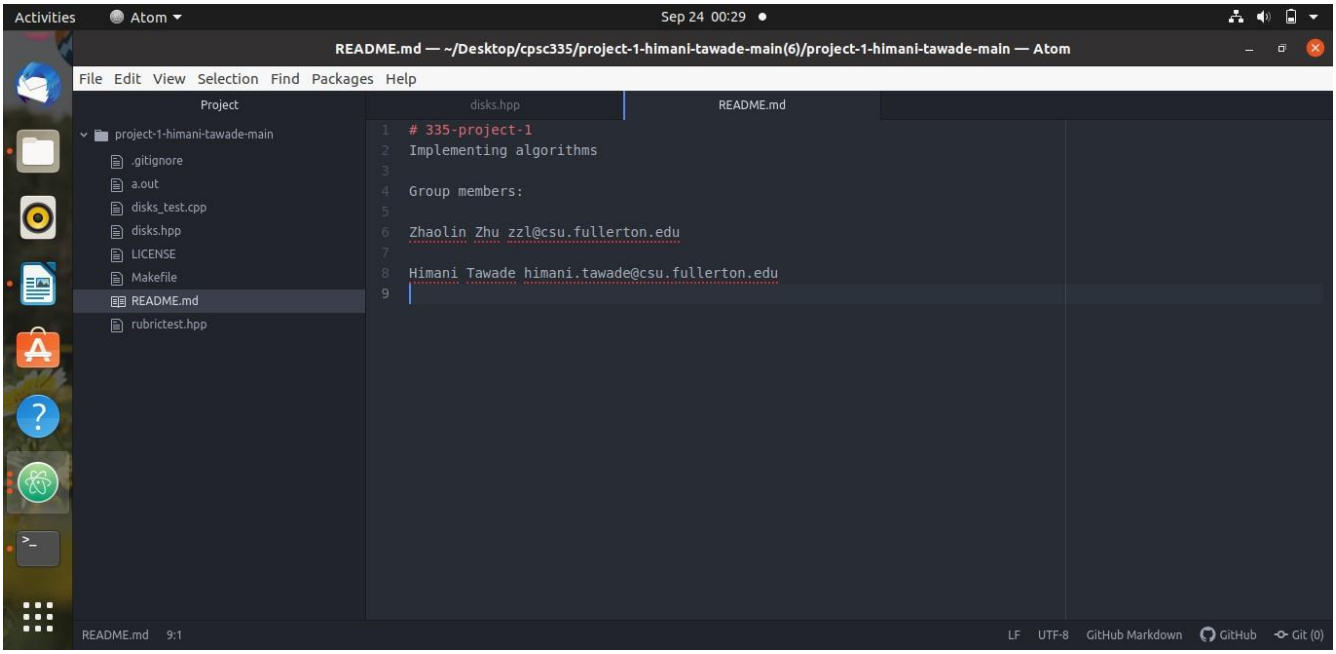
PROJECT 1 REPORT

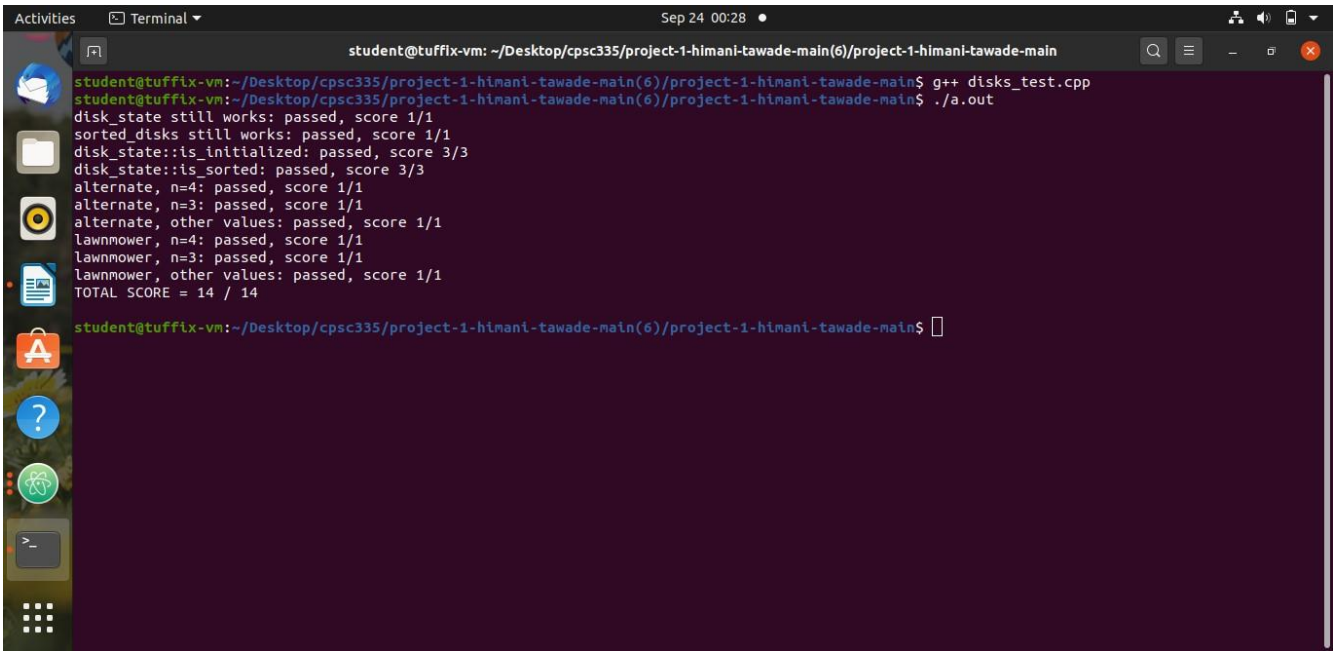
Group Members:

