Assignment 1: Alternating disks problem

SangYong Jin, Danny Navarro, Shelley Pham

Given 2n alternating disks (dark, light) the program reads the number of single color disks (light or dark), arranges the disks in the correct order and outputs the number of swaps

Contents

Pseudocode

Left-to-Right

```
for (k = 0 to number_of_disks):
      for (y = 0 to number_of_disks - 1){
            if (disks[y] == dark_disk){
                  if (disks[y + 1] != dark_disk){
                        swap(disks[y], disks[y + 1]);
                        m++;
                  }
            }
      }
}
```

Efficiency

```
S.C. =
( (n - 1)/1  + 1)  *  [  (n-2)/1   + 1) *  ( 1+ max( (1+max(2,0)), 0) ) ]
=  ( (n-1)/1  + 1)  *  [  (n-2)/1   + 1) *  ( 1+ max( 3, 0) ) ]
=  ( (n-1)/1  + 1)  *  [  (n-2)/1   + 1) *  4 ]
=  ( (n-1)/1  + 1)  *  [ 4n  -  4 ]
= n * (4n - 4) = 4n^2 - 4n
```

Keep only leading terms

=> `4n^2`

Drop multiplicative constants

=> `n^2, therefore O(n^2)`

Back to Contents

Lawnmower

```
n = num_single_disks
x = 2 * n
for (k = 0 to (n/2 + 1)){
      for (i = 0 to x-1){
            if (disks[i] == dark_disk){
                  if (disks[i + 1] != dark_disk){
                        swap(disks[i], disks[i + 1]);
                        m++;
                  }
            }
      }

      for (i = (x-1) down to 0){
            if (disks[i-1] == dark_disk){
                  if (disks[i] != dark_disk){
                        swap(disks[i], disks[i-1]);
                        m++;
                  }
            }
      }
}
```

Efficiency

S.C. =

Back to Contents

```
( (n/2)/1  + 1)  * { [  (n-2)/1   + 1) *  ( 1+ max( (1+max(2,0)), 0) ) ]   + [  (1 - (n-1))/-1   + 1) *  ( 1+ max( (1+max(2,0)), 0) ) ] }
= ( (n/2)/1  + 1)  * { [4n  -  4] + [  (1 - (n-1))/-1   + 1) *  ( 1+ max( (1+max(2,0)), 0) ) ] }
= ( (n/2)/1  + 1)  * { [4n  -  4] + [  (1 - (n-1))/-1   + 1) *  4  ] }
= ( (n/2)/1  + 1)  * { [4n  -  4] + [  ( n - 2  - 1) *  4  ] }
= ( (n/2)/1  + 1)  * { 4n  -  4  +  4n - 12 }
= ( (n/2)  + 1)  * { 4n - 12 }
= 2n^2 + 4n - 6n - 12
= 2n^2 - 2n - 12
```

Drop additive constant

=> 2n^2

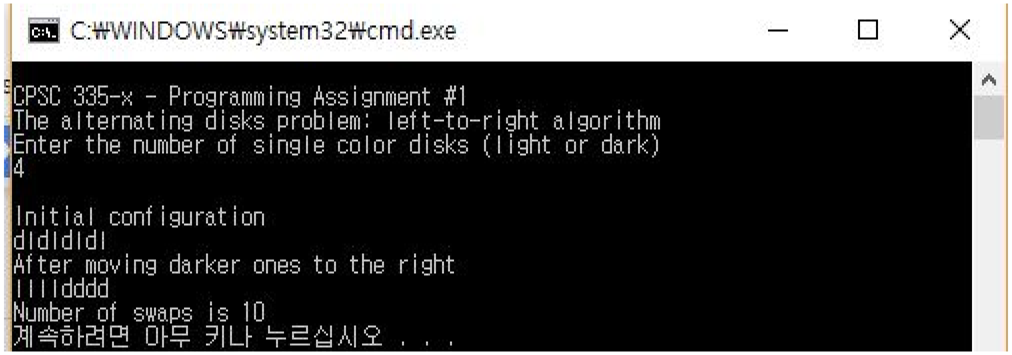
Drop multiplicative constant

=>n^2, therefore this is O(n^2)

