Birla Institute of Technology & Science Pilani, Pilani
Design and Analysis of Algorithms (CS F364) Mid Sem Exam (Second Semester 2019-2020)

There are 3 questions in all and total marks is $(8 + 16) + 16 + 16 = 56$. This is an **open book exam**. Only hard copies of textbooks, reference books, and lecture notes are allowed. No electronic instruments (calculator, mobile phone, tablet, laptop etc.) are allowed. Show all computation steps for solving any problem, and write complete algorithms with time complexity derivations. Time: 90 minutes.

1. (a) The *chirp transform* of a vector $a = (a_0, a_1, \ldots, a_{n-1})$ is the vector $y = (y_0, y_1, \ldots, y_{n-1})$, where

$$y_k = \sum_{j=0}^{n-1} a_j z^{kj}, \quad \forall k \in [0 \mathinner{.\,.} n-1],$$

and z is any complex number. The DFT is therefore a special case of the chirp transform, obtained by taking $z = \omega_n$. Using the equation

$$y_k = z^{\frac{k^2}{2}} \sum_{j=0}^{n-1} \left(a_j z^{\frac{j^2}{2}} \right) \left(z^{-\frac{(k-j)^2}{2}} \right), \quad \forall k \in [0 \mathinner{.\,.} n-1],$$

to view the chirp transform as a convolution, design an efficient algorithm using FFT and convolution to evaluate the chirp transform in time $O(n \log n)$ for any complex number z.

(b) Using your algorithm in part (a) above, evaluate the chirp transform of the vector $a = (1, 2, 3, 4)$ at $z = 2i$ showing your computations in divide and conquer graphs.

2. Professor F. Lake suggests the following algorithm for finding the shortest path from node s to node t in a directed graph with some negative edges (with no negative cycles):

   Add a large constant to each edge weight so that all the weights become positive, then run Dijkstra's algorithm starting at node s, and return the shortest path found to node t.

   Is this a correct algorithm? Either prove that it works correctly, or give a counterexample.

3. Given an unlimited supply of coins of denominations $x_1, x_2, \ldots x_n$, we wish to make change for a value v; that is, we wish to find a set of coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 then we can make change for 15 but not for 12. Design an $O(nv)$-time dynamic-programming algorithm for the following problem:

   Input: $x_1, \ldots x_n; v$.
   Question: Is it possible to make change for v using coins of denominations $x_1, \ldots x_n$?