As with context-free grammars, derivations for general grammars involve finding occurrences of the left side of a rule and replacing them with the right side of a rule. See the derivations examples from class on 4/19 — the answer shows just the derivation (note the notation!) but the derivation also shows what rules are being applied in each step.

For "explain how the grammar works", addressing each rule's role is a good start but also give the overall strategy behind the rules. How do the pieces of each individual rule come together to generate the language?

For #1b and #2b, also compare the strategies of the two grammars. This goes deeper than "in the first grammar, the second rule generates all the symbols, while it is the first rule in the other grammar" — the order the rules are listed in is arbitrary. For #1, both grammars generate the symbols, get them in the right order, and then sweep through to convert non-terminals to terminals. What's the difference in the steps between the two? What symbols (terminals and non-terminals) are generated by the generate symbols step in each grammar? And what's the difference in the sweep steps?

When creating a grammar, it is helpful to think of grammars for similar languages. For $a^nba^nba^n$ (#3), think of three groups of symbols with the same number in each — this is similar to $a^nb^nc^n$, the language from #1. For $a^nb^mbc^{nm}$ (#4), think of multiplication and remember that $n^2$ is $n \times n$ — use $a^{n^2}$, which was discussed in class (see the slides), as a pattern.

Keep in mind that your grammar needs to generate the right strings, but also only the right strings.