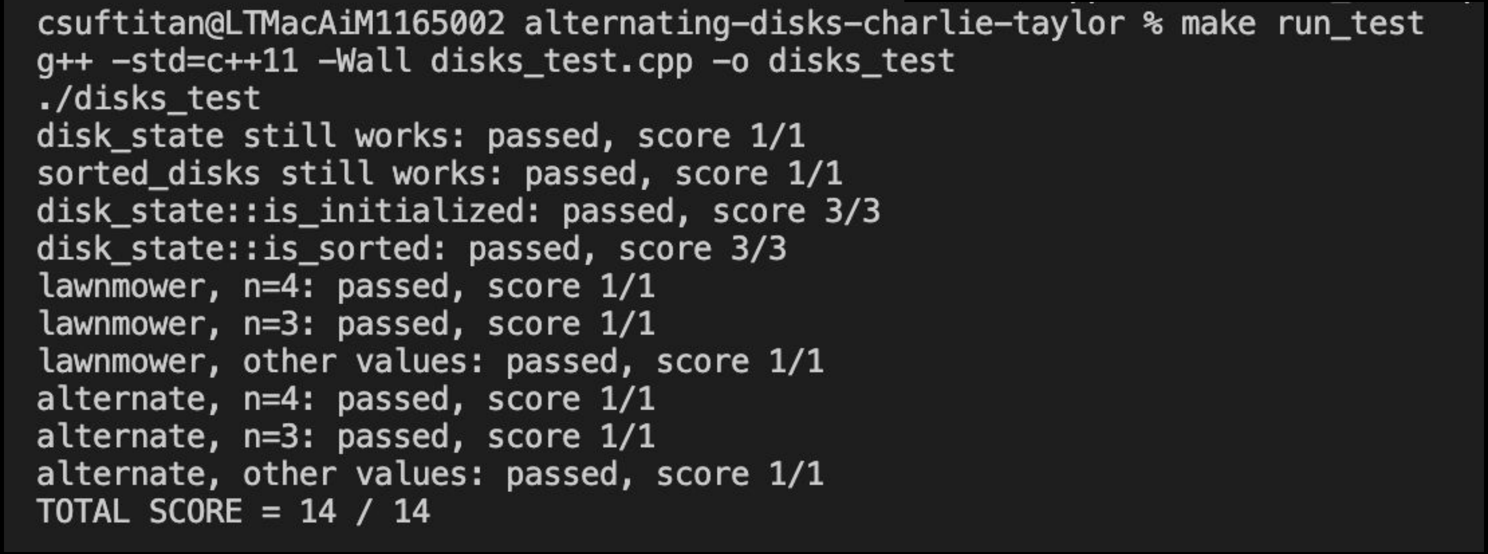
Project 1: Algorithms Implementation

By Matthew Butner and Charlie Taylor
ctaylor27@csu.fullerton.edu
mbutner@csu.fullerton.edu

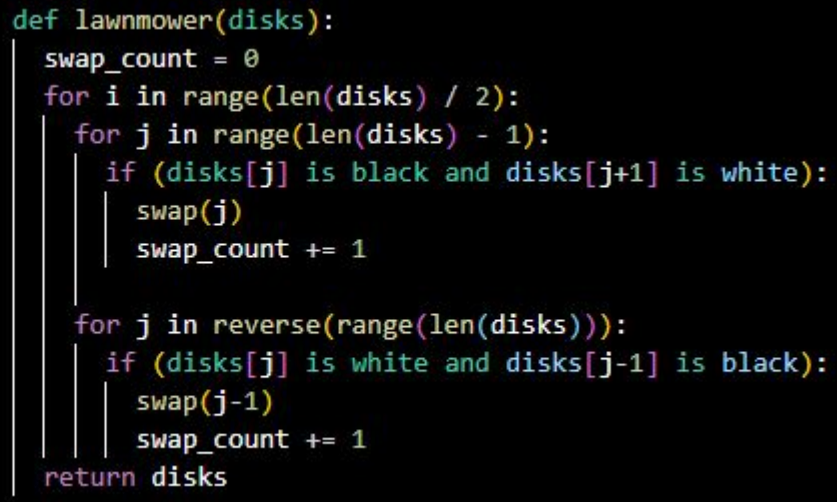


```
# proj-1-alternating-disks
Project 1: Alternating Disks | DLDL LLDD


Group members:


Charlie Taylor ctaylor27@csu.fullerton.edu
Matthew Butner mbutner@csu.fullerton.edu
```



```
csuftitan@LTMacAiM1165002 alternating-disks-charlie-taylor % make run_test
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
lawnmower, n=4: passed, score 1/1
lawnmower, n=3: passed, score 1/1
lawnmower, other values: passed, score 1/1
alternate, n=4: passed, score 1/1
alternate, n=3: passed, score 1/1
alternate, other values: passed, score 1/1
TOTAL SCORE = 14 / 14
```

