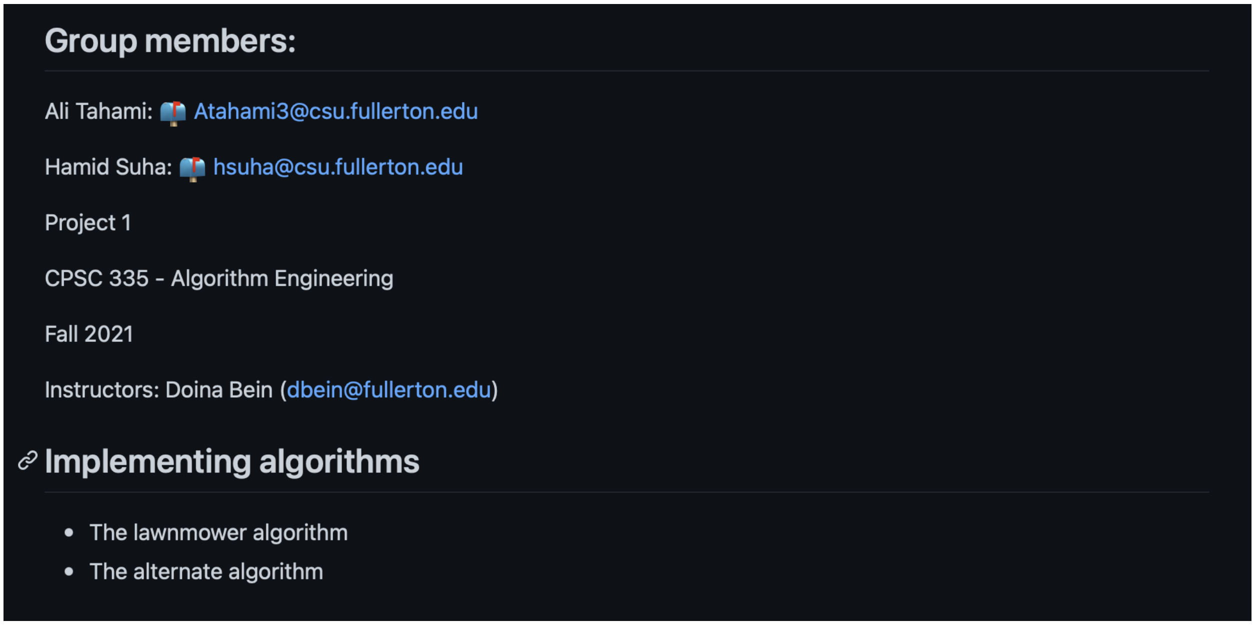
CPSC 335 - Project 1
Ali Tahami
atahami3@csu.fullerton.edu

Hamid Suha
hsuha@csu.fullerton.edu



Group members:

