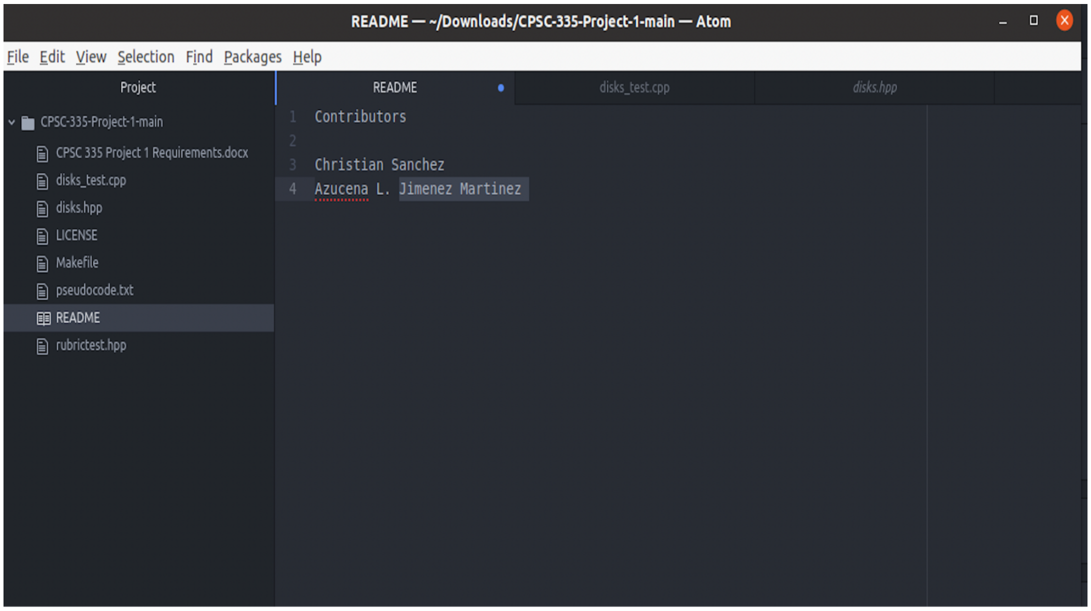
<u>Group Members:</u>

Christian Sanchez **<u>Email:</u>** <u>christiansanchez1258@csu.fullerton.edu</u>
Azucena Lizbeth Jimenez Martinez **<u>Email:</u>** <u>azucenaljim1@csu.fullerton.edu</u>

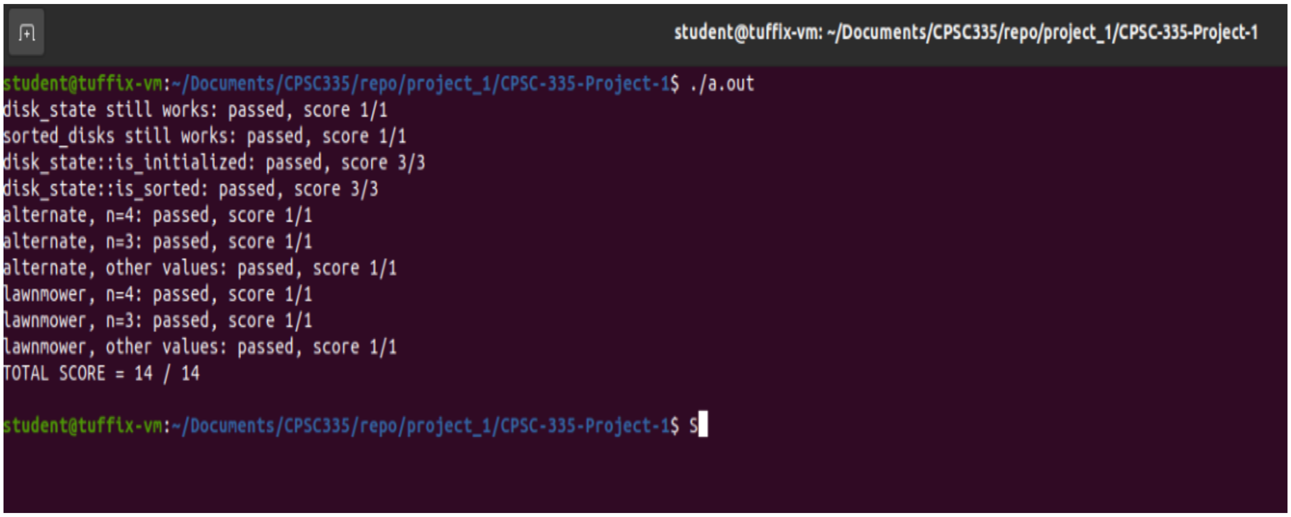
CPSC 335 Project 1 - implementing algorithms Submission

