A. [Because I sometimes assign this problem as homework, I'd rather not post a solution. I'll happily show you my solution in person. In grading, I tested each student's code against three examples: Fibonacci numbers and combinatorial coefficients, which were in the tutorial, and depth-first search, which was not.]

B.A. The start variable is S. The comments at right should aid comprehension. [Notice that each variable has a clearly defined role or meaning. Wherever possible, the names of the variables are chosen to help the reader remember their meanings.]

$S \to TbTaT \mid TcTbT \mid TcTaT$	strings out of order (not in $L(a^*b^*c^*)$)
$T \to \epsilon \mid aT \mid bT \mid cT$	all strings over $\{a, b, c\}$
$S \to aA\binom{A}{B}C$	strings in order but more as than bs
$S \to \binom{A}{B}bBC$	strings in order but fewer as than bs
$S \to AbB\binom{B}{C}$	strings in order but more bs than cs
$S \to A\binom{B}{C}cC$	strings in order but fewer bs than cs
$\binom{A}{B} \to \epsilon \mid a\binom{A}{B}b$	strings of the form $a^m b^m$
$\binom{B}{C} \to \epsilon \mid b\binom{B}{C}c$	strings of the form $b^m c^m$
$A \to \epsilon \mid aA$	strings of the form a^m
$B \to \epsilon \mid bB$	strings of the form b^m
$C \to \epsilon \mid cC$	strings of the form c^m

B.B. No, the argument does not work. The problem is that PDAs are non-deterministic. Suppose that, when a PDA P processes an input w, some of its computational branches accept and some reject. Let N be the negation of P as described in the argument. Then, on input w, some of N's branches reject and others accept. So both P and N accept w. So $L(N) \neq L(P)^c$.

[Also, problem B.A shows that the class of context-free languages is not closed under complementation. But that does not explain why the proposed argument fails.]

B.C. No, the argument does not work. The problem is that the product PDA would have to simulate two stacks using its single stack. It is not clear how to do this.

[In fact, the class of context-free languages is not closed under intersection. But that does not explain why the proposed argument fails. Also, a later homework problem gives us strong reason to doubt that a PDA can simulate two stacks at once. Of which problem am I thinking?]

C.A. Assume for the sake of contradiction that A is context-free. Let p be the pumping length for A guaranteed by the context-free pumping lemma. Let $w = 0^{p^2}$. Then the pumping lemma guarantees the existence of strings u, v, x, y, z satisfying certain properties. In particular, because $|vxy| \leq p$, and $|vy| \geq 1$, we can conclude that $vy = 0^\ell$ for some ℓ such that $1 \leq \ell \leq p$.

Then $uv^2xy^2z = 0^{p^2+\ell}$, and

$$\begin{aligned} p^2 &< p^2 + 1 \\ &\leq p^2 + \ell \\ &\leq p^2 + p \\ &< p^2 + 2p + 1 \\ &= (p+1)^2. \end{aligned}$$

Because $p^2 + \ell$ is strictly between two consecutive squares, it cannot itself be a square. So $uv^2xy^2z \notin A$. This contradiction implies that A is not context-free.

C.B. Intuitively, we try $m = 0, 1, 2, \ldots$ until the m being tried is too large. Let's use three tapes. Upon startup, the first tape holds the input, and the other tapes are empty. On each pass through the loop, the first tape holds the input, the second tape holds m^2 0s, and the third tape holds m 0s. The third tape helps update the second tape, which is compared to the first tape. In detail, the algorithm is:

1. Wrap the contents of all three tapes in turnstiles.
2. Repeat:
   (a) Scan the first and second tapes left-to-right. If they are equal in content, then accept. If the second tape has more 0s, then reject. If the first tape has more 0s, then continue.
   (b) Starting from the end of the second tape and the beginning of the third tape, scan right, appending the third tape to the second tape. (Now the second tape has m^2+m 0s.)
   (c) Append the third tape to the second tape. (Now the second tape has $m^2 + 2m$ 0s.)
   (d) Increment the second tape by appending a 0 to it. (Now the second tape has $m^2 + 2m + 1 = (m+1)^2$ 0s.)
   (e) Increment the third tape. (Now the third tape has $m + 1$ 0s.)

D. The pumping length $p = 4$ suffices. Let $w \in A$ be a string such that $|w| \geq p$. Then $w = a^m b^m c^n$ where $m \neq n$ and $2m + n \geq 4$. There are four cases:

- Suppose that $m \geq n + 2$. Set $v = ab$ and $x = y = \epsilon$. Then u and z are determined. For all $i = 0, 1, 2, \ldots$, $uv^ixy^iz = a^{m-1+i}b^{m-1+i}c^n$, which is a string in A because $m - 1 + i \geq m - 1 > n$.

- Suppose that $m = n + 1$. Set $v = aabb$ and $x = y = \epsilon$. Then u and z are determined. For all i, $uv^ixy^iz = a^{m-2+2i}b^{m-2+2i}c^n$, which is a string in A because $m - 2 + 2i$ and n have different parity.

- Suppose that $m = n-1$. Set $v = cc$ (anyhow) and $x = y = \epsilon$. Then u and z are determined. For all i, $uv^ixy^iz = a^mb^mc^{n-2+2i}$, which is a string in A because m and $n - 2 - 2i$ have different parity.

- Suppose that $m \leq n-2$. Set $v = c$ (anyhow) and $x = y = \epsilon$. Then u and z are determined. For all i, $uv^ixy^iz = a^mb^mc^{n-1+i}$, which is a string in A because $n - 1 + i \geq n - 1 > m$.

In all four cases, we have shown that the chosen w can be pumped and remain in the language A. Therefore A satisfies the conclusions of the pumping lemma.

[I think that $p = 3$ works too. All cases other than the second one seem safe. The second case also survives. Is that right?]