Ali Tahami: 📫 Atahami3@csu.fullerton.edu

Hamid Suha: 📫 hsuha@csu.fullerton.edu

Project 1

CPSC 335 - Algorithm Engineering

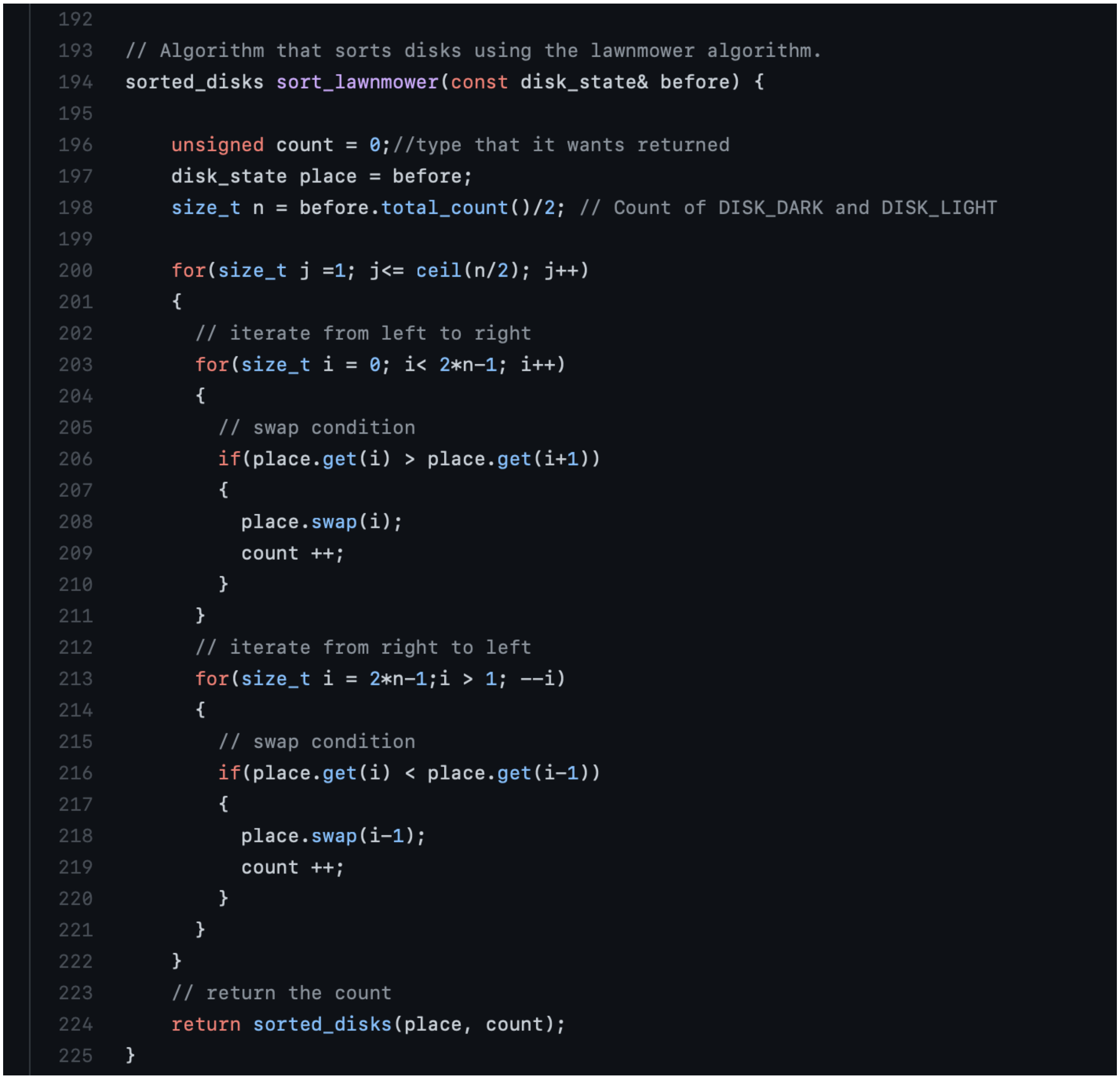
Fall 2021

Instructors: Doina Bein (dbein@fullerton.edu)

Implementing algorithms

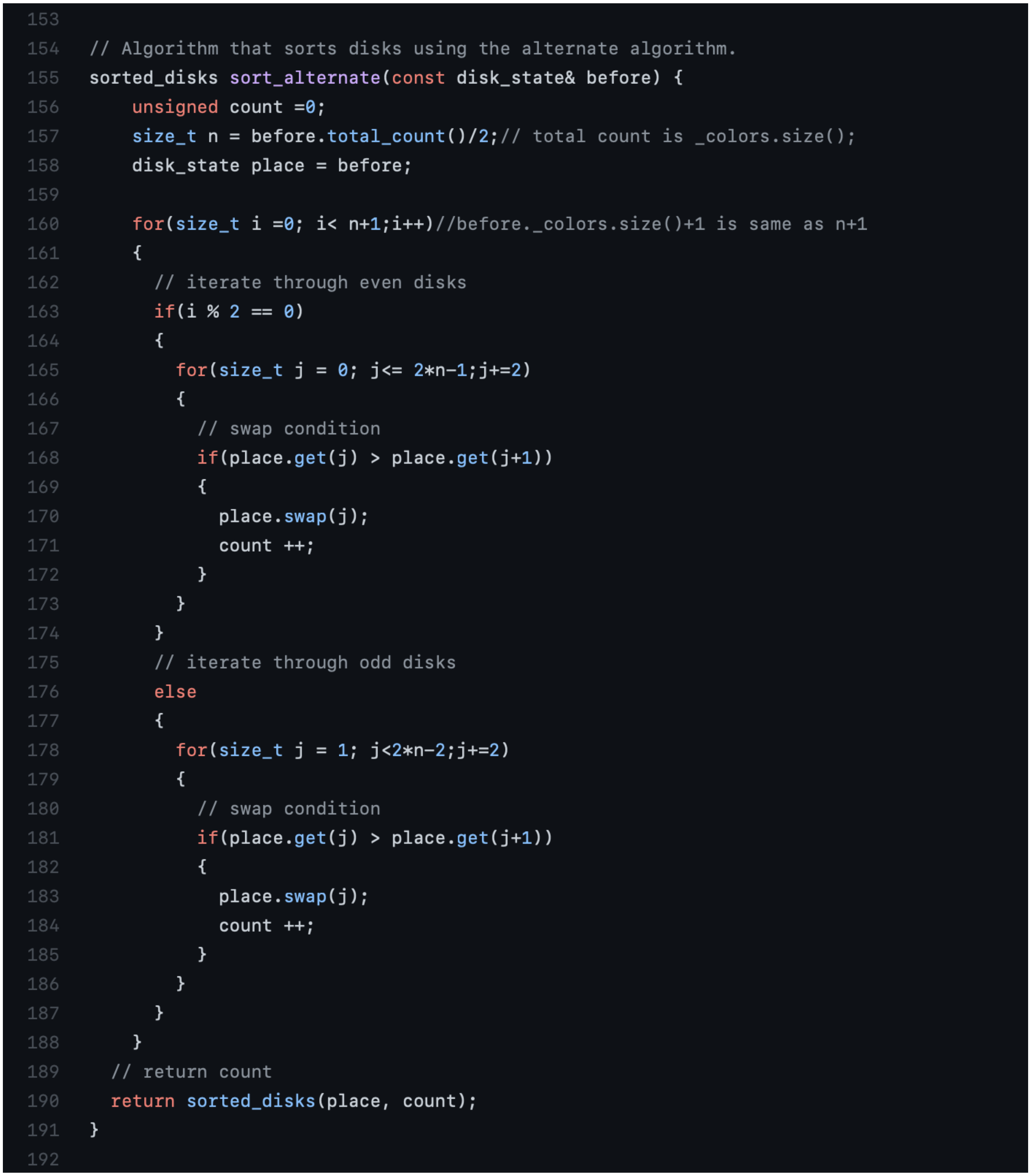
- The lawnmower algorithm
- The alternate algorithm

