CS601: Algorithms and Complexity **2019-20 Sem I**

Self-assessment Quiz

Total Marks: 80

Note. You are supposed to solve these problems without any discussions with anyone. Try to write your answers succinctly.

Que 1. [20] A graph is called bipartite if its vertex set can be partitioned into two parts, say V_1 and V_2, such that all the edges connect a vertex from V_1 to a vertex from V_2. Let us say we are given a graph by its adjacency matrix. Can you design an algorithm which can test whether the graph is bipartite? Formally argue that your algorithm is correct.

Hint: Start by assuming that the graph is indeed bipartite with V_1, V_2 as the two parts. Pick an arbitrary vertex v, which we can assume belongs to V_1 (why?). Now, any neighbor of v should belong to V_2. Similarly, for any vertex $u \in V_2$, any neighbor of u must belong to V_1. This way you can try to mark all the vertices as either belonging to V_1 or V_2. Now, how would you detect if the graph was not actually bipartite?

You can try to use a depth-first-search or breadth-first-search for exploring the graph.

Que 2. [10] Mark true or false.

- $(2n+1)^2 = O(n^2)$.
- $f(n) = O(g(n)) \implies 2^{f(n)} = O(2^{g(n)})$.

Prove that if $f(n) = O(f(n/2))$ then there is a polynomial function $g(n)$ such that $f(n) = O(g(n))$.

Que 3. [10+10] Part(a). Let G be a directed graph with $N+1$ vertices, for a large number N. Imagine the vertices to be points on the number line. That is, the vertices are indexed $0, 1, 2, 3, \ldots, N$ For each $i \geq 2$, we have two edges going out of the vertex i – one going into $i-1$ and the other into $i-2$. Moreover, there is one edge from 1 to 0. Show that the number of distinct paths from vertex N to vertex 0 is at least $2^{\lfloor N/2 \rfloor}$.

Hint: Observe that there are two paths from vertex i to vertex $i-2$.
It suffices to count only a subset of all paths.

Part (b). Consider the following recursive procedure to compute the fibonacci series.

```
function Fibonacci(n):
  if n = 0 or n = 1: return 1;
  else: return Fibonacci(n-1)+Fibonacci(n-2);
```

Using part (a) show that this procedure takes $\Omega(2^{\lfloor N/2 \rfloor})$ time to compute Fibonacci(N).

Que 4. [10] Prove the following. In a party with 100 people, there must at least two people who have the same number of friends in the party.

Que 5. [10] Consider the following algorithm.

input: a positive number n;
$i \leftarrow 2$;
while($i \leq n$):
  $i \leftarrow 2 \times i$;
end while

The number of iterations the algorithm makes is $O(?)$.

Que 6. [10] Let a class have 25 students. Show that the probability that at least two students in the class will have the same birthday is at least 1/2. (Ignore February 29 and assume that each student got a birthday uniformly randomly and independently from the 365 possible dates.)

Hint: Think about the probability that all of them have different birthdays.