Name	Email ID
Himani Tawade	himani.tawade@csu.fullerton.edu
Zhaolin Zhu	zzl@csu.fullerton.edu

Read Me File:



Code Compilation and Test Run Output:



Psuedo Code:

Alternate Method

Input: a vector of two kinds of element, containing n pairs of disks, length = n * 2
Output: an integer represents the counter of swap operation

```
int swapCounter = 0;                                        //1

for(int i = 0; i < n+2; i++)                                //from 0 to n+1 --> n+2
{
        if (i % 2 == 0)                                     //2
        {
                for (int j = 0; j < 2*n – 1; j = j+2)       //from 0 to 2*n - 2 --> 2*n - 1
                {
                        if (input[j] == input[j + 1])       //4
                        {
                                //do nothing
                        }
                        else if
                        {       if (input [j] == light && input [j + 1] == dark)          //6
```

```
                    //do nothing
                }
                else
                { //swap                                    //3
                 swapCounter++;                             //1
                }
            }

        }
        else
        {
            for (int j = 1; j < 2*n – 2; j = j+2)           //from 1 to 2*n - 3 --> 2*n - 3
            {
                if (input[j] == input[j + 1])               //4
                {
                    //do nothing
                }
                else if
                {       if (input [j] == light && input [j + 1] == dark)     //6
                        //do nothing
                }
                else
                {//swap                                    //3
                 swapCounter++;                             //1
                }
            }

        }
}

return swapCounter;                                         //1
```

Step Count Analysis

S.C. = $\mathbf{1 + \sum_{0}^{n+1}(2 + max(\sum_{0}^{2n-2}(4 + max(0, 6, 4)), \sum_{1}^{2n-3}(4 + max(0, 6, 4)))) + 1}$

= 1 + (n + 2) * (2 + max (((2n - 1) * (4 + max(0, 6, 4))) , ((2n-3) * (4 + max(0 , 6, 4)))) + 1

= 1 + (n + 2) * (2 + max(((2n -1) * 10), ((2n-3) * 10))) + 1

= 1 + (n + 2) * (2 + max((20n – 10), (20n – 30))) + 1

= 1 + (n + 2) * (2 + 20n – 10) + 1

= 1 + (n + 2) * (20n – 8) + 1

= 1 + (20n^2 – 8n + 40n - 16) + 1

= 20n^2 +32n -14

Big O Notation

1. **Definition**
   c = (20 + 32 + 14) = 66
   n0 = 1

20n^2 +32n -14 <= 66n^2, for all n >= 1
So 20n^2 +32n -14 belongs to O(n^2).

2. **Limit**

$$\lim_{n \to \infty} \frac{20n^2 + 32n - 14}{n^2} = \lim_{n \to \infty} \frac{40n + 32}{2n} = \text{constant}$$

So 20n^2 +32n -14 belongs to O(n^2).

Lawnmower

Input: a vector of two kinds of element, containing n pairs of disks, length = n * 2
Output: an integer represents the counter of swap operation

```
int swapCounter = 0;                                                    //1
for (int i = 0; i < n; i++){                                            //from 0 to n – 1 → n
        int currentIndex = 0;                                           //      1
        while (currentIndex < 2 * n - 1){                               //from 0 to 2 * n     – 2 → 2n – 1
                if (input[currentIndex] == input[currentIndex + 1]){    //              4
                        //do nothing                                    //
                } else {                                                //
                        if (input[currentIndex] == light && input[currentIndex + 1] == dark){   //6
                                //do nothing                            //
                        } else {                                        //
                                //swap                                  //3
                                increment swapCounter;                  //1
                        }
                }
                currentIndex++;                                         //1
        }

        while (currentIndex > 0){                                       //from 2 * n – 1 to 1 → 2n
                if (input[currentIndex] == input[currentIndex - 1]){    //4
                        //do nothing                                    //
                } else {                                                //
                        if (input[currentIndex - 1] == light && input[currentIndex] == dark){   //6
                                //do nothing                            //
                        } else {                                        //
                                //swap                                  //3
                                increment swapCounter;                  //1
                        }
                }
                currentIndex--;                                         //1
        }
```

}

return swapCounter; //1

Step Count Analysis

S.C. = $\mathbf{1 + \sum_{0}^{n-1}(1 + \sum_{0}^{2n-2}(3 + 4 + max(0,\ 6 + max(0, 3 + 1)) + 1) + \sum_{1}^{2n-1}(1 + 4 + max(0,\ 6 + max(0, 3 + 1)) + 1)\ + 1}$

=1 + (n)*((1 + (2n - 1) * (3 + 4 + max(0, 6 + 4) + 1) + (2n) * (1 + 4 + max(0, 6 + 4) + 1)) + 1

=1 + (n)*(1 + (2n – 1) * 18 + (2n)*16) + 1

=1 + (n)*(1 + 36n – 18 + 32n) + 1

=1 + (n)*(68n – 17) + 1

= 68n ^ 2 – 17n + 2

Big O Notation

3. **Definition**
   c = (68 + 17 + 2) = 87
   n0 = 1
   68n ^ 2 – 17n + 2 <= 87n^2, for all n >= 1
   So 68n ^ 2 – 17n + 2 belongs to O(n^2).
4. **Limit**

$$\lim_{n \to \infty} \frac{68n^2 - 17n + 2}{n^2} = \lim_{n \to \infty} \frac{136n - 17}{2n} = \text{constant}$$

So 68n ^ 2 – 17n + 2 belongs to O(n^2).