The lawnmower algorithm



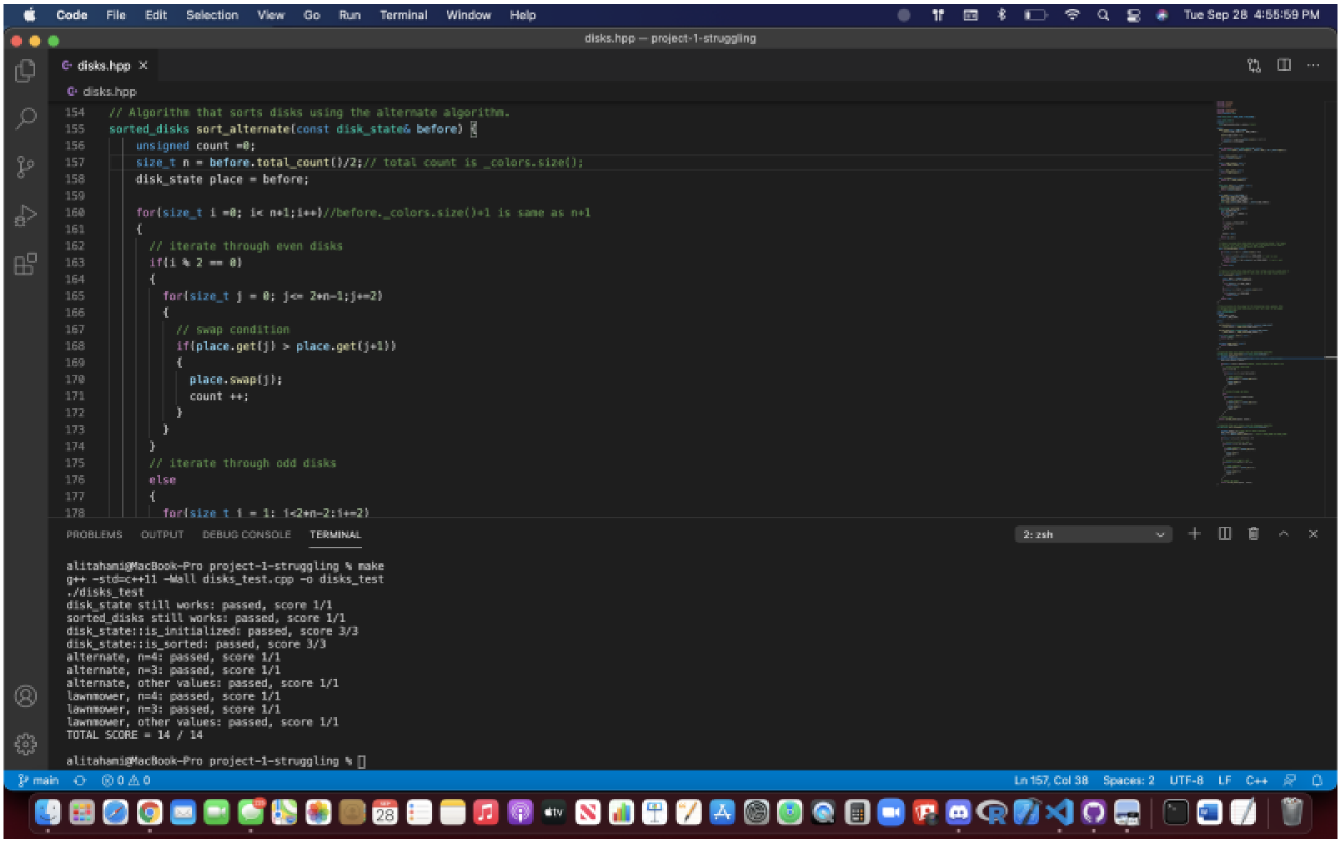
```cpp

// Algorithm that sorts disks using the lawnmower algorithm.
sorted_disks sort_lawnmower(const disk_state& before) {

    unsigned count = 0;//type that it wants returned
    disk_state place = before;
    size_t n = before.total_count()/2; // Count of DISK_DARK and DISK_LIGHT

    for(size_t j =1; j<= ceil(n/2); j++)
    {
      // iterate from left to right
      for(size_t i = 0; i< 2*n-1; i++)
      {
        // swap condition
        if(place.get(i) > place.get(i+1))
        {
          place.swap(i);
          count ++;
        }
      }
      // iterate from right to left
      for(size_t i = 2*n-1;i > 1; --i)
      {
        // swap condition
        if(place.get(i) < place.get(i-1))
        {
          place.swap(i-1);
          count ++;
        }
      }
    }
    // return the count
    return sorted_disks(place, count);
}
```

The alternate algorithm



```cpp
// Algorithm that sorts disks using the alternate algorithm.
sorted_disks sort_alternate(const disk_state& before) {
    unsigned count =0;
    size_t n = before.total_count()/2;// total count is _colors.size();
    disk_state place = before;

    for(size_t i =0; i< n+1;i++)//before._colors.size()+1 is same as n+1
    {
      // iterate through even disks
      if(i % 2 == 0)
      {
        for(size_t j = 0; j<= 2*n-1;j+=2)
        {
          // swap condition
          if(place.get(j) > place.get(j+1))
          {
            place.swap(j);
            count ++;
          }
        }
      }
      // iterate through odd disks
      else
      {
        for(size_t j = 1; j<2*n-2;j+=2)
        {
          // swap condition
          if(place.get(j) > place.get(j+1))
          {
            place.swap(j);
            count ++;
          }
        }
      }
    }
  // return count
  return sorted_disks(place, count);
}
```

