CSE 2321 Homework 8 Template

Problem 1

1A

Insertion sort:

$[8, 3, 4, 5, 2, 1, 0, 9, 7, 6]$

$[3, 8, 4, 5, 2, 1, 0, 9, 7, 6]$

$[3, 4, 8, 5, 2, 1, 0, 9, 7, 6]$

$[3, 4, 5, 8, 2, 1, 0, 9, 7, 6]$

$[2, 3, 4, 5, 8, 1, 0, 9, 7, 6]$

$[1, 2, 3, 4, 5, 8, 0, 9, 7, 6]$

$[0, 1, 2, 3, 4, 5, 8, 9, 7, 6]$

$[0, 1, 2, 3, 4, 5, 8, 9, 7, 6]$

$[0, 1, 2, 3, 4, 5, 7, 8, 9, 6]$

$[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$

1B

MergeSort:

$[8, 3, 4, 5, 2, 1, 0, 9, 7, 6]$

$[8, 3, 4][5, 2][1, 0, 9][7, 6]$

$[8, 3][4][5][2][1, 0][9][7][6]$

$[8][3][4][5][2][1][0][9][7][6]$

$[3, 8][4][5][2][0, 1][9][7][6]$

$[3, 4, 8][2, 5][0, 1, 9][6, 7]$

$[2, 3, 4, 5, 8][0, 1, 6, 7, 9]$

$[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$

Problem 2

Algorithm for Adjacency Matrix:
The algorithm assumes that you have a (n x n) adjacency matrix A and a (n x n) zero matrix B of the same size. The dimension n is equal to the number of vertices in the graph.
The algorithm returns B, the transpose of A, which provides all of the information necessary to construct the transpose graph.

```
EqualAlgorithm1(A, n):
    for i=0 to n
        for j=0 to n
            if A[i][j]==1:
                B[j][i] = 1
    return B
```

Runtime: $T(n) = \Theta(n^2) = \Theta(|v|^2)$

Algorithm for Adjacency List:
The algorithm represents the adjacency list as an array A of length n of type

list. So, each index in the array holds the corresponding adjacency list of that vertex.
The length of the array, n, is equal to the number of vertices in the graph.
The algorithm creates an array B of length n of type list, but all lists are empty and will represent the adjacency list of the transpose of the graph.

```
EqualAlgorithm2(A, n):
    for i=0 to n
        // Iterates through the length of the list at A[i].
        for j=0 to A[i].length
            // If the list is non-empty, store the value at
            // index j of the list into x.
            int x = A[i].at(j)
                // Add i into the list of the array at B[x].
                B[x].add(j)
    return B
```

Runtime: $T(n) = \Theta(|V| + |E|)$

Problem 3

If we start with a sorted list A, and Alice creates a new list A' according to that algorithm, the resulting list will have 3 subsections:

The first subsection is in order.

The second/middle subsection is reversed.

The third/last subsection is in order.

So by analyzing the running times of each subsection and adding them, it will be possible to determine the running time for each type of sort algorithm.

3A

Insertion sort:
The running time is $T(n) = \Theta(n + n^2)$

A sorted subsection is the best case for Insertion sort, the best time is $T(n) = \Theta(n)$.
A reversed subsection is the worst case for Insertion sorts and for Insertion sort, the worst time is $T(n) = \Theta(n^2)$.

3B

Merge sort:
The running time is $T(n) = \Theta(nlog(n))$

For any list of size n, Merge sort has the same number of steps/running time which is $T(n) = \Theta(nlog(n))$.
So it doesn't matter if the subsections are in order or not because they all will have the same running time.

Problem 4

For the lower bound, use $2^{k-1} < n$
$M(n) \geq 2M\left(\lfloor \frac{n}{2} \rfloor - 1\right) + n$
$M(n) \geq 2M(2^{k-1} - 1) + n$
$T(n) = 2T(2^{k-1} - 1) + 2 \cdot 2^{k-1}$
$T(n) = T(2 \cdot 2^{k-1})$
$T(n) = T(4n)$
$T(4n) = 2T(2n) + 4n$
$2^{k+1}T\left(\frac{n}{2^{k-1}}\right) + 4(k+1)n$
$4n + 4n\log_2(n) + 8n$
$\Omega(n\log(n))$