Pseudocode (Lawnmower Sort)

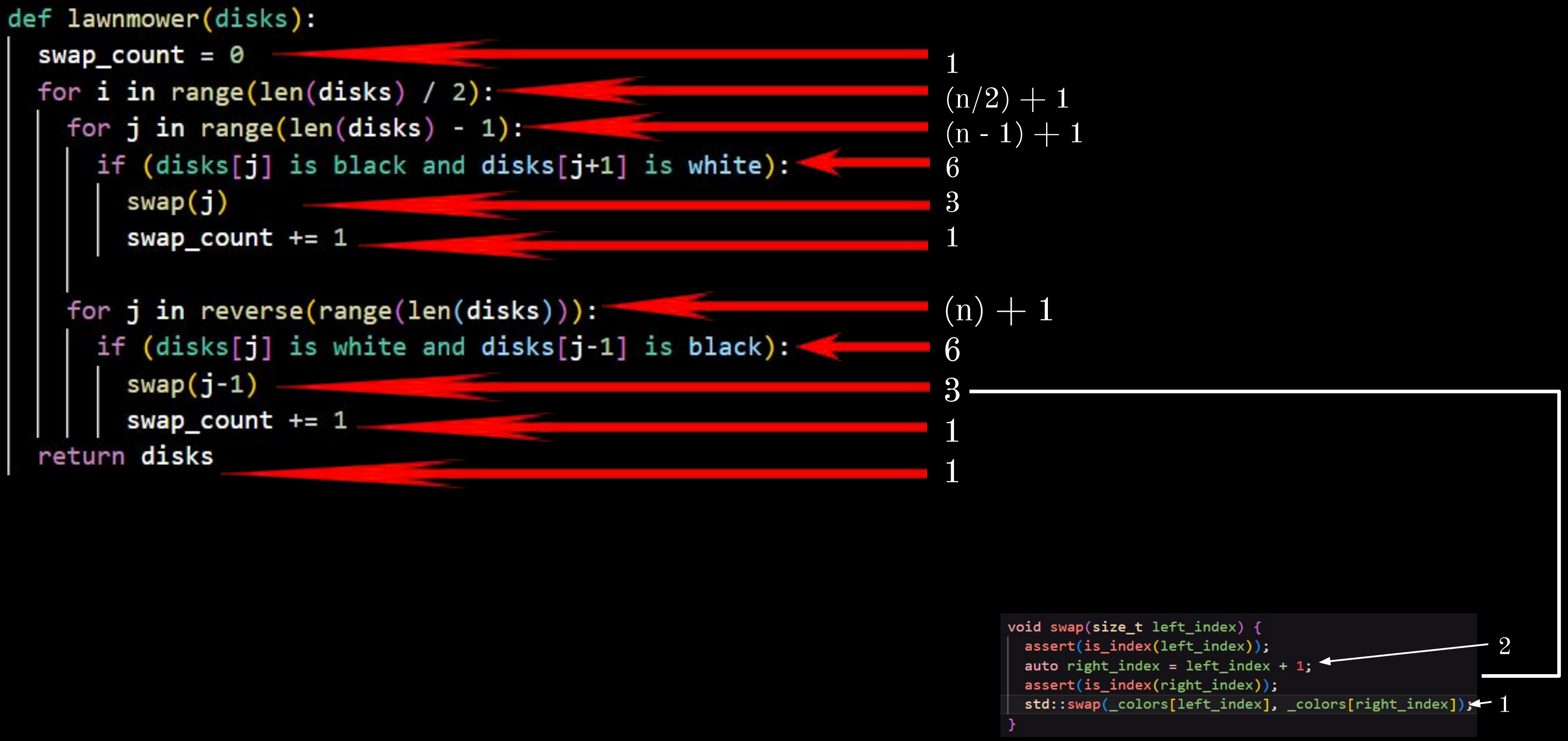


```
def lawnmower(disks):
    swap_count = 0
    for i in range(len(disks) / 2):
        for j in range(len(disks) - 1):
            if (disks[j] is black and disks[j+1] is white):
                swap(j)
                swap_count += 1

        for j in reverse(range(len(disks))):
            if (disks[j] is white and disks[j-1] is black):
                swap(j-1)
                swap_count += 1
    return disks
```