Code execution

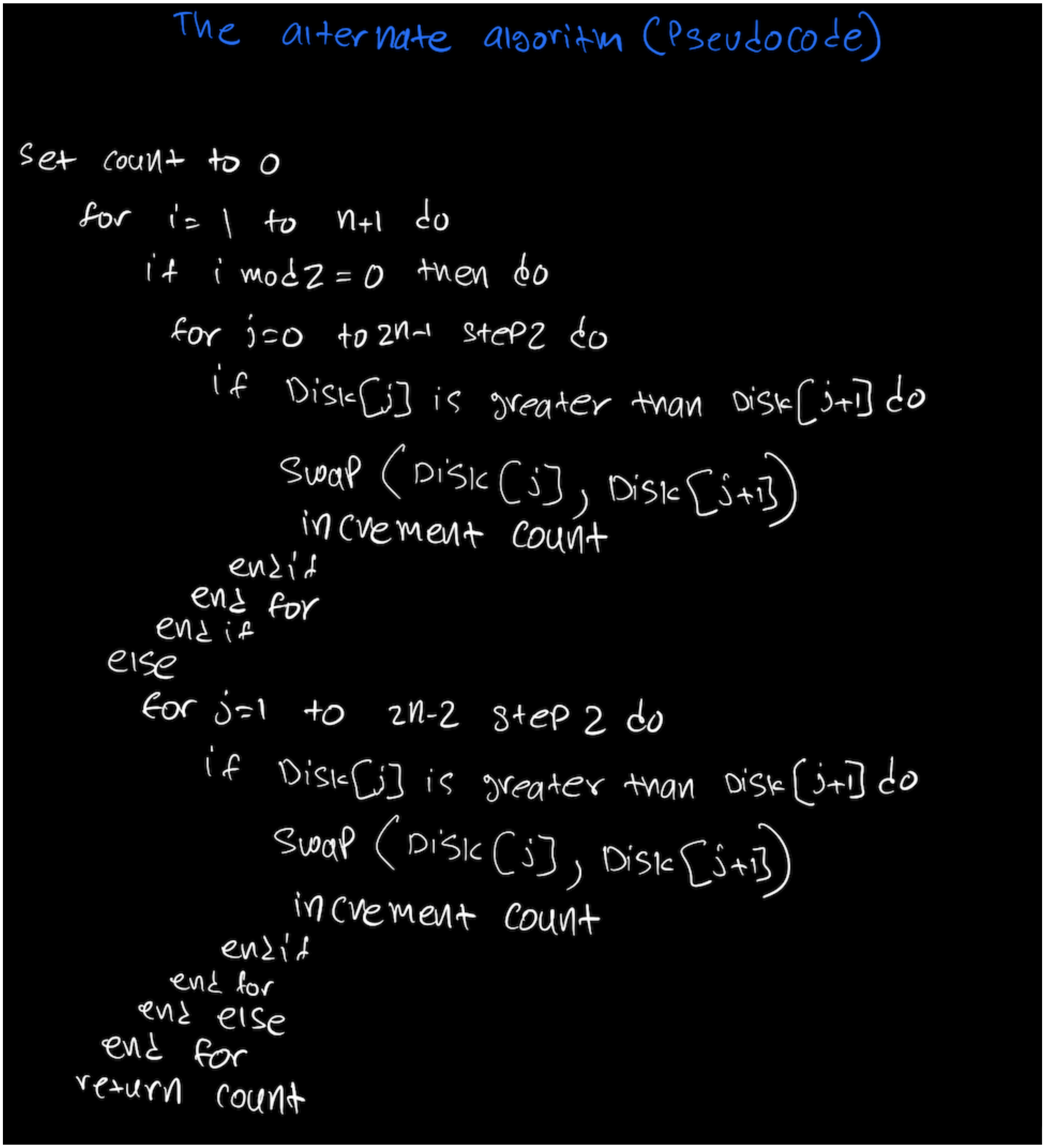


Code File Edit Selection View Go Run Terminal Window Help
Tue Sep 28 4:55:59 PM
disks.hpp — project-1-struggling
disks.hpp
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: zsh
main
⊗ 0 ⚠ 0
Ln 157, Col 38 Spaces: 2 UTF-8 LF C++

```cpp
// Algorithm that sorts disks using the alternate algorithm.
sorted_disks sort_alternate(const disk_state& before) {
    unsigned count =0;
    size_t n = before.total_count()/2;// total count is _colors.size();
    disk_state place = before;

    for(size_t i =0; i< n+1;i++)//before._colors.size()+1 is same as n+1
    {
      // iterate through even disks
      if(i % 2 == 0)
      {
        for(size_t j = 0; j<= 2*n-1;j+=2)
        {
          // swap condition
          if(place.get(j) > place.get(j+1))
          {
            place.swap(j);
            count ++;
          }
        }
      }
      // iterate through odd disks
      else
      {
        for(size_t i = 1; i<2*n-2;i+=2)
```

