1. To prove the lower bound in the question, we can view the relevant randomized protocol as a distribution over deterministic protocols, where for each deterministic protocol the public random string is fixed. Since we only allow zero error, each of these deterministic protocols must calculate f exactly.

   Assume that f has a fooling set of size t. It follows that any determenistic protocol computing f has its communication complexity bounded below by $\log t$. Combining this assumption with the idea from the previous paragraph, we can say that whatever deterministic protocol we pick from our distribution, it is guaranteed to have communication complexity $D(f) \geq \log t$. Hence we can conclude that $R_0^{\text{pub}}(f) \geq \log t$.

   An example of a function that has an exponential gap between R_0^{pub} and $R_{1/3}^{\text{pub}}$ is the EQ_2. If the outputs of f can be encoded in a matrix of size $n \times n$, we know that the only values for which EQ_2 evaluates to 1 are the values on the diagonal, and they can be used as the fooling set. Since the size of this fooling set is n, we know that $R_0(\text{EQ}_2) \geq \log n$ (by argument above).

   On the other hand, if we're allowed to be wrong on 1/3 fraction of inputs, our protocol can pick a part of the public random string, $r \in \{0,1\}^n$. Then, Alice can calculate the dot product of r and her value a (mod 2), send the resulting 1 bit over to Bob, and let Bob take the same inner product and compare the values. If the actual values a and b held by Alice and Bob are equal, this will produce the correct result with probability 1. If the values were not equal, this approach can produce an incorrect result with probablity 1/2. To satisfy the error bound of 1/3, we can repeat this process $k \geq 2$ times for some fixed k, each time using a different substring of the random string, halving the probability of error. Since k is constant, the communication complexity is $O(1)$, and hence we have an exponential decrease from R_0^{pub}.

2. TRIBES function requires that x and y share at least one 1 on every row of the matrix. Solutions:

   (a) The most straightforward way to solve TRIBES(x, y) is for Alice to just send over her entire matrix, which requires n bits, and wait for Bob's answer, giving overall communication complexity $O(n)$. This proves the upper bound for the problem.

   The lower bound can be proved by showing a 0-fooling set of size 2^n. Consider a $\sqrt{n} \times \sqrt{n}$ matrix consisting of 1's, call it M. We can pick a row i and a column j and define two new matrices $M_{i,j}^0$ and $M_{i,j}^1$. $M_{i,j}^0$ will have the same structure as M, except on row i it will have 0's everywhere but the jth column. $M_{i,j}^1$ will also have the same structure as M, but on row i it will have 1's everywhere except the jth column.

   Note that TRIBES$(M_{i,j}^0, M_{i,j}^1) = 0$ because $M_{i,j}^0$ and $M_{i,j}^1$ have different boolean values at each position of row i. We can now define a fooling set S:

   $$S = \{(M_{i,j}^0, M_{i,j}^1) : i, j \in \{1, \ldots, \sqrt{n}\}\}$$

   S is a fooling set because none of the pairs in S can belong to the same 0-rectangle. We can show this by taking two distinct pairs $(M_{i,j}^0, M_{i,j}^1), (M_{p,q}^0, M_{p,q}^1) \in S$ such that $p \neq i$ or $q \neq j$ (or both). When $p \neq i$, $M_{i,j}^0$ and $M_{p,q}^1$ (or vice versa) have 0's on different rows, so in each row both matrices have a 1 in at least one position, and hence TRIBES will evaluate to 1. When $q \neq j$, $M_{i,j}^0$ and $M_{p,q}^1$ (or vice versa) have 0's in the same column, but because of the $q \neq j$ mismatch they will still share exactly one 1 element on that row. Therefore TRIBES will evaluate to 1.

   Now we can consider the size of S. We can flatten out each array M^* into a row vector of size n. Note that by definition of our $M_{i,j}^1$, the flat version of $M_{i,j}^1$ will have 1's everywhere but the $(i\sqrt{n} + j)$th index, where it will have a 0. By this simple observation we can conclude that there.

(b) By definition of the problem, to show that TRIBES$(x, y) = 1$ we only need to find one index j on every row i such that $x_{i,j} = y_{i,j} = 1$. Therefore an all powerful prover just needs to point out these indices on each row. There are $\sqrt{n}$ rows, and each column index takes $\log_2(\sqrt{n})$ bits to represent, giving us an overall bit count of $\sqrt{n} \cdot \frac{1}{2}\log_2 n$. Alice can send over these values to Bob and wait for him to confirm that there is a match. Hence the upper bound on N^1(TRIBES) is $O(\sqrt{n} \log n)$.

We can use the fooling set technique to show the lower bound. Consider a $1 \times \sqrt{n}$ vector e_i^T, which has zeros everywhere except the ith position. Consider a matrix x where each row is a vector e_{i_k} for some $i_k \in \{1, \ldots, \sqrt{n}\}$. Clearly, TRIBES$(x, x) = 1$ because x has at least one 1 on each row. At the same time, if we permute any of the rows of x by shifting the 1 in that row left or right and define the new matrix x^*, we'll see that TRIBES$(x, x^*) = 0$ because there is now at least one row where x and x^* do not match up. We can exploit this to generate a fooling set. Let M be the set of matrices that start off as an identity matrix, but have either 1 within some row shifted left or right, or have some rows swapped, or both. Then we can define the fooling set S as:

$$S = \{(M^*, M^*) : \text{distinct } M^* \in M\}$$

Clearly, this a 1-fooling set because for any two distinct pairs $(M_1, M_2), (M_1^*, M_2^*) \in S$, neither (M_1, M_2^*) nor (M_1^*, M_2) can be evaluated to 1 since they must differ in at least one place (we can show by contradiction that if this is not the case, the pairs must be identical). Now for the size of this fooling set: there are $\sqrt{n}$ rows in total, and for each row we have $\sqrt{n}$ different choices for the vector e_i^T. This means that $|S| = \sqrt{n}^{\sqrt{n}}$, and we can obtain a lower bound $N^1(\text{TRIBES}) \geq \Omega(\log\left(\sqrt{n}^{\sqrt{n}}\right))$, or, equivalently, $N^1(\text{TRIBES}) \geq \Omega(\sqrt{n} \log n)$.