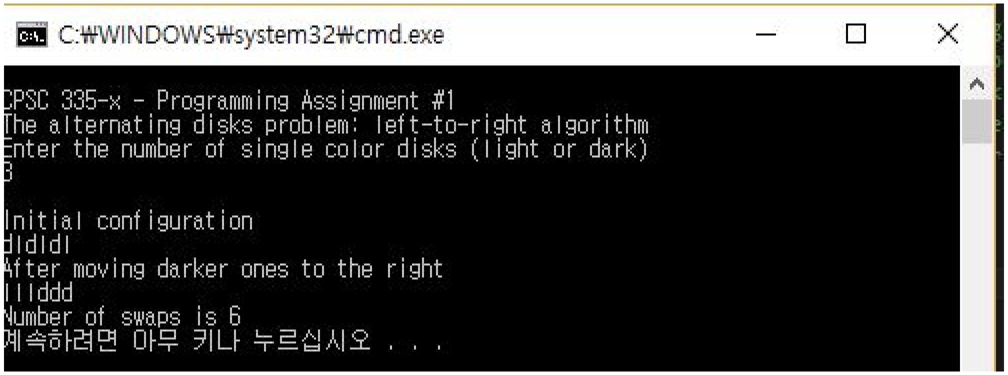
Back to Contents

Output

Left-to-Right



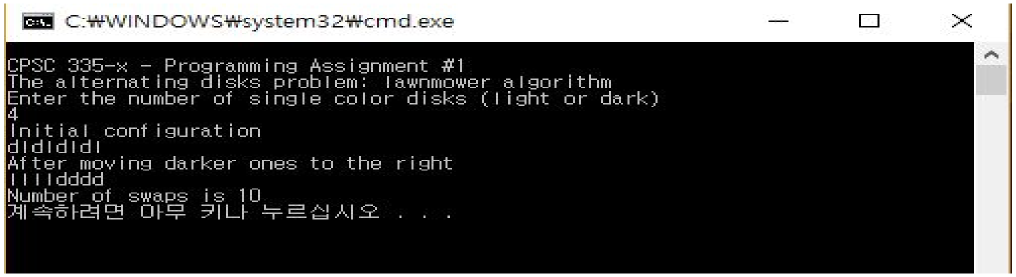
```
CPSC 335-x - Programming Assignment #1
The alternating disks problem: left-to-right algorithm
Enter the number of single color disks (light or dark)
4

Initial configuration
dldldldl
After moving darker ones to the right
lllldddd
Number of swaps is 10
계속하려면 아무 키나 누르십시오 . . .
```



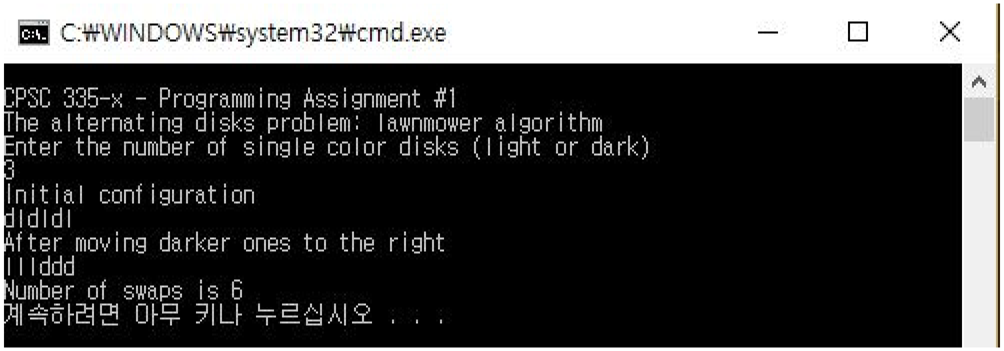
```
CPSC 335-x - Programming Assignment #1
The alternating disks problem: left-to-right algorithm
Enter the number of single color disks (light or dark)
3

Initial configuration
dldldl
After moving darker ones to the right
lllddd
Number of swaps is 6
계속하려면 아무 키나 누르십시오 . . .
```

[Back to Contents]

Lawnmower



```
CPSC 335-x - Programming Assignment #1
The alternating disks problem: lawnmower algorithm
Enter the number of single color disks (light or dark)
4
Initial configuration
dldldldl
After moving darker ones to the right
lllldddd
Number of swaps is 10
계속하려면 아무 키나 누르십시오 . . .
```



```
CPSC 335-x - Programming Assignment #1
The alternating disks problem: lawnmower algorithm
Enter the number of single color disks (light or dark)
3
Initial configuration
dldldl
After moving darker ones to the right
lllddd
Number of swaps is 6
계속하려면 아무 키나 누르십시오 . . .
```