Pseudocode (Alternating Sort)

```
def alternate(disks):
  swap_count = 0
  for(i = 0; i < disks.dark_count() + 1; i++):
    for(j = i; j < len(disks) - 1; j += 2):
      if disks[j] is black and disks[j+1] is white:
        swap(j)
        swap_count += 1

  return disks
```

Efficiency Proof: Lawnmower



```
def lawnmower(disks):
  swap_count = 0
  for i in range(len(disks) / 2):
    for j in range(len(disks) - 1):
      if (disks[j] is black and disks[j+1] is white):
        swap(j)
        swap_count += 1

    for j in reverse(range(len(disks))):
      if (disks[j] is white and disks[j-1] is black):
        swap(j-1)
        swap_count += 1
  return disks
```

1

$(n/2) + 1$

$(n - 1) + 1$

6

3

1

$(n) + 1$

6

3

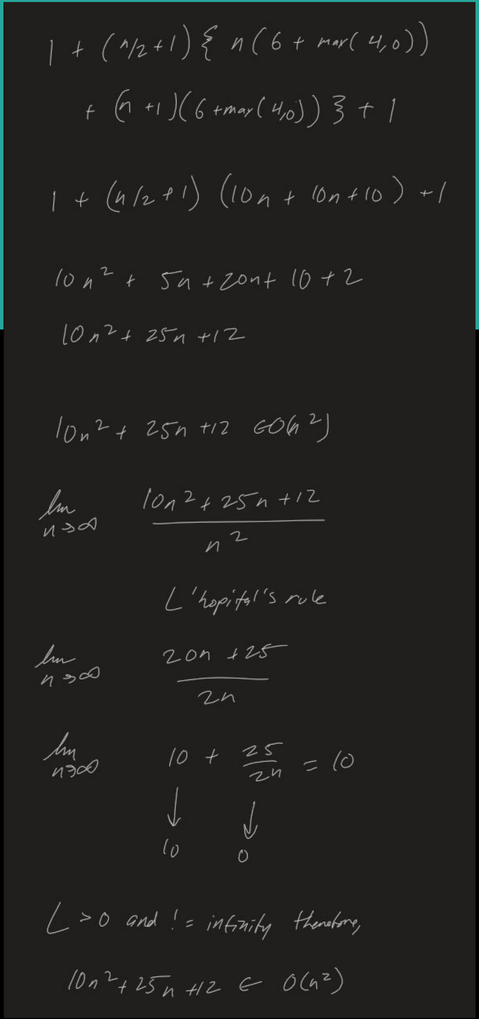
1

1

```
void swap(size_t left_index) {
  assert(is_index(left_index));
  auto right_index = left_index + 1;
  assert(is_index(right_index));
  std::swap(_colors[left_index], _colors[right_index]);
}
```

