Homework 11 - Computer Science 461 **Name:** _____

Due Monday, April 21.

1. Recall that a graph G is **bipartite** if it is 2-colorable, i.e., the vertices of G can be colored either white or black so that every edge connects one black vertex with one white vertex. Let

$$\text{BIPARTITE} = \{\langle G \rangle : G \text{ is a bipartite graph}\}.$$

   Prove that BIPARTITE is in class NP by describing what counts as a **solution** for a graph G and describing a **verifier** algorithm that can confirm the solution is correct in polynomial time.

2. A graph is **connected** if there is a path from any vertex to any other. Let

$$\text{CONNECTED} = \{\langle G \rangle : G \text{ is a connected graph}\}.$$

   Consider the following algorithm to decide whether a graph is connected:

   - **Step 1.** Select the first vertex of G and mark it.
   - **Step 2.** Loop through the edges of G. For any edge that touches one marked vertex, mark the other vertex it touches.
   - **Step 3.** Repeat step 2 until you don't find any more vertices to mark.
   - **Step 4.** Loop through all of the vertices and check that they are marked. Reject if any are not marked, otherwise accept.

   Determine the run time of the algorithm in big-O notation as a function of the number of vertices (n) in the graph. Hint: What is the maximum number of edges a graph with n vertices can have?

3. The Kleene star of a language is the set $L^* = \{w_1 w_2 \dots w_k : k \in \mathbb{N} \text{ and each } w_i \in L\}$. The following algorithm shows that if L is a language in class P, then so is L^*.

```
-------------------------------------------------------------------------------
# Given an input string w, determine whether w is in L* (the Kleene star of L).

n = length(w)
table = [0] # table is a list to store indices k such that w[0:k] is in L*.
            # 0 is always in table since the empty string is in L*.

for k from 1 to n:
    for j from 0 to k-1:
        if (j in table) and (w[j:k] in L):
            add k to table

if n in table:
    return TRUE
else:
    return FALSE
-------------------------------------------------------------------------------
```

If L can be decided in $O(n^p)$ time, then what is the big-O run time for the algorithm above?

4. Given two strings $a, b \in \{0,1\}^*$, can we decide if a and b are equal in polynomial time? Explain why or why not.