1. (20 points) Identify all of the following statements that are true about transactions and processes of various kinds

A. Two-phase commit is biased in favor of the *No* decision: even if one party out of one hundred says no, the decision is necessarily a *No*

   **Solution:** A is true:

B. In practical settings, we can ensure compositional serializability across independent services (such as for making bookings on an airline and a hotel) by employing approaches such as tickets

   **Solution:** B is false: it is possible to ensure compositional serializability by employing approaches such as tickets—though such approaches are not practical because they would either cause multiple composed transactions to be aborted or cause one site to await progress at another site

C. Serial schedules yield poor performance but otherwise provide an excellent basis for correctness of executions of services in open environments

   **Solution:** C is false: serial schedules oppose collaboration—where there is naturally some (or, rather, a lot of) back and forth among the constituent computations

D. If a transaction $T_1$ occurs entirely before a transaction $T_2$ in some schedule, then it is still possible that in an equivalent serial schedule $T_2$ precedes $T_1$

   **Solution:** D is true: a particularly simple situation is when $T_1$ and $T_2$ are mutually independent

E. Using binary mode locks, we can realize any schedule, serializable or otherwise

   **Solution:** E is true: as we saw in class, locks in themselves are inadequate to prevent nonserializable schedules

F. Two-phase locking over binary mode locks avoids deadlocks

   **Solution:** F is false: deadlocks can arise between two transactions where each needs a resource that the other is locking

G. If transaction $T_1$ writes item $x$ but doesn’t occur entirely before a transaction $T_2$ that reads $x$, then $T_2$ may still read $x$ from $T_1$

   **Solution:** G is true: if $T_1$ commits before $T_2$ reads $x$, even if $T_1$ and $T_2$ are interleaved for a part of the schedule

H. A compensating transaction seeks to semantically undo the effect of a transaction that has committed but which under the circumstances we would prefer hadn’t committed

   **Solution:** H is true:

I. An orchestration views a process in a naturally distributed manner

   **Solution:** I is false: centralized in terms of where the orchestrated process executes, i.e., at the site of the orchestration engine
J. ebXML involves ways for business partners to specify their collaboration profiles and ways for a business to select partners based on such specifications

Solution: J is true:

2. (10 points) Identify true statements from among the following about events, computations, and guards

A. We require that any significant event or its complement event must occur on any possible enactment

Solution: A is true: that is how we treat complement events

B. Residuation is an operation that takes a dependency and an event and outputs a dependency

Solution: B is true:

C. Given that event $e$ is immediate, let us map some arbitrary dependency $D$ to dependency $Z$; then either $Z \equiv 0$ or $Z/e = \top$

Solution: C is false: let $D = e \cdot f \lor \bar{f} \cdot e$; then $Z = D$ and $Z/e \neq \top$

D. Given that event $e$ is immediate, we map dependency $D$ to dependency $F$ where $F \neq 0$; then $G(D, e)$ must differ from $G(F, e)$

Solution: D is false: for example (from class notes), let $D = \bar{e} \lor \bar{f} \lor e \cdot f$, which reduces to $F = \bar{e} \cdot f \lor \bar{f} \lor e \cdot f$ when $e$ is immediate; $G(D, e) = G(F, e) = \neg f$

E. “Neither $e$ nor $\bar{e}$ may occur whether or not $f$ does” may be expressed as $f$

Solution: E is false: this dependency equals 0 because exactly one of $e$ and $\bar{e}$ must occur regardless of anything else; in contrast, $f$ means that $f$ must occur whereas the desired dependency is talking about “whether or not $f$” occurs