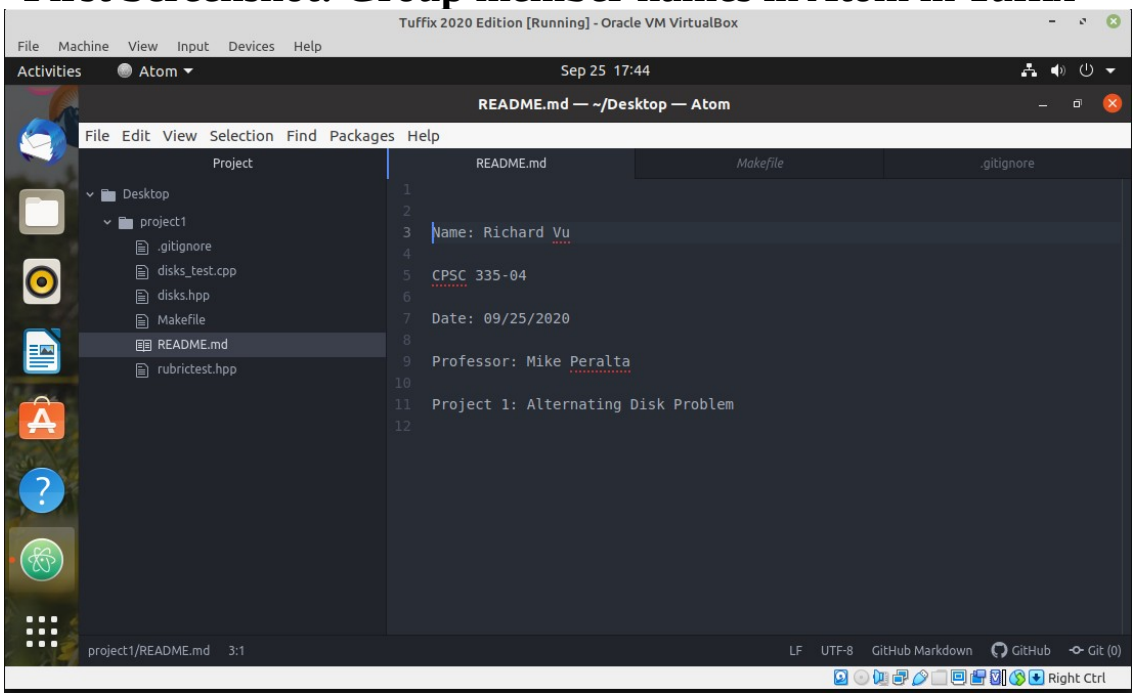
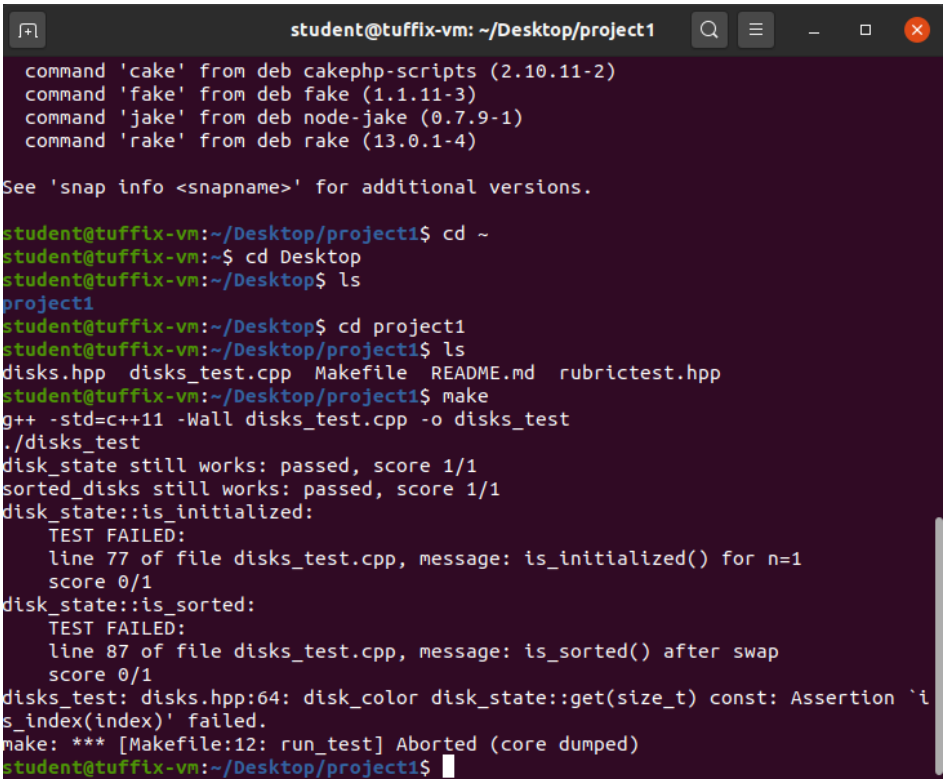
Richard Vu
CPSC 335
Project 1
09/25/2020