Combining the lower and upper bound, we get $N^1(\text{TRIBES}) = \Theta(\sqrt{n} \log n)$.

(c) To show that TRIBES$(x, y) = 0$, we need to find a single row i where in each position j we have $x_{i,j} \wedge y_{i,j} = 0$. Our all powerful prover can identify this row and show its index to Alice and Bob. Then, Alice can send Bob a bitmask of said row, taking up $\sqrt{n}$ bits in total. Including the 1 bit of Bob's reply, we get the overall complexity of $O(\sqrt{n})$, proving the upper bound for N^0(TRIBES).

The lower bound for N^0 can be proved by showing a 0-fooling set of size $2^{\sqrt{n}}$. Consider a $\sqrt{n} \times \sqrt{n}$ matrix consisting of 1's, call it M. We can pick a row i and a column j and define two new matrices $M_{i,j}^0$ and $M_{i,j}^1$. $M_{i,j}^0$ will have the same structure as M, except on row i it will have 0's everywhere but the jth column. $M_{i,j}^1$ will also have the same structure as M, but on row i it will have 1's everywhere except the jth column.

Note that TRIBES$(M_{i,j}^0, M_{i,j}^1) = 0$ because $M_{i,j}^0$ and $M_{i,j}^1$ have different boolean values at each position of row i. We can now define a fooling set S:

$$S = \{(M_{i,j}^0, M_{i,j}^1) : i, j \in \{1, \ldots, \sqrt{n}\}\}$$

3. CIS$_G$ solutions:

(a) We can prove the lower bound $\Omega(\log n)$ by finding a fooling set of size n for some G. Consider a complete graph G with n nodes. We can generate a clique-independent-set pair (C, I) by picking a single node x^* and defining $I = \{x^*\}$, then putting the remaining $n - 1$ nodes into the clique C. Clearly, I is an independent set because it only has one node, and C is a clique because any subgraph of a complete graph is also complete.

Since there are n nodes in total, we can define n distinct clique-independent-set pairs using this procedure. Putting them all into a set S we get a 0-fooling set: for any pair $(C, I) \in S$, $\text{CIS}_G(C, I) = 0$, but for any two distinct pairs $(C, I), (C^*, I^*) \in S$ we have $\text{CIS}_G(C, I^*) = 1$ and $\text{CIS}_G(C^*, I) = 1$ (by definition of pairs in S).

It follows that the deterministic complexity for CIS_G with our choice of G is bounded below by $\Omega(\log |S|)$, or, equivalently, $\Omega(\log n)$.

(b) The question wants us to prove that $D(\text{CIS}_G) \leq O(\log^c n)$ implies $D(f) \leq O(\log^c C^D(f))$ for an arbitrary choice of a boolean function f.

Let M_f be the matrix corresponding to outputs of f. We know that there exists some disjoint cover of M_f using monochromatic rectangles. Let C^* be the smallest such cover. By definition, $C^D(f) = |C^*|$. Now, consider only the 1-rectangles from C^*, denoting that set as C_1^*. Note that $|C_1^*| \leq |C^*|$. We can produce a graph G that encodes our problem using the following procedure. Start off with a disconnected graph G that has a node corresponding to every 1-rectangle in C_1^*. Next, for every row x_i in M_f, if said row intersects some 1-rectangles from C_1^*, connect these rectanges into a clique in G. Repeat until we've considered all rows from M_f. The resulting G only depends on the function f (and not its inputs), so it can be encoded as a part of the protocol. As such, it is known to both Alice and Bob beforehand.

Now, assume the input for function f is (x, y), where Alice holds x and Bob holds y. Alice looks at row x in M_f, and defines her clique to be all 1-rectangles from C_1^* that intersect row x. Bob looks at row y and defines his independent set to be all 1-rectangles that intersect row y. Note that this set is guaranteed to be indepdendent because all nodes in G were initially disconnected, and we've only connected 1-rectangles that intersected a common row. Since our rectangle cover is disjoint, rectangles that intersect some common row cannot also intersect a common column.

At this point we have an instance of CIS_G, where G is the graph we generated earlier and the inputs are the clique and the independent set picked by Alice and Bob respectively. By our assumption, there exists a protocol that computes CIS_G in $O(\log^c n)$ bits, where n is the size of the graph. Since our graph G has $|C_1^*| \leq C^D(f)$ nodes, we can safely say that there exists a protocol that computes our instance of CIS_G in $O(\log^c C^D(f))$ bits. We can interpret the output as follows: if we get a 1, then x and y both hit the same 1-rectangle, which is only possible when $f(x, y) = 1$. On the other hand, if the output is 0, then x and y don't share any 1-rectangles, so $f(x, y) = 0$. Since we didn't communicate any extra bits beyond solving the CIS_G problem, we can conclude that $D(f) \leq O(\log^c C^D(f))$.