Project 1 was completed using both Tuffix and VS code, therefore screenshot submissions are included for both.

<u>Tuffix</u>

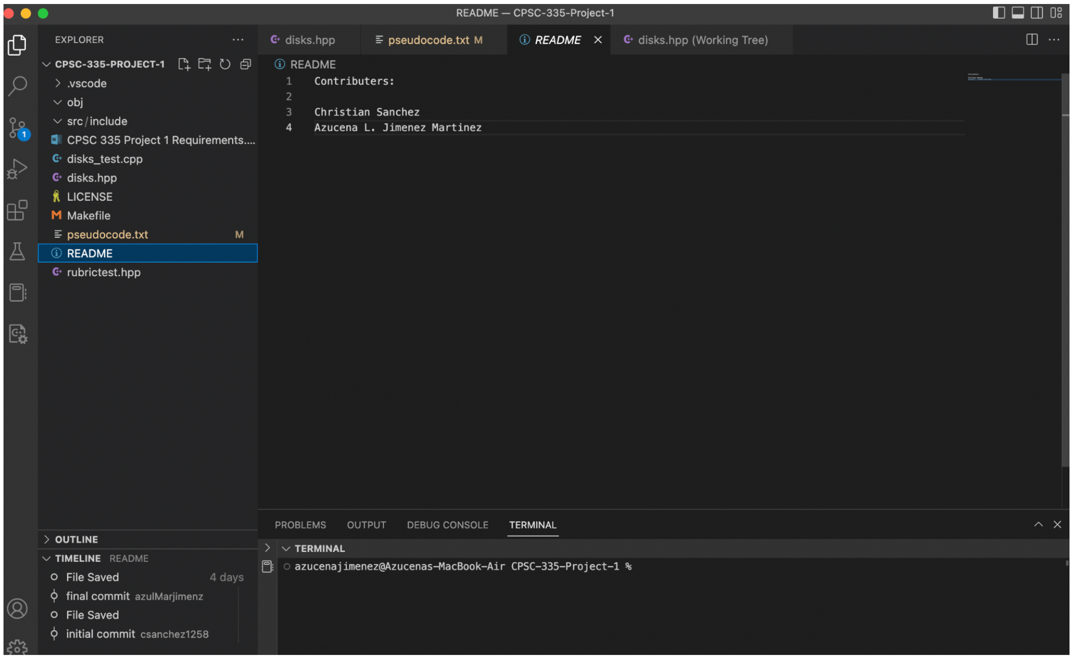


```
Contributors

Christian Sanchez
Azucena L. Jimenez Martinez
```


