Eleanor Barry

CSCE 350 Section 002

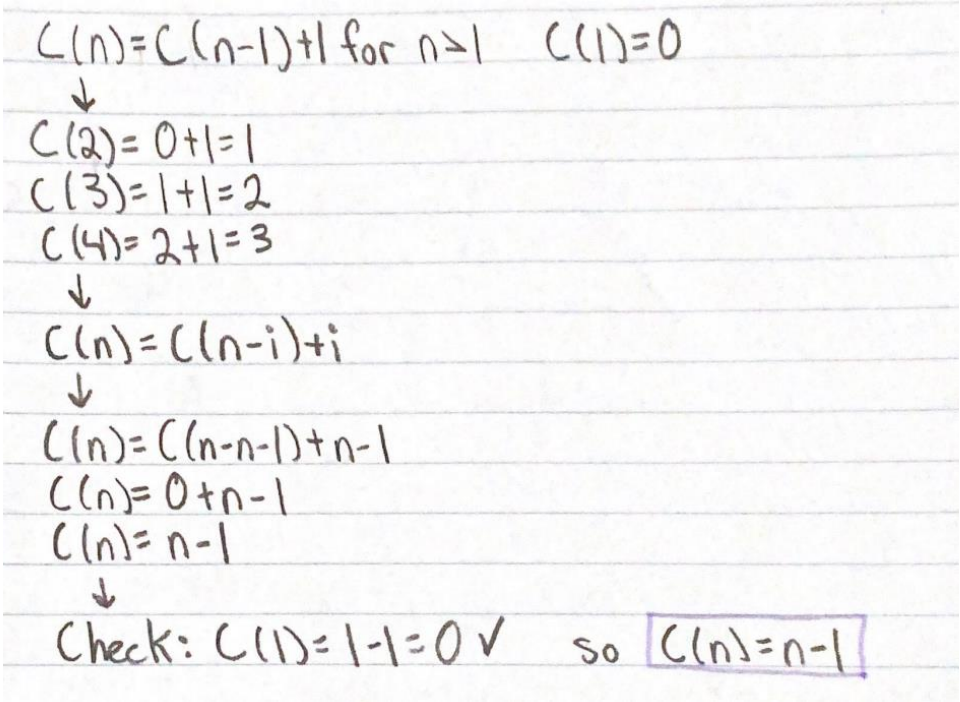
1 March 2021

Homework 5

**2.4.9**

a) This algorithm solves for the smallest number in an array of real numbers by using recursion.

b)



$C(n)=C(n-1)+1$ for $n \geq 1$ $\quad C(1)=0$

↓

$C(2)=0+1=1$

$C(3)=1+1=2$

$C(4)=2+1=3$

↓

$C(n)=C(n-i)+i$

↓

$C(n)=C(n-n-1)+n-1$

$C(n)=0+n-1$

$C(n)=n-1$

↓

Check: $C(1)=1-1=0$ ✓ so $\boxed{C(n)=n-1}$

**3.1.5**

```
$d_0 = 0$
for i = 0 to (n – 1) do
    $d_0$ += A[0, i]
end for

if $d_0 = 2$ then
    return “Ring”
end if

if $d_0 = 1$ then
    return “Star”
end if

$d_1 = 0$
for i = 0 to n-1 do
    $d_1$ += A[1, i]
end for

if $d_0 = d_1$ then
    return “Fully Connected Mesh”
else
    return “Star”
end if
```

**3.1.10**

Yes, it is possible to implement selection sort for linked lists with the same $\Theta(n^2)$ efficiency as the array version. This is done by sorting and then linking the minimal element to the unsorted part instead of swaps.

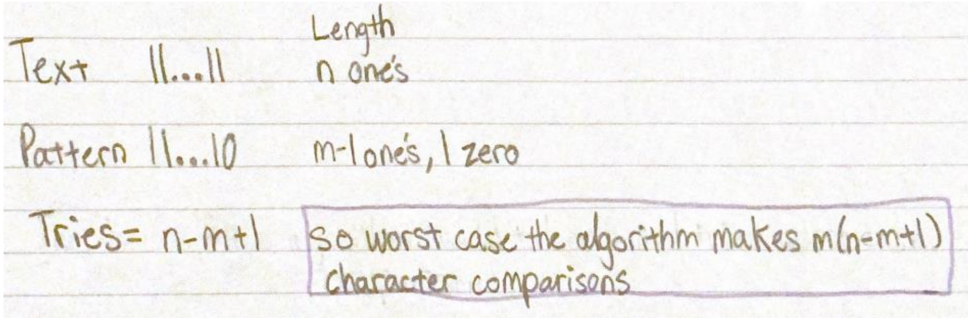
Code:

```
void selectionSort(node head) {
    node temp = head;

    while (temp) {
        node min = temp;
        node nextN = temp.next;

        while (nextN) {
            if (min.data > nextN.data) {
                min = nextN;
            }
            nextN = nextN.next;
        }

        int swapD = temp.data;
        temp.data = min.data;
        min.data = swapD;
        temp = temp.next;
    }
}
```

### 3.1.13

Bubble sort is stable because the order of any two equal elements is kept the same in the sorted array list. By using the book pseudocode, you can see that the if statement does not swap the order of two equal elements so bubble sort is stable.

### 3.2.6



| | | Length |
|---|---|---|
| Text | 11...11 | n one's |
| Pattern | 11...10 | m-1 one's, 1 zero |
| Tries= n-m+1 | | So worst case the algorithm makes m(n-m+1) character comparisons |