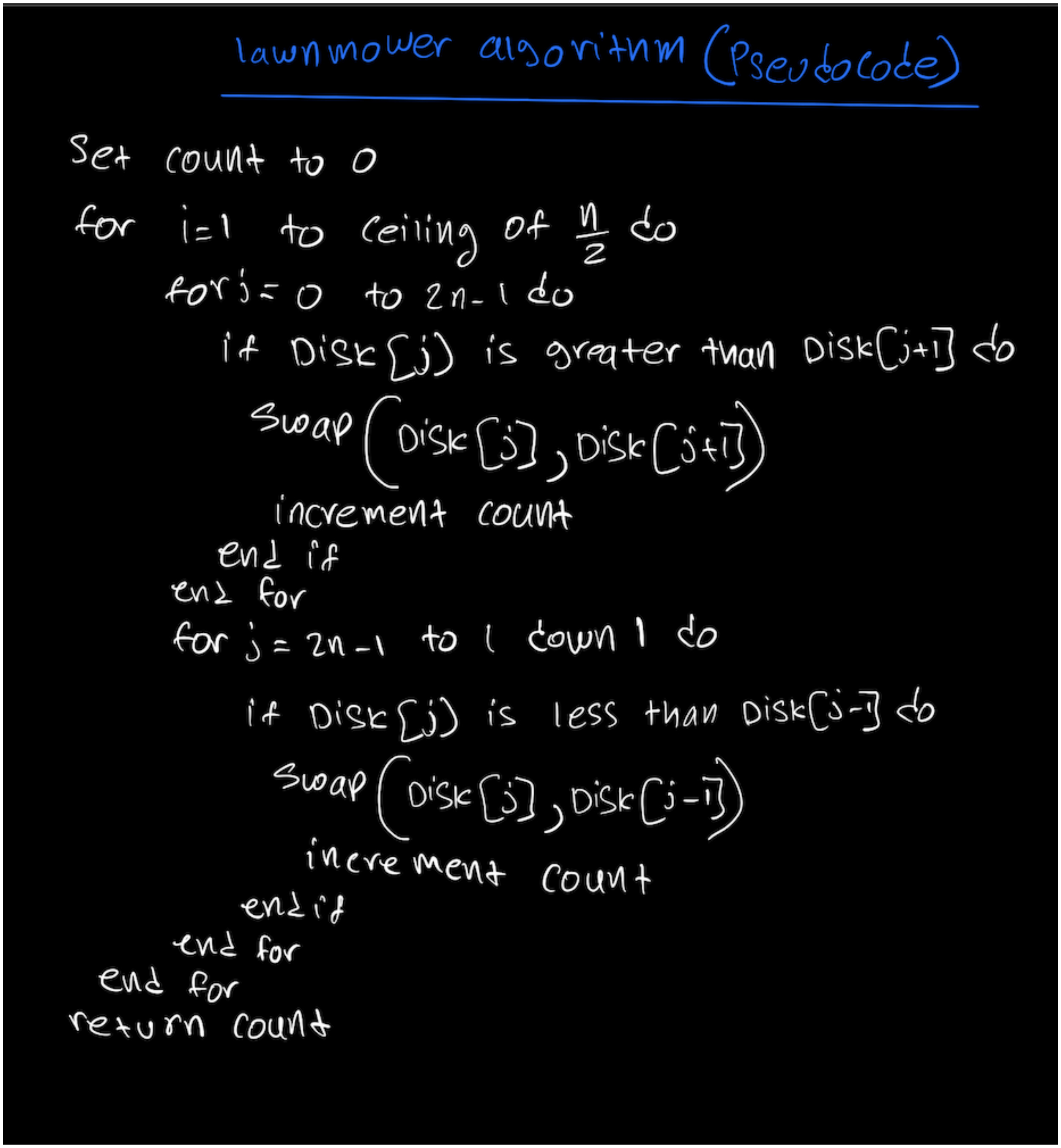
```
alitahami@MacBook-Pro project-1-struggling % make
g++ -std=c++11 -Wall disks_test.cpp -o disks_test
./disks_test
disk_state still works: passed, score 1/1
sorted_disks still works: passed, score 1/1
disk_state::is_initialized: passed, score 3/3
disk_state::is_sorted: passed, score 3/3
alternate, 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

alitahami@MacBook-Pro project-1-struggling % []
```

Pseudocode



The alternate algoritm (Pseudocode)