student@tuffix-vm: ~/Documents/CPSC335/repo/project_1/CPSC-335-Project-1

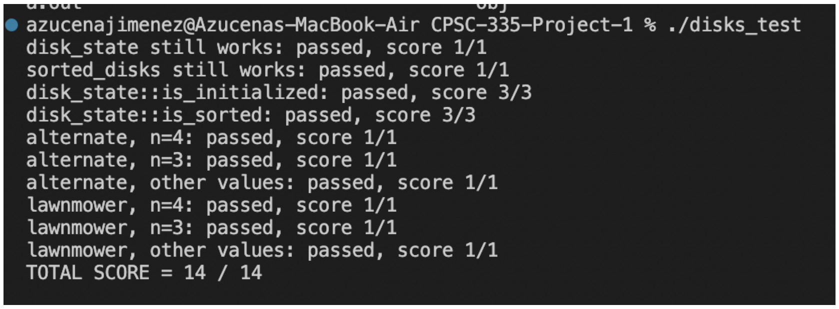
```
student@tuffix-vm:~/Documents/CPSC335/repo/project_1/CPSC-335-Project-1$ ./a.out
disk_state still works: passed, score 1/1
sorted_disks still works: passed, score 1/1
disk_state::is_initialized: passed, score 3/3
disk_state::is_sorted: passed, score 3/3
alternate, n=4: passed, score 1/1
alternate, 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:~/Documents/CPSC335/repo/project_1/CPSC-335-Project-1$ $
```

Visual Studio Code:



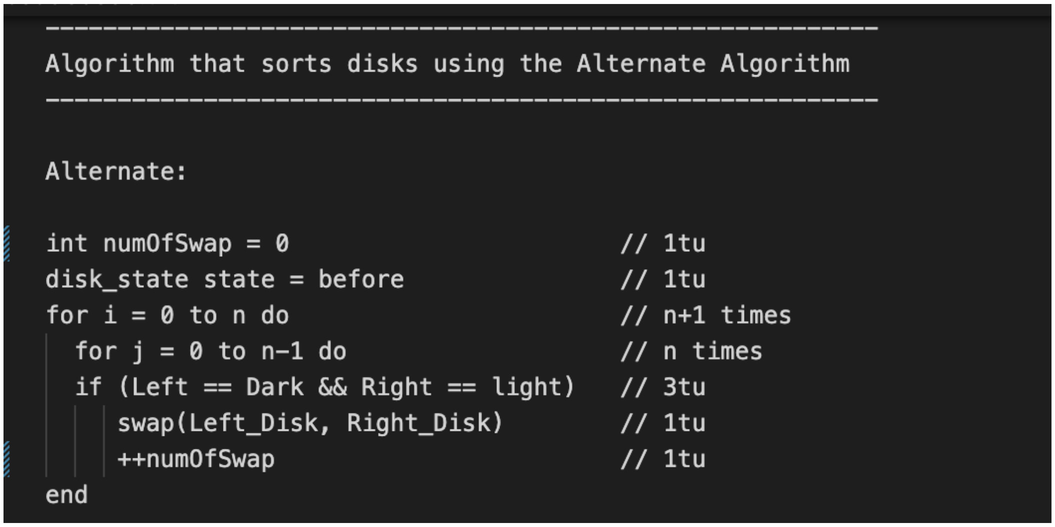
```
README
Contributers:

Christian Sanchez
Azucena L. Jimenez Martinez
```

```
TERMINAL
azucenajimenez@Azucenas-MacBook-Air CPSC-335-Project-1 %
```