Programming Assignment 1

First Screenshot: Group member names in Atom in Tuffix



Second Screenshot: code executing with the command make



Lawnmower Algorithm:

Lawnmower Algorithm
Input: positive integer N, A list of 2 * N disks of alternating colors light-dark, starting with light.
Output: A list of 2*N disks, where the first *n* disks are dark, the next *n* disks are light, and an integer *m* representing total number of *swaps* to move the light disks after the dark disks.

Lawnmower Pseudo-code:

def sort_lawnmower(disks):

```
    if len(disks) == 0:
            return 0
    endif
    swaps = 0
    for k = 0 to 2*n do
            for j = 0 to 2*n – 1  do
                if disks[j] greater than disks[j + 1] do
                        swap(disks[j], disks[j + 1]
                        swaps += 1
                endif
            endfor
            for m = 2 * n – 1 to 1  do
                if disks[i] is less than disks[i – 1]  do
                        swaps(disk[i], disks[i – 1])
                        swaps += 1
                endif
            endfor
    endfor
return (disks, swaps)
```

Big Oh O(n) mathematical analysis for the Lawnmower Algorithm

```
if len(disks) == 0: +1
        return 0 + 1
swaps = 0 → +1
for k = 0 to 2*n do      +1  →   [(2n – 0)/ 1] + 1 = 2n + 1

    for j = 0 to 2*n – 1  do + 1 → [(2n – 1 – 0) / 1] + 1 = 2n
        if disks[j] greater than disks[j + 1] do +1
            swap(disks[j], disks[j + 1]          +1
                                                    max(1,4) = 4
            swaps += 1                              +1
        endif
    endfor

    for m = 2 * n – 1 to 1  do  + 1 → abs([1 – (2n – 1) / 1] + 1 )=
                                                              2n + 3
        if disks[i] is less than disks[i – 1]  do +1
            swaps(disk[i], disks[i – 1])         +1
            swaps += 1                              +1
                                                    max(1, 4) = 4
        endif
    endfor
endfor
return (disks, swaps) → + 1
```

Mathematical Analysis and efficiency class for Lawnmower Algorithm.

$$\sum_{0}^{2n} 1(2n+1) \sum_{0}^{2n-1} 4(2n) + \sum_{2n-1}^{1} 4|-2n+3|$$

$$(2n+1)[8n+8n+12]$$

+2 initial if
+ 1 swaps = 0
+ 1 return
statement

$$32n^2+40n+12+4$$

$$32n^2+40n+16$$

$$32n^2+40n+16 \in O(n^2)(trivial\ case)$$

$$32n^2 = O(n^2)(dominated\ term)$$

$$O(n^2)(constant\ factor)$$

Alternating Algorithm:

Alternating Algorithm
Input: positive integer N, A list of 2 * N disks of alternating colors light-dark, starting with light.
Output: A list of 2*N disks, where the first *n* disks are dark, the next *n* disks are light, and an integer *m* representing total number of *swaps* to move the light disks after the dark disks.

Alternating Pseudo-code:

```
def sort_alternate(disks):
    if len(disks) == 0:
        return 0
    endif
    swaps = 0

    for j = 0 to 2*n do   + 1 → [(2n – 0 )/ 1] + 1 = 2n + 1
            for k = 0 to 2*n – 1 do
                if disks[i] is greater than disk[j+1] do
                        swap(disks[i], disks[j+1)
                        swaps += 1

                endif
    endfor

    return (disks, swaps)
```

Mathematical Analysis of the Alternating Algorithm:

```
if len(disks) == 0:        +1
        return 0                + 1
    endif
    swaps = 0        + 1

    for j = 0 to 2*n do   + 1 → [(2n – 0 )/ 1] + 1 = 2n + 1
            for k = 0 to 2*n – 1 do  + 1     → [(2n – 1 – 0) / 1] + 1 = 2n
                if disks[i] is greater than disk[j+1] do      + 1
                        swap(disks[i], disks[j+1)             +1
                        swaps += 1                            +1
                                                  max(1, 3) = 3 + 1 = 4
                endif
    endfor

    return (disks, swaps) + 1
```

$$\sum_{0}^{2n+1} 1 \sum_{0}^{2n} 4$$

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

$$(16n^2+8n)+4$$

$$16n^2+8n+4$$

$$4n^2+10n+8 \in O(n^2)(trivial)$$

$$4n^2 = O(n^2)(dominated\ term)$$

$$n^2 = O(n^2)(dropped\ constant)$$