Data Structure

Fall 2020, Homework #1 Solution

---

1. $(0.5)^{\sqrt{n}} \quad \frac{1}{n} \quad 10^{1000} \quad \log_2 n \quad n \quad n\log_2 n \quad 4^{\log_2 n} \quad 2n^2+n+n^5 \quad 2^n \quad 2.1^n$

2. Program 1: $\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\sum_{k=j}^{n-1} 1 = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1}(n-j) = \sum_{i=0}^{n-1}(\frac{n}{2}(1+n)) = \frac{n^2}{2}(1+n) = \Theta(n^3)$

   Program 2: $\sum_{i=1}^{\lfloor\frac{2n-1}{5}\rfloor+1}(\sum_{j=1}^{\lfloor\sqrt{n}\rfloor}1 + \sum_{k=1}^{\lfloor\sqrt{1000}\rfloor}1) = \sum_{i=1}^{\lfloor\frac{2n-1}{5}\rfloor+1}(\lfloor\sqrt{n}\rfloor + \lfloor\sqrt{1000}\rfloor)$

   $$= (\lfloor\frac{2n-1}{5}\rfloor + 1)(\lfloor\sqrt{n}\rfloor + \lfloor\sqrt{1000}\rfloor) = \Theta(n\sqrt{n})$$

3. Solution:

   (i) 12 steps to reach the END node

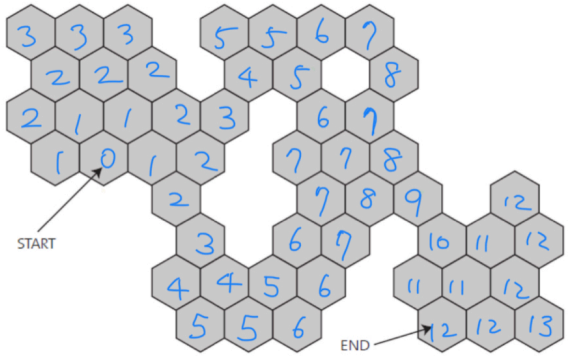

Figure 3: Floor plan.

   (ii) Yes

   (iii) The worst case scenario w.r.t. the problem is the floor plan that contains a single line of cells, with the START and END cells each at one end of the line. $N-1$ steps is needed in this case to reach the END cell from the START cell. This is the worst case because one must visit every cell in the floor plan in order to reach the END cell. (Note that N-1 is also the maximum number of steps one can make without visiting at least one cell twice.)

4. Here's the insertion sort example from the class notes. The running time is:

$$c_1 \cdot n + (c_2 + c_3 + c_7) \cdot (n-1) + c_4 \cdot \sum_{j=2}^{n} t_j + (c_5 + c_6) \cdot \sum_{j=2}^{n} t_{j-1}$$

	cost	times
for j=2 **to** *length*(A)	c_1	n
do key=A[j]	c_2	n-1
"insert A[j] into the sorted sequence A[1..j-1]"	0	n-1
i=j-1	c_3	n-1
while i>0 **and** A[i]>key	c_4	$\sum_{j=2}^{n} t_j$
do A[i+1]=A[i]	c_5	$\sum_{j=2}^{n} (t_j - 1)$
i--	c_6	$\sum_{j=2}^{n} (t_j - 1)$
A[i+1]:=key	c_7	n-1

where t_j is the # of comparisons needed for key $A[j]$.

i. Best case scenario is when the list is already sorted in ascending order. In this case, $t_j = 1$.

So the running time $= c_1 \cdot n + (c_2 + c_3 + c_7) \cdot (n-1) + c_4 \cdot \sum_{j=2}^{n} 1$

$$= c_1 \cdot n + (c_2 + c_3 + c_7) \cdot (n-1) + c_4 \cdot (n-1) = \Theta(n)$$

ii. Worst case scenario is when the list is already sorted in descending order. In this case, $t_j = j - 1$. So the running time is

$$c_1 \cdot n + (c_2 + c_3 + c_7) \cdot (n-1) + c_4 \cdot \sum_{j=2}^{n} (j-1) + (c_5 + c_6) \cdot \sum_{j=2}^{n} (j-2)$$

$$= c_1 \cdot n + (c_2 + c_3 + c_7) \cdot (n-1) + c_4 \cdot \left(\frac{n-1}{2} \cdot n\right) + (c_5 + c_6) \cdot \left(\frac{n-1}{2} \cdot (n-2)\right) = \Theta(n^2)$$

5. $f(N) = \sin(N) + 1; g(N) = \cos(N) + 1$ (Note that f and g are made to be both positive functions to avoid confusion.)

6.

```
void ReversePrint(Node* listHeader)
{
      if(listHeader == null)    return; # optional as we only
ask for pseudo code

      if(listHeader->next != null)
      {
            ReversePrint(listHeader->next);
      }
      system.print(listHeader->data);
}
```