2

1



$$1 + (n/2+1)\{n(6+\max(4,0))$$

$$+ (n+1)(6+\max(4,0))\} + 1$$

$$1 + (n/2+1)(10n + 10n+10) + 1$$

$$10n^2 + 5n + 20n + 10 + 2$$

$$10n^2 + 25n + 12$$

$$10n^2 + 25n + 12 \in O(n^2)$$

$$\lim_{n \to \infty} \frac{10n^2 + 25n + 12}{n^2}$$

L'hopital's rule

$$\lim_{n \to \infty} \frac{20n + 25}{2n}$$

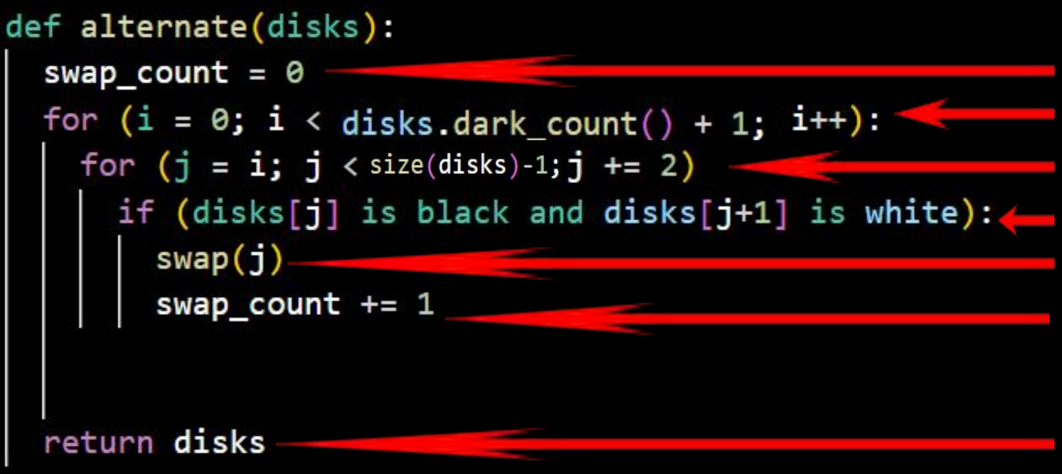
$$\lim_{n \to \infty} \underset{\substack{\downarrow \\ 10}}{10} + \underset{\substack{\downarrow \\ 0}}{\frac{25}{2n}} = 10$$

L > 0 and != infinity therefore,

$$10n^2 + 25n + 12 \in O(n^2)$$

Calculate the step count, and proves efficiency class using limit theorem

Efficiency Proof: Alternating



```
def alternate(disks):
  swap_count = 0
  for (i = 0; i < disks.dark_count() + 1; i++):
    for (j = i; j < size(disks)-1;j += 2)
      if (disks[j] is black and disks[j+1] is white):
        swap(j)
        swap_count += 1


  return disks
```

1

$(n/2) + 1$

$((n - 1) - i)/2 + 1$

6

3

1

1

1

(n/2 +1) * {(n-i) + 1*(5+max(3+1,0)