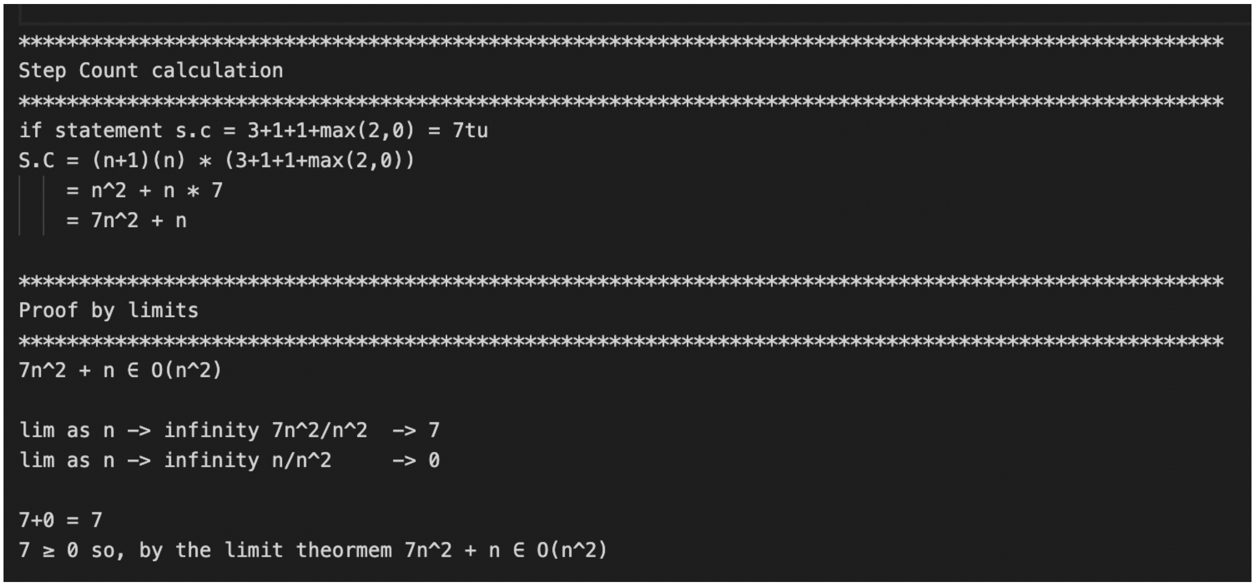


```
azucenajimenez@Azucenas-MacBook-Air CPSC-335-Project-1 % ./disks_test
disk_state still works: passed, score 1/1
sorted_disks still works: passed, score 1/1
disk_state::is_initialized: passed, score 3/3
disk_state::is_sorted: passed, score 3/3
alternate, n=4: passed, score 1/1
alternate, n=3: passed, 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
```

The alternate algorithm - Pseudocode listing + Step count + Proof

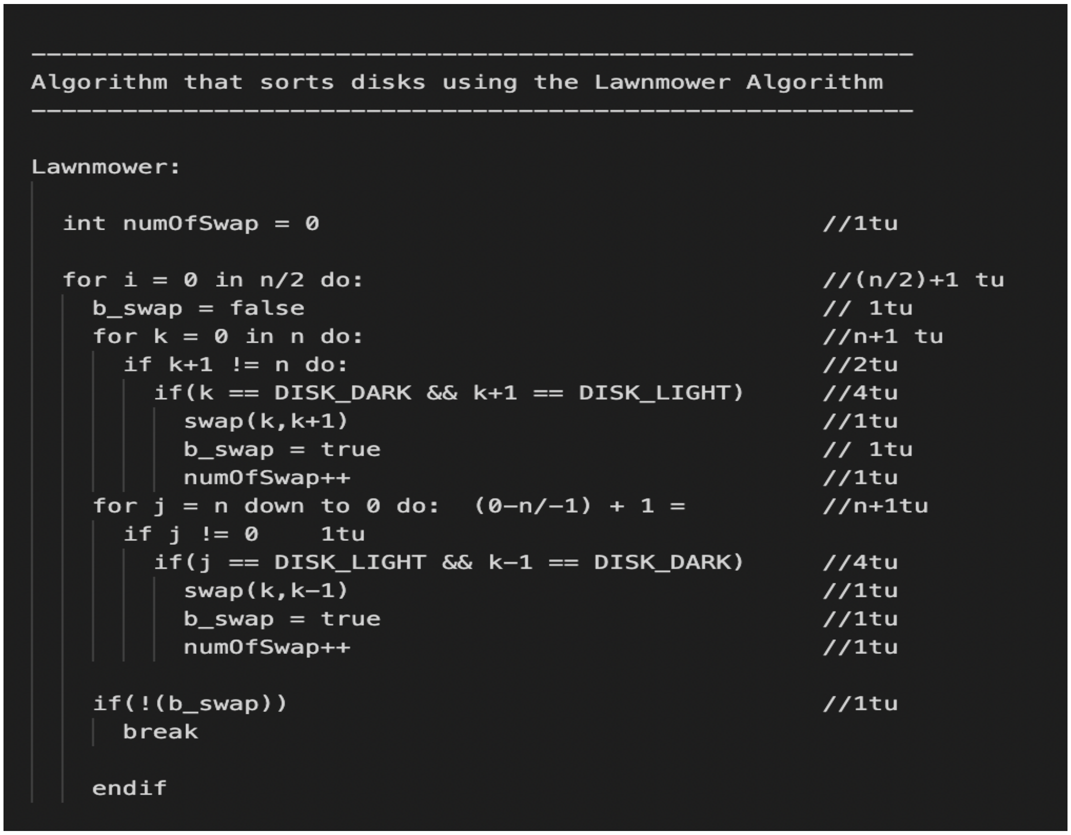


```
----------------------------------------------------------
Algorithm that sorts disks using the Alternate Algorithm
----------------------------------------------------------

Alternate:

int numOfSwap = 0                          // 1tu
disk_state state = before                  // 1tu
for i = 0 to n do                          // n+1 times
  for j = 0 to n-1 do                      // n times
  if (Left == Dark && Right == light)      // 3tu
     swap(Left_Disk, Right_Disk)           // 1tu
     ++numOfSwap                           // 1tu
end
```

