These "C puzzle" problems provide a clear demonstration that programmers must understand the properties of computer arithmetic:

A. `(x > 0) || (x-1 < 0)`
False. Let `x` be −2,147,483,648 ($TMin_{32}$). We will then have `x-1` equal to 2,147,483,647 ($TMax_{32}$).

B. `(x & 7) != 7 || (x<<29 < 0)`
True. If `(x & 7) != 7` evaluates to 0, then we must have bit x_2 equal to 1. When shifted left by 29, this will become the sign bit.

C. `(x * x) >= 0`
False. When `x` is 65,535 (`0xFFFF`), `x*x` is −131,071 (`0xFFFE0001`).

D. `x < 0 || -x <= 0`
True. If `x` is nonnegative, then `-x` is nonpositive.

E. `x > 0 || -x >= 0`
False. Let `x` be −2,147,483,648 ($TMin_{32}$). Then both `x` and `-x` are negative.

F. `x+y == uy+ux`
True. Two's-complement and unsigned addition have the same bit-level behavior, and they are commutative.

G. `x*~y + uy*ux == -x`
True. `~y` equals `-y-1`. `uy*ux` equals `x*y`. Thus, the left-hand side is equivalent to `x*-y-x+x*y`.