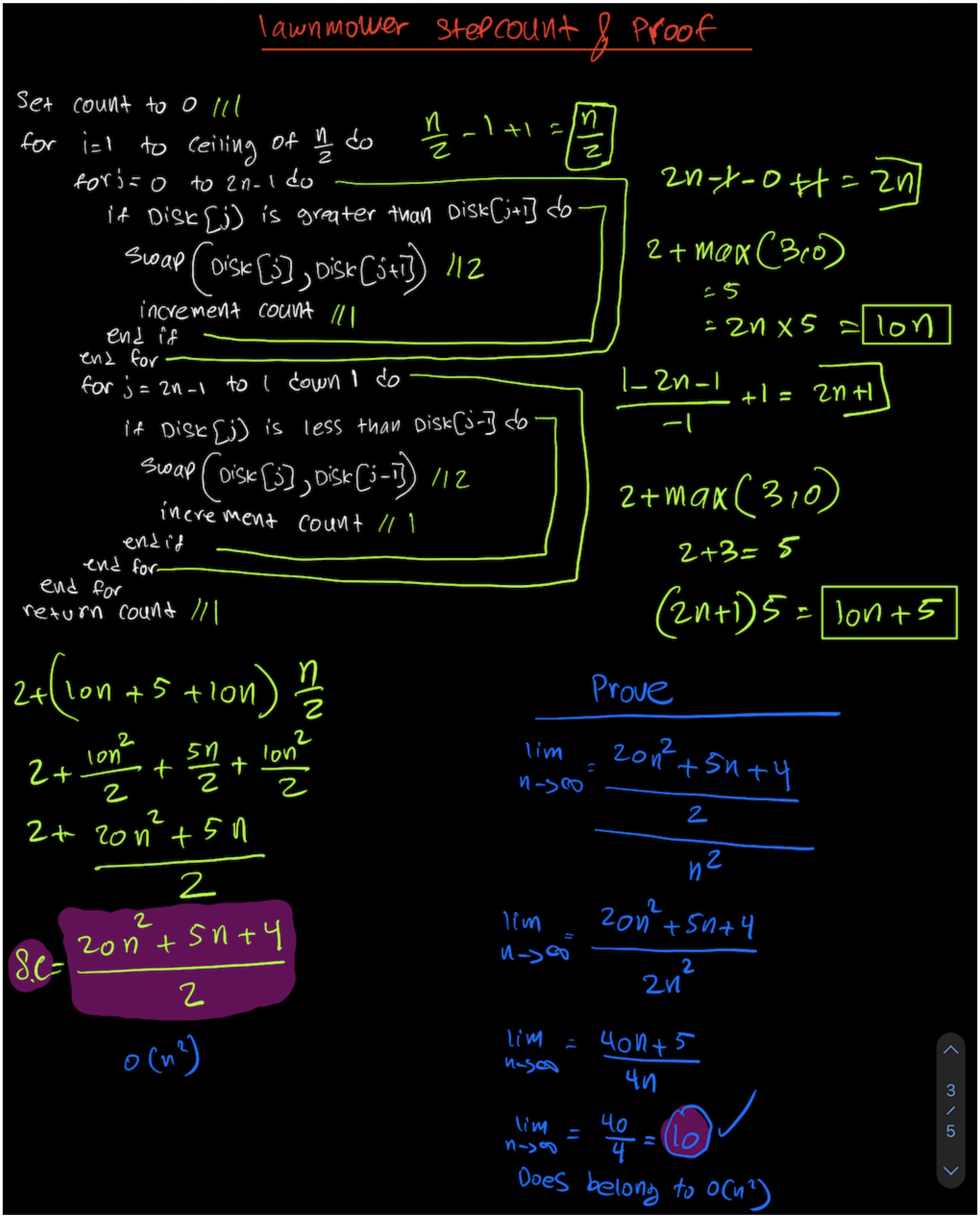
```
Set count to 0
    for i=1 to n+1 do
        if i mod2 = 0 then do
          for j=0 to 2n-1 step2 do
            if Disk[j] is greater than Disk[j+1] do
                swap (Disk(j], Disk[j+1])
                increment count
            endif
          end for
        end if
      else
        for j=1 to 2n-2 step 2 do
            if Disk[j] is greater than Disk[j+1] do
                swap (Disk(j], Disk[j+1])
                increment count
            endif
          end for
        end else
      end for
    return count
```



lawnmower algorithm (Pseudocode)

```
Set count to 0
for i=1 to ceiling of $\frac{n}{2}$ do
    for j=0 to 2n-1 do
        if Disk[j) is greater than Disk[j+1] do
            Swap(Disk[j], Disk[j+1])
            increment count
        end if
    end for
    for j = 2n-1 to 1 down 1 do
        if Disk[j) is less than Disk[j-1] do
            Swap(Disk[j], Disk[j-1])
            increment count
        end if
    end for
end for
return count
```

Step Count’s and proof for lawnmower



lawnmower stepcount & Proof

```
Set count to 0 //1
for i=1 to ceiling of n/2 do
    for j=0 to 2n-1 do
        if Disk[j] is greater than Disk[j+1] do
            swap(Disk[j], Disk[j+1]) //2
            increment count //1
        end if
    end for
    for j = 2n-1 to 1 down 1 do
        if Disk[j] is less than Disk[j-1] do
            swap(Disk[j], Disk[j-1]) //2
            increment count //1
        end if
    end for
end for
return count //1
```

