Problem 1. To prove that the \( (n, \frac{n(n+1)}{2}) \)-assignment object has consensus number at least \( n \), we just have to devise a consensus algorithm for \( n \) processes. The \( (n, \frac{n(n+1)}{2}) \)-assignment object has \( \frac{n(n+1)}{2} \) fields. For convenience we name the fields as follows. There are \( n \) fields \( r_0, \ldots, r_{n-1} \) where process \( i \) writes to register \( r_i \), and \( \frac{n(n-1)}{2} \) fields \( r_{ij} \), where \( i > j \), where processes \( i \) and \( j \) both write to field \( r_{ij} \). All fields are initialized to \textit{null}. Each process \( i \) atomically assigns its input value to \( n \) fields: its single-writer field \( r_i \) and its \( n-1 \) multi-writer registers \( r_{ij} \). For example, if \( n = 3 \), process 1 will write to single-writer register \( r_1 \) and to multi-writer registers \( r_{10} \) and \( r_{21} \). The algorithm decides the first value to be assigned. After assigning to its fields, a thread determines the relative ordering of the assignments for every two processes \( i \) and \( j \) as follows:

- Read \( r_{ij} \). If the value is \textit{null}, then neither assignment has occurred.
- Otherwise, read \( r_i \) and \( r_j \). If \( r_i \)'s value is \textit{null}, then \( j \) precedes \( i \), and similarly for \( r_j \).
- If neither \( r_i \) nor \( r_j \) is null, reread \( r_{ij} \). If its value is equal to the value read from \( r_i \), then \( j \) precedes \( i \), else vice versa.

Repeating this procedure, a process can determine which value was written by the earliest assignment.

This described algorithm is taken from the book “The Art of Multiprocessor Programming.” The interested student can also have a look at Section 3.6 of the “Wait-free Synchronization” paper on the consensus number of the \( (m, n) \)-assignment object.