CS 5592: Design and Analysis of Algorithms
Homework 5

Author: Hayden McParlane

(1.a) Prove that this decision problem is in the NP-class.

Proof that a decision problem is in the NP-class simply requires writing a verification algorithm that can verify a given certificate in polynomial time.

Algorithm 1 A verification algorithm for the problem above. This algorithm assumes that the projects in need of completion, P, are known to keep its format consistent with typical NP-class verification algorithms.

```
function VERIFY(C, k)
    for project in P do
        for company in k do
            1. Retrieve projects from L completable by the current company.
            2. Remove the project from the project list.
        end for
    end for
    if project list is empty then
        return Yes
    else
        return No
    end if
end function
```

This algorithm proves that the problem is in the NP-class because it executes the verification in polynomial time. For each project chosen, all of the companies are checked to determine which projects that company can complete. If a company is found that can complete every project, the algorithm returns yes. Otherwise, no is returned. This is polynomially related to the number of projects in P because the problem states that all of the projects can be completed but that none of the companies can complete all of the projects alone. This means that the upper bound on the list of projects associated with a given company is $|P| = m$. So, even if each company could complete all of the projects the verification have the upper bound $|P|\cdot(!|P|)^k \leq |P|\cdot|P|^k = |P|^{k+1} = m^{k+1} = m^{constant}$. Therefore, this problem is no harder than the NP-class and belongs to the NP-class.

(1.c) Suppose you have an algorithm A(P, m, C, n, k) that can determine, for any given integer k, whether k companies are enough to handle all projects. (It gives a "yes" or "no" answer.). Please show how we can use the algorithm A(P, m, C, n, k) to select the smallest number k of companies to finish all m projects so we can maximally save our budget. You need to output the companies that have been selected by your method (algorithm).

Algorithm 2 An algorithm that uses the given function to find the minimum cost. This algorithm assumes that the cost of each company is the same.

```
function FindMinCost(P, m, C, n)
    k ← n
    repeat
        k ← k − 1
    until A(P, m, C, n, k) equals NO
    k ← k + 1
    while sizeOf(C) > 0 do
        current ← pop(C)
        if A(P, n, C, n, k) equals NO then
            C ← current ⋃ C
            break from while
        end if
    end while
    return C
end function
```

This algorithm starts off by giving k a value of n. The problem statement states that n companies will be able to complete all of the projects so we know the result of a call to A will be YES giving us the ability to skip that call. Each iteration of the repeat decrements k until the call to A returns No at which point the loop breaks. When this occurs, the current value of k received a No so we can assume that the smallest k would be the previous k value; k + 1. Therefore, k is assigned k + 1.

The while-loop following that assignment ensures that companies are still present in C. If there are companies present, a next company is popped off of C at which point another call to A is done to determine if removal of that company from C resulted in No. During the while portion, k is kept constant while the elements of C vary. When the call to A is done and results in no in the while-loop, we can assume that the previously eliminated company was vital and needs to be in the solution.