CS200 - Problem Set 8

Due: Monday, Nov. 13 to submission server before class

Please read the sections of the syllabus on problem sets and honor code before starting this homework.

1. **[11 points]** Prove $x^2 - 5x + 100 = \Theta(x^2)$.

2. **[6 points]** Suppose I have an algorithm whose runtime on inputs of size 1, 2, 3, 4, and 5 are

Input Size	1	2	3	4	5
Runtime	1	1	1	1	1

   Explain how this algorithm could have an asymptotic runtime of $\Omega(2^n)$.

3. In the following, you may assume that the graph (V, E) is undirected and does not have self loops or multi-edges. Let $deg(v)$ be the degree of a vertex v. Rewrite each predicate using math.

   (a) **[3 points]** $D(v, u, (V, E)) \equiv$ In the graph (V, E) there is a path of length 2 from vertex v to vertex u.

   (b) **[3 points]** $R(v, (V, E)) \equiv v$ is the vertex with the smallest degree in the graph (V, E)

   (c) **[3 points]** $W(V, E) \equiv$ There is a vertex in the graph (V, E) that is not connected to any other vertices.

   (d) **[3 points]** $M(V, E) \equiv$ There is a vertex in the graph (V, E) that is connected to all other vertices.

   (e) **[3 points]** $T(V, E) \equiv$ All vertices in the graph (V, E) have the same degree.

   (f) **[3 points]** $K(V, E) \equiv$ All vertices in the graph (V, E) have even degree.

   (g) **[3 points]** $R(V, E) \equiv$ The graph (V, E) is bipartite, which means there are two sets of vertices A, B that are not empty, and such that there are not any edges between two vertices in A, and there are not any edges between two vertices in B.

   (h) **[3 points]** $L(V, E) \equiv$ The graph (V, E) is bipartite with sets A and B, and all vertices in A are connected to all vertices in B.

4. Graph Search. In this problem we will use the graph described by the following adjacency list:

A :

vertex	adjacency list
s	u, y
u	v, z, s
y	s, z
v	w, a, u
z	w, u, y
a	v, t
w	v, t, z
t	a, w

We will also use the algorithm `DepthFirstSearch`, which is a version of the Graph Search algorithm we saw in class. It's pseudocode is:

Algorithm 1: `DepthFirstSearch`(A, X, s, f)

Input : Adjacency list A for a graph $G = (V, E)$, an array X of length $|V|$ such that $X[v] = 1$ if v has been explored and 0 otherwise, a starting vertex s, a goal vertex f

Output: String "f found!" or "f not found" depending on whether f can be found from s.

```
1  if s == f then
2  |  Return "f found!";
3  else
4  |  X[s] = 1;
5  |  d = A[s].length;
6  |  for k = 1 to d do
7  |  |  if X[A[s, k]] == 0 then
8  |  |  |  DepthFirstSearch(A, X, A[s, k], f);
9  |  |  end
10 |  end
11 end
12 Return "f not found"
```

(a) **[6 points]** Please draw the graph the adjacency list corresponds to.

(b) **[6 points]** Suppose you start at s, and want to find t. In what order are vertices explored if you use `DepthFirstSearch`(A, X, s, t)? (Where X is initially set to all zeros)

(c) **[6 points]** Suppose you start at s, and want to find y. In what order are vertices explored if you use `DepthFirstSearch`(A, X, s, y)? (Where X is initially set to all zeros)?

(d) **[6 points]** Qualitatively describe how `DepthFirstSearch` explores the graph.

(e) **[6 points]** In our generic Graph Search algorithm from class on Monday, we did not decide how to choose the next explored edge, in the case that there were multiple edges we could explore. How does `DepthFirstSearch` choose which edge to explore next?

5. **[6 points]** Suppose you have a weighted coin such that the probability of heads is 0.3 and the probability of tails is 0.7. What is the probability of getting at least 3 heads, if you flip the coin 10 timess?

6. How long did you spend on this homework?