• On line 12, we initialize startingPos[0] to 1. By the description of startingPos[], and since we know that the non-zero elements of b[] are stored starting from index 1, this initialization is correct.

Loop invariant of Lines 13-14: Just before the start of the ith iteration of the loop, for all indices K, where $1 \leq K < i < numCols$, the entry startingPos[K] stores the index of the triple $\langle b[k].row, b[x].col, b[x].value \rangle$, where startingPos[K] = x, $b[x].row = K$, $b[x].col = K_1$ and $\forall x'$, where $x < x'$, if $b[x'].row = K$, then $b[x'].col = K_2 \geq K_1 = b[x].col$. Or in other words, startingPos[K] stores that non-zero element of b[] having row K and minimum indexed column. Hence, startingPos[K] has been computed correctly.

Rewritten (cleaner and Equivalent) Version by ChatGPT

Just before the start of the ith iteration of the loop, for all indices K where $0 \leq K < i$, startingPos[K] correctly stores the index in array b[] where the first non-zero entry of row K in the transposed matrix should be placed. This ensures that all future elements belonging to row K in b[] will be inserted starting from this position in increasing column order.

Initialization: At the start of the 1st iteration of the loop, i=1. Just before the start of this iteration, for all indices $0 \leq K < i = 1$, i.e. only for index 0, we have already justified before why startingPos[0] being initialized to 1 is correct.

∴ startingPos[0] correctly stores the index in array b[] where