Recall the class scheduling problem described in lecture on Tuesday. We are given two arrays $S[1..n]$ and $F[1..n]$, where $S[i] < F[i]$ for each i, representing the start and finish times of n classes. Your goal is to find the largest number of classes you can take without ever taking two classes simultaneously. We showed in class that the following greedy algorithm constructs an optimal schedule:

> Choose the course that *ends first*, discard all conflicting classes, and recurse.

But this is not the only greedy strategy we could have tried. For each of the following alternative greedy algorithms, either prove that the algorithm always constructs an optimal schedule, or describe a small input example for which the algorithm does not produce an optimal schedule. Assume that all algorithms break ties arbitrarily (that is, in a manner that is completely out of your control). ***Exactly three of these greedy strategies actually work.***

1. Choose the course x that *ends last*, discard classes that conflict with x, and recurse.
2. Choose the course x that *starts first*, discard all classes that conflict with x, and recurse.
3. Choose the course x that *starts last*, discard all classes that conflict with x, and recurse.
4. Choose the course x with *shortest duration*, discard all classes that conflict with x, and recurse.
5. Choose a course x that *conflicts with the fewest other courses*, discard all classes that conflict with x, and recurse.
6. If no classes conflict, choose them all. Otherwise, discard the course with *longest duration* and recurse.
7. If no classes conflict, choose them all. Otherwise, discard a course that *conflicts with the most other courses* and recurse.
8. Let x be the class with the *earliest start time*, and let y be the class with the *second earliest start time*.
   - If x and y are disjoint, choose x and recurse on everything but x.
   - If x completely contains y, discard x and recurse.
   - Otherwise, discard y and recurse.
9. If any course x completely contains another course, discard x and recurse. Otherwise, choose the course y that *ends last*, discard all classes that conflict with y, and recurse.