Back to Contents

Code

Left-to-Right

```
#include <iostream>
#include <iomanip>
#include <cstdlib>
using namespace std;
void print_disks(int n, char *disks){

      for (int i = 0; i < n; i++)

      cout << disks[i];
}
// YOU NEED TO IMPLEMENT THIS FUNCTION
// function to print the list of disks, given the number of single color disks and the actual
list
// n represents the number of single color disks
// disks represents the list of disks (index 0 being the first disk) where
// 0 = a light color disks
// 1 = a dark color disks
int main() {
      int n, m=0, k, i;
      char *disks;
      // display the header
      cout << endl << "CPSC 335-x - Programming Assignment #1" << endl;
      cout << "The alternating disks problem: left-to-right algorithm" << endl;
      cout << "Enter the number of single color disks (light or dark)" << endl;
      // read the number of disks
      cin >> n;
      int x = 2 * n;
```

Back to Contents

```
// allocate space for the disks
disks = new char[2 * n];
// set the initial configurations for the disks to alternate
for (i = 0; i < (x/2); i++) {
     disks[2 * i] = 'd';
     disks[2 * i + 1] = 'l';
}
cout << endl;

// print the initial configuration of the list of disks
cout << "Initial configuration" << endl;
print_disks(x, disks);
// PART OF CODE MISSING
// loop to push light one before the darks ones
for (k = 0; k < x; k++) {
     // YOU NEED TO COMPLETE THIS PART OF CODE FOR GOING LEFT TO RIGHT
     for (int y = 0; y < x-1; y++){
          if (disks[y] == 'd'){
               if (disks[y + 1] != 'd'){
                    swap(disks[y], disks[y + 1]);
                    m++;
               }
          }
     }


}
cout << endl;
// after shuffling them
cout << "After moving darker ones to the right" << endl;
print_disks(x, disks);
```

Back to Contents

```
	cout << endl;
	// print the total number of moves
	cout << "Number of swaps is " << m << endl;
	// de-allocate the dynamic memory space
	delete[] disks;
	return EXIT_SUCCESS;
}
```

Lawnmower

```
#include <iostream>
#include <iomanip>
#include <cstdlib>
using namespace std;
void print_disks(int n, char *disks){
	for (int i = 0; i < n; i++){

		cout << disks[i];
	}

}
// YOU NEED TO IMPLEMENT THIS FUNCTION
// function to print the list of disks, given the number of single color disks and the actual
list
// n represents the number of single color disks
// disks represents the list of disks (index 0 being the first disk) where
// 0 = a light color disks
// 1 = a dark color disks
int main() {
	int n, m=0, k, i;
	char *disks;
```

Back to Contents

```
// display the header
cout << endl << "CPSC 335-x - Programming Assignment #1" << endl;
cout << "The alternating disks problem: lawnmower algorithm" << endl;
cout << "Enter the number of single color disks (light or dark)" << endl;
// read the number of disks
cin >> n;
// allocate space for the disks
disks = new char[2 * n];
int x = n * 2;
// set the initial configurations for the disks to alternate
for (i = 0; i < (x/2); i++) {
      disks[2 * i] = 'd';
      disks[2 * i + 1] = 'l';
}
// print the initial configuration of the list of disks
cout << "Initial configuration" << endl;
print_disks(x, disks);
// PART OF CODE MISSING
// loop to push light one before darks ones
for (k = 0; k < n / 2 + 1; k++) {
      // DEVELOP ONE FOR LOOP FOR GOING LEFT TO RIGHT
      // DEVELOP ANOTHER FOR LOOP FOR GOING RIGHT TO LEFT
      for (int i = 0; i < (x-1); i++){
            if (disks[i] == 'd'){
                  if (disks[i + 1] != 'd'){
                        swap(disks[i], disks[i + 1]);
                        m++;
                  }
            }
      }
```

Back to Contents

```
        for (int i = (x-1); i > 0; i--){
            if (disks[i-1] == 'd'){
                if (disks[i] != 'd'){
                    swap(disks[i], disks[i-1]);
                    m++;
                }
            }
        }

    }
    cout << endl;
    // after shuffling them
    cout << "After moving darker ones to the right" << endl;
    print_disks(x, disks);
    cout << endl;
    // print the total number of moves
    cout << "Number of swaps is " << m << endl;
    // de-allocate the dynamic memory space
    delete[] disks;
    return EXIT_SUCCESS;
}
```

Back to Contents