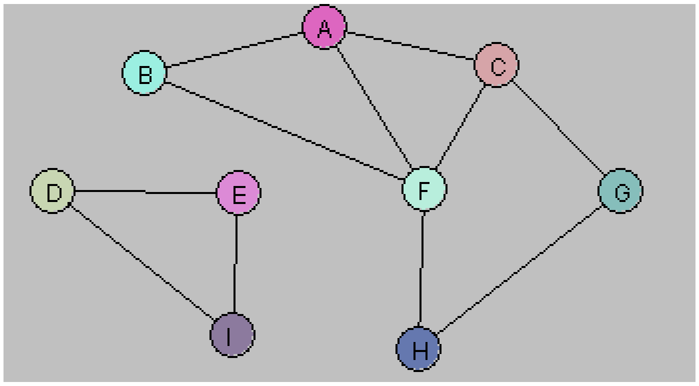
# Lab 8

1. *Induced Graphs.* Answer questions about the graph G = (V,E) displayed below.

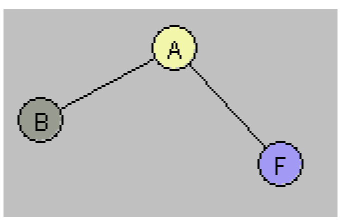



A. Let U = {A, B}. Draw G[U].

B. Let W = {A, C, G, F}. Draw G[W].

C. Let Y = {A, B, D, E}. Draw G[Y].

D. Consider the following subgraph H of G:




Is there a subset X of the vertex set V so that H = G[X]? Explain.

E. Find a way to partition the vertex set V into two subsets $V_1$, $V_2$ so that each of the induced graphs $G[V_1]$ and $G[V_2]$ is connected and $G = G[V_1] \cup G[V_2]$.

2. *Graph Implementation.* Use the BFS class to solve the following problems. Implement by implementing the unimplemented methods in the Graph class.

- Given two vertices, is there a path that joins them?
- Is the graph connected? If not, how many connected components does it have?
- Does the graph contain a cycle?

*Hint:* For the third problem, you may use the following Fact (which we will prove in tomorrow's class):

**Fact**: Suppose G is a graph with $n$ vertices and $m$ edges and $m = n - 1$. Then $G$ is acyclic if and only if $G$ is connected.

3. *Graph Exercises.*

A. Suppose G = (V, E) is a connected simple graph. Suppose $V_1, V_2, \ldots, V_k$ are disjoint subsets of V and that $V_1 \cup V_2 \cup \ldots \cup V_k = V$. Show that there is an edge (x,y) in E such that for some $i \neq j$, x is in $V_i$ and y is in $V_j$.

B. In class it was shown that a graph G = (V, E) is connected whenever the following is true,

$$\epsilon > \binom{\nu - 1}{2} \quad (*)$$

where $\nu$ is the number of vertices and $\epsilon$ is the number of edges. Is the following true or false?

*Every connected graph satisfies the inequality* (*).

Prove your answer.

C. Suppose G is a graph with two vertices. What is the minimum number of edges it must have in order to be a connected graph? Suppose instead G has three vertices; what is the minimum number of edges it must have in order to be connected? Fill in the blank with a reasonable conjecture:

If G has n vertices, G must have at least _____ edges in order to be connected.

4. *IsPrime Problem, Revisited.* The goal of this exercise is to devise a feasible algorithm that decides whether an input integer is prime. The key fact that you will make use of is the following:

Fact: There is a function $f$, which runs in $O(\log n)$ (that is, O(length($n$))), such that for any odd positive integer $n$ and any $a$ chosen randomly in $[1, n - 1]$, if $f(a, n) = 1$, then $n$ is composite, but if $f(a,n) = 0$, $n$ is "probably" prime, but is in fact composite with probability $< ½$.

A first try at such an algorithm would be:

*Algorithm* **FirstTry**:

*Input:* A positive integer n
*Ouptut:* TRUE if n is prime, FALSE if n is composite

```
if n % 2 = 0 return FALSE
a ← random number in [1, n-1]
if f(a,n) = 1
   return FALSE
return TRUE
```

Notice that **FirstTry** runs in O(log *n*). It also produces a correct result more than half the time.

What could be done to improve the degree of correctness of **FirstTry** but still preserve a reasonably good running time? Explain.