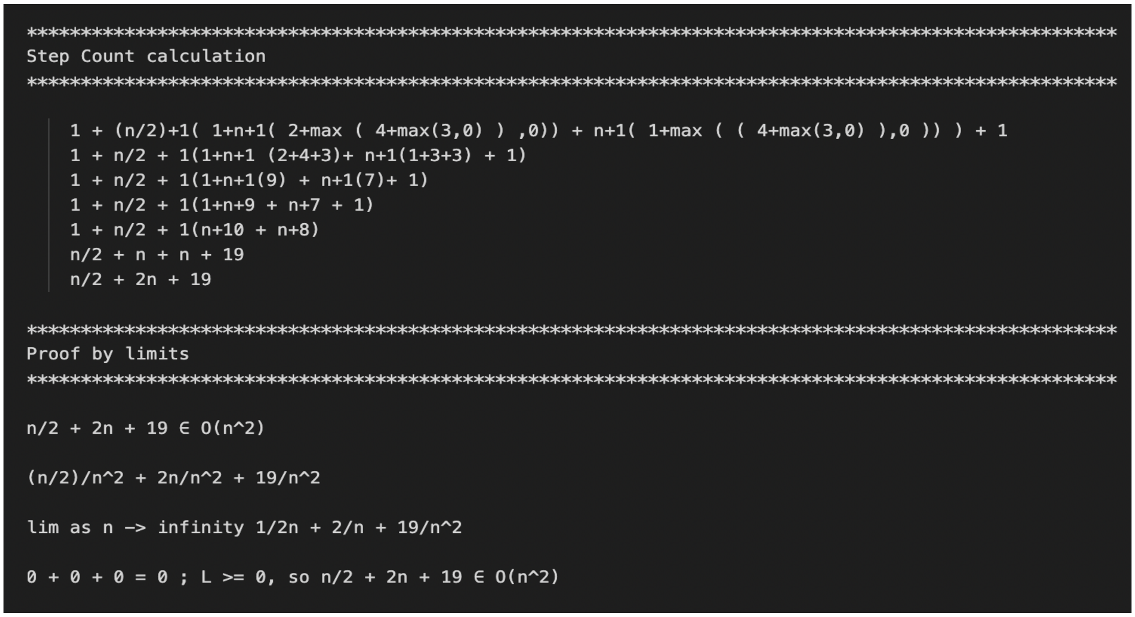


```
****************************************************************************************************
Step Count calculation
****************************************************************************************************
if statement s.c = 3+1+1+max(2,0) = 7tu
S.C = (n+1)(n) * (3+1+1+max(2,0))
    = n^2 + n * 7
    = 7n^2 + n

****************************************************************************************************
Proof by limits
****************************************************************************************************
7n^2 + n ∈ O(n^2)

lim as n -> infinity 7n^2/n^2  -> 7
lim as n -> infinity n/n^2     -> 0

7+0 = 7
7 ≥ 0 so, by the limit theormem 7n^2 + n ∈ O(n^2)
```