$\frac{n}{2}-1+1=\boxed{\frac{n}{2}}$

$2n-\cancel{1}-0+\cancel{1}=\boxed{2n}$

$2+\max(3,0)$

$=5$

$=2n\times 5=\boxed{10n}$

$\frac{1-2n-1}{-1}+1=\boxed{2n+1}$

$2+\max(3,0)$

$2+3=5$

$(2n+1)5=\boxed{10n+5}$

$2+(10n+5+10n)\frac{n}{2}$

$2+\frac{10n^2}{2}+\frac{5n}{2}+\frac{10n^2}{2}$

$2+\frac{20n^2+5n}{2}$

$S.C=\boxed{\frac{20n^2+5n+4}{2}}$

$O(n^2)$

Prove

$\lim_{n\to\infty}=\frac{\frac{20n^2+5n+4}{2}}{n^2}$

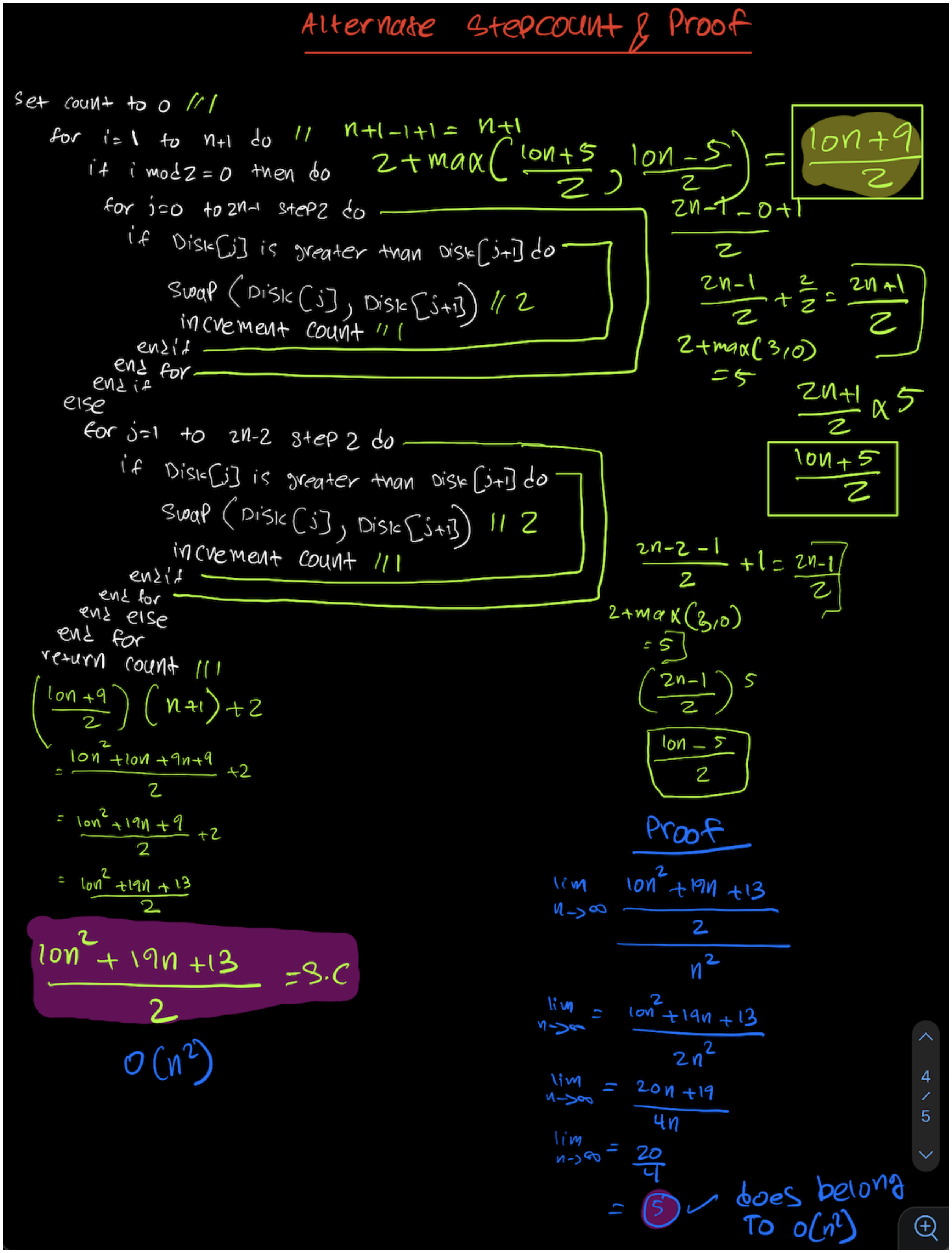
$\lim_{n\to\infty}=\frac{20n^2+5n+4}{2n^2}$

$\lim_{n\to\infty}=\frac{40n+5}{4n}$

$\lim_{n\to\infty}=\frac{40}{4}=10$ ✓

Does belong to $O(n^2)$

Step count and proof for alternate



Alternate Stepcount & Proof

```
Set count to 0 //1
  for i=1 to n+1 do //  n+1-1+1 = n+1
    if i mod2 = 0 then do
      for j=0 to 2n-1 step2 do
        if Disk[j] is greater than Disk[j+1] do
          Swap (Disk[j], Disk[j+1]) // 2
          incvement count //1
        endif
      end for
    end if
  else
    for j=1 to 2n-2 step 2 do
      if Disk[j] is greater than Disk[j+1] do
        Swap (Disk[j], Disk[j+1]) // 2
        incvement count //1
      endif
    end for
  end else
end for
return count //1
```

$$2+\max\left(\frac{10n+5}{2}, \frac{10n-5}{2}\right) = \boxed{\frac{10n+9}{2}}$$

