HW4

Porf. Helmut Alt
Team 31. 20140174 Hyungrok Kim
20150710 Kanghee Cho

1-a

for every outdegree->$\theta(|V| + |E|)$ indegree->$\theta(|V| * |E|)$

from the given adjacency-list of a direct graph, Say that Adj[u] is the list of vertices adjacent to vertex u. then the length of Adj[u] is the outdegree of vertex u, and the sum of the lengths of all the adjacency-list in Adj is |E| (total number of edges). first, the time complexity of the outdegree of every vertex, to check a vertex u's outdegree, we need to get Adj[u] and check the number of elements in Adj[u]. Thus to compute the outdegree of every vertex, the time complexity is $\theta(|V| + |E|)$, |V| to check all the vertices and |E| to check all the outdegree. For the indegree, in adjacency-list for a vertex u, we need to check every vertex's adjacency-list to check if it has u as an element. So for a vertex, we need |E| times to check. And for every vertex, the time that takes to compute every vertex's outdegree is

$$\theta(|V| * |E|)$$

1-b

G = (V, E) is the graph GT = (V, ET) where ET = {(v, u) ∈ V × V |(u, v) ∈ E}.

first, to produce the transpose of a graph G with its adjacency-matrix, we need to check every section, assume that it is Adj[i][j] then it goes to the new adjacency-matrix Adj'[j][i](i and j changed), when the procedure is done for every section then graph GT represented by Adj' becomes the transpose of G.
the code for this is simple,

```
for (i=1; i<=|V|; i++){
	for(j=i+1; j<=|V|; j++){
		A'[j][i] = A[i][j];  // A is the adjacency-matrix for G and A' for GT
	}
}
```

in here because the procedure loop acts V^2times, the time complexity to compute a transpose of a graph follows $\theta(|V|^2)$

then, to produce the transpose of a graph G with its adjacency-list, there will be A which is G's adjacency-list of |V| length. A[i] has vertices which are adjacent to vertex i as elements. so what we need to do is, if the element in A[i] is vertex u, then implement vertex i to A'[u]. this algorithm is liks,

```
for(i=1; i<=|V|, i++){

	for every vertex u in A[i]

	add vertex i to A'[u]

}
```

like what we said in problem number 1 above, the sum of every adjacency-list A[i] equals to the number of edges |E|, and to go through every A[i] we need |V|

so the running time is $\theta(|V| + |E|)$