(n-i) + 1*(5+max

5

3

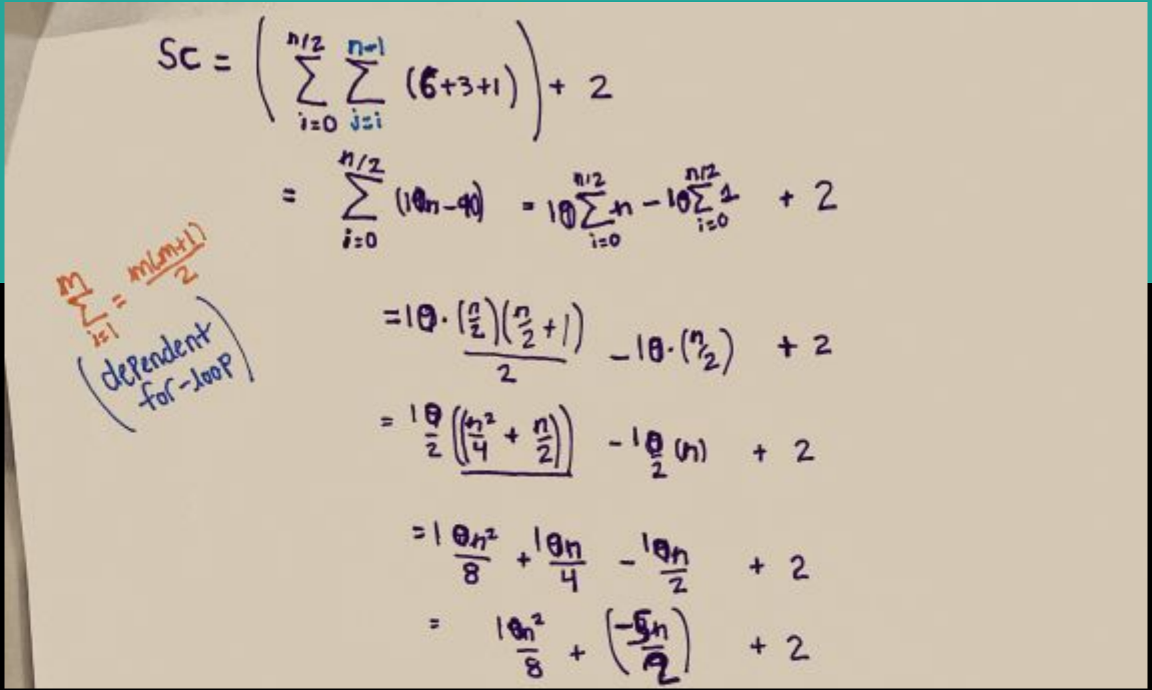
1

1

```
void swap(size_t left_index) {
  assert(is_index(left_index));
  auto right_index = left_index + 1;
  assert(is_index(right_index));
  std::swap(_colors[left_index], _colors[right_index]);
}
```

2

1



$$Sc = \left(\sum_{i=0}^{n/2} \sum_{j=i}^{n-1} (6+3+1) \right) + 2$$

$$= \sum_{i=0}^{n/2} (10n-40) = 10\sum_{i=0}^{n/2} n - 10\sum_{i=0}^{n/2} 1 + 2$$

$$\sum_{i=1}^{m} = \frac{m(m+1)}{2}$$

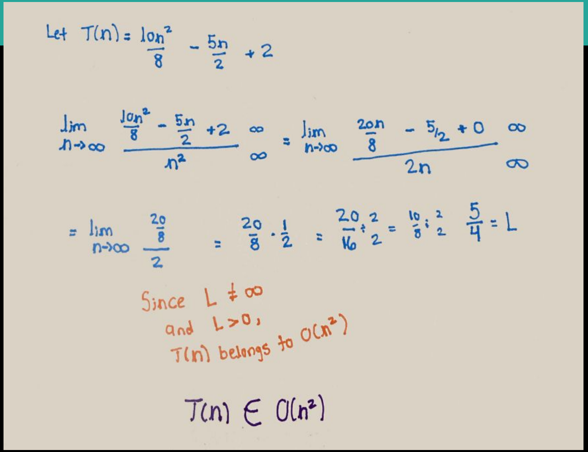
(dependent for-loop)

$$= 10 \cdot \frac{\left(\frac{n}{2}\right)\left(\frac{n}{2}+1\right)}{2} - 10 \cdot \left(\frac{n}{2}\right) + 2$$

$$= \frac{10}{2}\left(\left(\frac{n^2}{4} + \frac{n}{2}\right)\right) - \frac{10}{2}(n) + 2$$

$$= \frac{10n^2}{8} + \frac{10n}{4} - \frac{10n}{2} + 2$$

$$= \frac{10n^2}{8} + \left(\frac{-5n}{2}\right) + 2$$



Let $T(n) = \frac{10n^2}{8} - \frac{5n}{2} + 2$

$$\lim_{n \to \infty} \frac{\frac{10n^2}{8} - \frac{5n}{2} + 2}{n^2} \frac{\infty}{\infty} = \lim_{n \to \infty} \frac{\frac{20n}{8} - 5/2 + 0}{2n} \frac{\infty}{\infty}$$

$$= \lim_{n \to \infty} \frac{\frac{20}{8}}{2} = \frac{20}{8} \cdot \frac{1}{2} = \frac{20}{16} \div \frac{2}{2} = \frac{10}{8} \div \frac{2}{2} \; \frac{5}{4} = L$$

Since $L \neq \infty$
and $L > 0$,
$T(n)$ belongs to $O(n^2)$

$T(n) \in O(n^2)$