Sander VanWilligen

Zackery Lovisa

**Com S 311 – Project 3 Report**

**minCostVC:**

For the minCostVC, the recurrence relation is as follow:

input: int[][]M with height h and width w, and indices I and J(representing width and height)

minCostVC(M, I, J){

if(I == h-1) return M[i][j];

else{

return minimum of the following:

minCostVC(M, I+1, J-1);

minCostVC(M, I+1, J);

minCostVC(M, I+1, J+1);

//also checking for the j index being 0 or w-1

}

}

The runtime (with our iterative solution, not with the recurrence relation requested above) is as follows:

Time to compute cost matrix = O(w\*h)

Time to find lowest cost column number = O(w)

Time to create the returned Integer Array List is O(h) (O(2h) technically)

Total Time = O(h\*w), where h and w are the height and width of M respectively

**stringAlignment:**

For the stringAlignment, the recurrence relation is as follow:

input: string x and string y

int n = x.length

int m = y.length

alignCost(x, y, n, m){

if(m == 0) return (n-m)\*4;

if(n == 0) return 0;

if(x[n-1] = y[m-1] return alignCost(x, y, n-1, m-1));

else{

return minimum of the following:

(alignCost(x, y, n-1, m-1) + 2);

(alignCost(x, y, n-1, m) + 4);

}

}

The runtime (with our iterative solution, not with the recurrence relation requested above) is as follows:

Time to compute cost matrix = O(n\*m) (n is x.length, m is y.length)

Time to build the String = O(n)

Total time = O(n\*m)