The lawnmower algorithm - Pseudocode listing + Step count + Proof



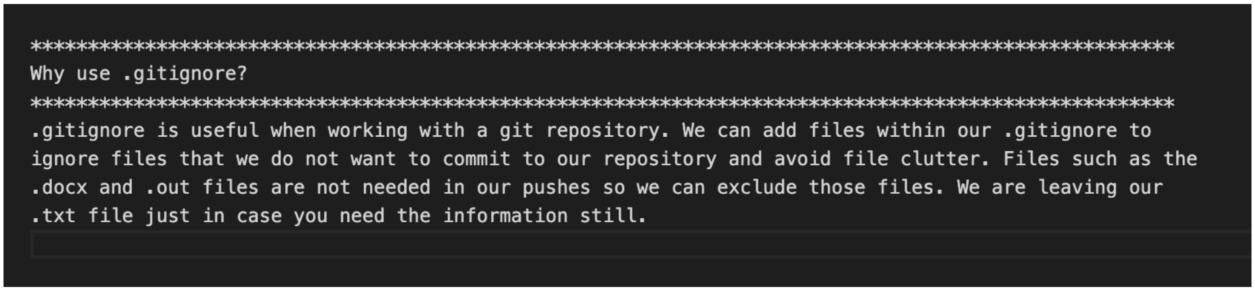
```
----------------------------------------------------------
Algorithm that sorts disks using the Lawnmower Algorithm
----------------------------------------------------------

Lawnmower:

  int numOfSwap = 0                                 //1tu

  for i = 0 in n/2 do:                              //(n/2)+1 tu
    b_swap = false                                  // 1tu
    for k = 0 in n do:                              //n+1 tu
      if k+1 != n do:                               //2tu
        if(k == DISK_DARK && k+1 == DISK_LIGHT)     //4tu
          swap(k,k+1)                               //1tu
          b_swap = true                             // 1tu
          numOfSwap++                               //1tu
    for j = n down to 0 do:  (0−n/−1) + 1 =         //n+1tu
      if j != 0    1tu
        if(j == DISK_LIGHT && k−1 == DISK_DARK)     //4tu
          swap(k,k−1)                               //1tu
          b_swap = true                             //1tu
          numOfSwap++                               //1tu

    if(!(b_swap))                                   //1tu
      break

    endif
```



```
****************************************************************************************************
Step Count calculation
****************************************************************************************************

    1 + (n/2)+1( 1+n+1( 2+max ( 4+max(3,0) ) ,0)) + n+1( 1+max ( ( 4+max(3,0) ),0 )) ) + 1
    1 + n/2 + 1(1+n+1 (2+4+3)+ n+1(1+3+3) + 1)
    1 + n/2 + 1(1+n+1(9) + n+1(7)+ 1)
    1 + n/2 + 1(1+n+9 + n+7 + 1)
    1 + n/2 + 1(n+10 + n+8)
    n/2 + n + n + 19
    n/2 + 2n + 19

****************************************************************************************************
Proof by limits
****************************************************************************************************

n/2 + 2n + 19 ∈ O(n^2)

(n/2)/n^2 + 2n/n^2 + 19/n^2

lim as n -> infinity 1/2n + 2/n + 19/n^2

0 + 0 + 0 = 0 ; L >= 0, so n/2 + 2n + 19 ∈ O(n^2)
```

(pseudocode + step counts + proofs are written inside the pseudocode.txt file)

.gitignore file explanation:



```
****************************************************************************************************
Why use .gitignore?
****************************************************************************************************
.gitignore is useful when working with a git repository. We can add files within our .gitignore to
ignore files that we do not want to commit to our repository and avoid file clutter. Files such as the
.docx and .out files are not needed in our pushes so we can exclude those files. We are leaving our
.txt file just in case you need the information still.
```

(This explanation is also found in the pseudocode.txt file)