**

Problem

6-1

Find strongest connection from u to v with vagueness of k at O(kE + V).
Connection(u, v, k) uses a modified Breadth First Search and uses a hashtable to reach
this time complexity.

```
Connection(u, v, k):
    define
        hashtable
        queue
        connection to None

    add u to queue
    set u to 1 at hashtable

    for i=0 to k do
        pop user from left of queue
        for every friend of user do
            append friend to queue

            weight = hashtable[user] * EdgeRank(user, friend)
            if hashtable[friend] is exists and hashtable[friend] > weight do
               continue
            hashtable[friend] = weight

            if friend is v and
                if connection is None or hashtable[friend] >
hashtable[connection] do
                    connection = friend

    return hashtable[connection]
```

** Problem 6-2

* a) Installation order can be done with topological sort at O(V + E) time. Using a set, a queue and Depth First Search for every library.

* b)

Depth subcycle will be called V times and recurse E times. Therefore O(V + E). Searching something in set is average of O(1). Queue is LIFO.

```
Depth(library, visited, queue):
    if library is not in visited do
        add library to visited
        for every dependency of library do
            Depth(dependency, visited, queue)
        add library to queue
```

```
Topologicalsort(V):
    define
        queue
        visited is set

    for every library in V do
        Depth(library, visited, queue)

    return queue
```