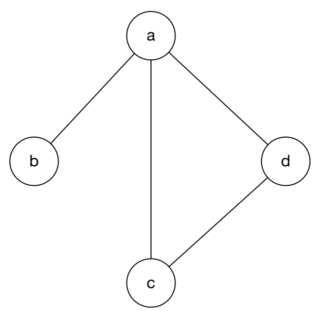
CS 261, HW9

Prof. Jared Saia, University of New Mexico

Due: May 2nd

1. A *graph* consists of a set of vertices, V, and a set of edges E, where each edge in E is a tuple of 2 vertices in V. For example, we could have $V = \{a, b, c, d\}$ and $E = \{(a, b), (c, d), (a, d), (a, c)\}$. The *degree* of a vertex $v \in V$ is the number of edges containing v. In the example graph, the degree of a is 3, and the maximum degree of the graph is 3.

   A *coloring* of a graph is an assignment of colors to all vertices in V such that for all $(u, v) \in E$, vertex u has a different color than vertex v. A coloring of the example graph with 3 colors is: vertex a gets color 1, vertex b gets color 2, vertex c gets color 2, and vertex d gets color 3. Prove the following.

   - Any graph with maximum degree of 3 can be colored with 4 colors. Prove this by induction over $n = |V|$. For your IH, assume that any graph with maximum degree of 3 and **less than** n vertices can be colored with 4 colors.

2. We say that a graph $G = (V, E)$ is *connected* if for every pair of vertices $u, v \in V$, there exists a path from u to v using only edges in G. To illustrate, the example graph in Problem 1 is connected, but becomes

disconnected if we remove edge (a,b). For a graph $G = (V,E)$ and vertex $v \in V$, we define $G - v$ to be the new graph $(V - v, E')$ where E' is E minus all edges containing v. Prove the following:

- Every connected graph $G = (V,E)$ with $|V| \geq 2$ has at least two vertices x_1 and x_2, such that $G - x_i$ is connected for $i = 1,2$. Prove this by induction on $n = |V|$. For your IH, assume that any graph with $2 < |V| < n$ has at least two vertices x_1 and x_2, such that $G - x_i$ is connected for $i = 1,2$.

3. Prove by induction that $\sum_{i=0}^{n} r^i = \frac{r^{n+1}-1}{r-1}$ for every $n > 0$, and all $r \neq 1$. For your IH, assume that $\sum_{i=0}^{n-1} r^i = \frac{r^n-1}{r-1}$.

4. Consider the recurrence $f(n) = 3f(n/2) + \sqrt{n}$, where $f(c) = \theta(1)$, for constants c. Use the Master method to solve this recurrence.

5. Consider the following function:

```
int f (int n){
  if (n==0) return 2;
  else if (n==1) return 5;
  else{
    int val = 2*f (n-1);
    val = val - f (n-2);
    return val;
  }
}
```

   (a) Write a recurrence relation for the *value* returned by f. Solve the recurrence exactly using annihilators. (Don't forget to check it)

   (b) Write a recurrence relation for the *running time* of f. Get a tight upperbound (i.e. big-O) on the solution to this recurrence using annihilators.