$$\frac{2n-1-0+1}{2}$$

$$\frac{2n-1}{2}+\frac{2}{2}=\frac{2n+1}{2}$$

$$2+\max(3,0)$$
$$=5$$

$$\frac{2n+1}{2}\times 5$$

$$\boxed{\frac{10n+5}{2}}$$

$$\frac{2n-2-1}{2}+1=\frac{2n-1}{2}$$

$$2+\max(3,0)$$
$$=5$$

$$\left(\frac{2n-1}{2}\right)5$$

$$\boxed{\frac{10n-5}{2}}$$

$$\left(\frac{10n+9}{2}\right)(n+1)+2$$

$$=\frac{10n^2+10n+9n+9}{2}+2$$

$$=\frac{10n^2+19n+9}{2}+2$$

$$=\frac{10n^2+19n+13}{2}$$

$$\frac{10n^2+19n+13}{2} = S.C$$

$O(n^2)$

Proof

$$\lim_{n\to\infty}\frac{\frac{10n^2+19n+13}{2}}{n^2}$$

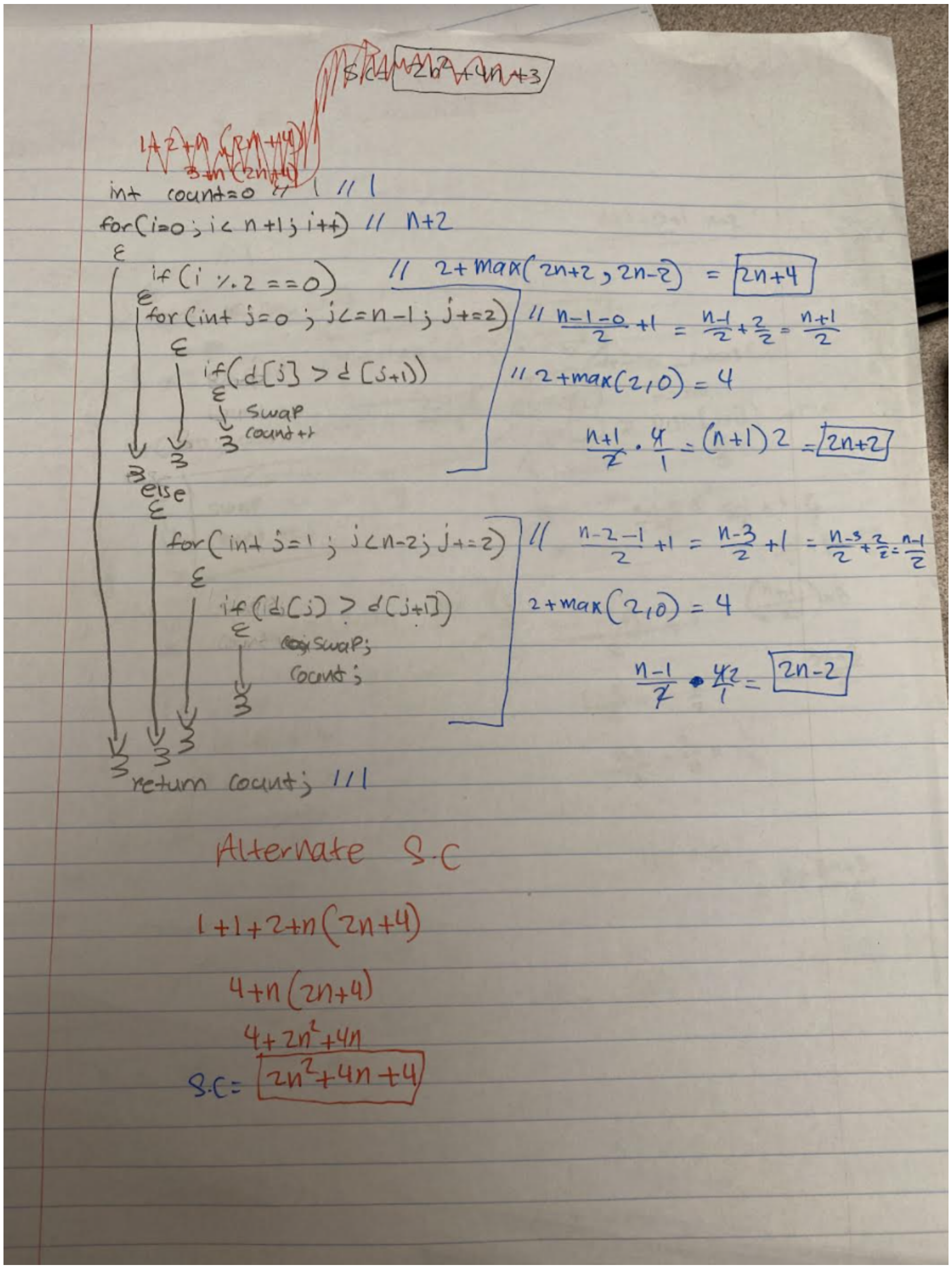
$$\lim_{n\to\infty} = \frac{10n^2+19n+13}{2n^2}$$

$$\lim_{n\to\infty} = \frac{20n+19}{4n}$$

$$\lim_{n\to\infty} = \frac{20}{4}$$

$= 5$ ✓ does belong To $O(n^2)$

images posted by mistake on Discord



~~1+2+n(2n+4)~~
~~3+n(2n+4)~~
~~S.C = $2n^2+4n+3$~~

```
int count=0 // 1 // 1
for(i=0; i< n+1; i++) // n+2
{
    if(i %2 ==0)   // 2+max(2n+2, 2n-2) = [2n+4]
    {
    for(int j=0; j<=n-1; j+=2)   // (n-1-0)/2 +1 = (n-1)/2 + 2/2 = (n+1)/2
      {
        if(d[j] > d[j+1])   // 2+max(2,0) = 4
        {
            swap
            count++
        }                    (n+1)/2 · 4/1 = (n+1)2 = [2n+2]
      }
    }
    else
    {
    for(int j=1; j<n-2; j+=2)   // (n-2-1)/2 +1 = (n-3)/2 +1 = (n-3)/2 + 2/2 = (n-1)/2
      {
        if(d[j] > d[j+1])   2+max(2,0) = 4
        {
            swap;
            count;
        }                    (n-1)/2 · 4 2/1 = [2n-2]
      }
    }
}
return count; // 1
```

Alternate S.C

$1+1+2+n(2n+4)$

$4+n(2n+4)$

$4+2n^2+4n$

S.C = $\boxed{2n^2+4n+4}$