Data Structures

Fall 2020, Homework #1

Date: Oct. 12, 2020

1. (20 pts) Place the following 10 functions in order such that if f appears before g, it means that $f = O(g)$. If multiple functions have the same complexity, please indicate so.

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

2. (20 pts) For the following two code fragments, determine in Θ-notation the number of times the function f is invoked. The function f does not invoke itself. Show your derivation in sufficient detail.

```
for(int i = 0; i < n; i++) {
    for(int j = 0; j < n; j++) {
        for(int k = j; k < n; k++) {
            f();
        }
    }
}
```

Figure 1: Program 1.

```
for(int i = 1; i <= 2*n; i+=5) {
    for(int j = 1; j*j <= n; j++) {
        f();
    }
    for(int k = 1; k*k < 1000; k = k+1) {
        f();
    }
}
```

Figure 2: Program 2.

3. (30 pts) Consider a hypothetical floor plan such that the area is organized in hexagonal cells as shown in Figure 3. We begin at the cell marked *start*, and we want to reach the cell marked *end*. We can travel from one cell to another, if the two cells are neighboring, i.e., they share an edge. Each time we cross from one cell to a destination cell, we consider the move as a single step and the destination cell as visited. We want to find the shortest path from the start to the end, minimizing the number of steps. Consider the following algorithm:

   (1) We consider the start cell as visited, and we write the number 0 on the cell.

   (2) If a cell has the number n written on it, we can visit all neighboring cells, writing the number $n + 1$ on it if either:

      (a) the neighboring cell has not been visited before, or

      (b) the neighboring cell has been visited before and holds the number m such that $m > n + 1$.

   (3) We stop the execution of the algorithm when we can no longer execute (2).

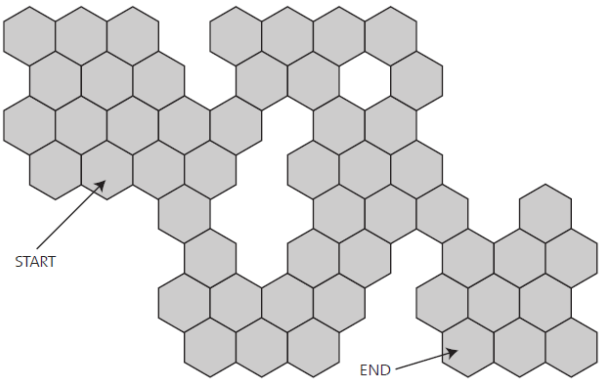

Figure 3: Floor plan.

Answer the following questions:

(i) (15 pts) Execute the algorithm on the floor plan given in Figure 3, writing numbers on each cell. Show the result.

(ii) (5 pts) Is the algorithm correct? That is, does it always find the shortest path? Yes or No? No explanations needed.

(iii) (10 pts) For a given floor plan with N cells including the start and the end cell, how many steps do we need to reach the end cell, starting from the start cell, in the worst case scenario (i.e., the worst floor plan layout with N cells)? Why?

(4) (10 pts) Consider the Insertion sort algorithm discussed in class. Consider sorting n numbers.

i. What is the best case scenario? What is the best case running time?

ii. What is the worst case scenario? What is the worst case running time?

(5) (10 pts) Find two functions $f(N)$ and $g(N)$ such that neither $f(N) = O(g(N))$ nor $g(N) = O(f(N))$.

(6) (10 pts) Given a singly linked list whose node type is defined as

```
class Node {
public:
        int     data;
        Node*next;}
```

Suppose *ptr* points to the beginning of the list (i.e., *pts* is the header of the list). Write a recursive program in pseud-code to print the content of the list in reverse order.