General hints for our series for mentors

The focus of this series is preparation for technical programming interviews where interviewees are expected to solve programming problems by writing code on a whiteboard. The role of the mentor is to be a (friendly, helpful) interviewer -- choose a problem and interviewee (from the group), provide guidance on the solution as needed to keep on track, and provide feedback for improvement. The goal is to improve confidence by practicing in a safe and positive environment.

We split participants in smaller groups under categories - intermediate & advanced. Each group has a mentor who guides the group in solving a problem. Groups switch mentors after 30 minutes. There will be 2-4 solved problems for each group over the course of the evening.

We try to focus more on practical approach and problem discussion is not enough -- ideally we end with a correct solution that handles all corner cases. Rule of thumb is to implement fully at least a brute force (naive) solution using a whiteboard. If they didn't implement it, they are not allowed to switch a problem and new mentor should continue to help with that solution (it's up to mentor to either finish with a naive implementation or start a new problem).

- As a good warm-up exercise you may practice an introduction with the first group:
  - Introduce yourself. Example: "I'm a software engineer working .. Using .. technologies. Focusing on... And I'm here as a mentor to help you"
  - Ask everyone "Please introduce yourself to the group".
  - Everyone talks about themselves. Under 1 min per talk. Please interrupt carefully when it's becoming too long.
  - You give feedback. Example:
    - "It is common to briefly introduce yourself at the beginning of almost any interview. You need to rehearse this ahead of time. They call it ice-breaking talk."
    - "That was too long. Don't talk about too informal things"
    - "That was good. However I would skip those complaining parts"

- Please give feedback after each performance using the whiteboard. What was good and on what should the participant work to improve? (Usually you'll get a 5-minute warning before the change of mentors. That is a good time to wrap up the solution quickly, and give feedback.) Areas of improvement might include:

    - asking questions before implementation
    - thinking about edge cases
    - how to use whiteboard space to write code
    - be able to debug written code
    - be able to implement brute force(naive) solution
    - be able to identify big-O (eg. order n log n) behavior of the solution

- Check that all participants are involved in the discussion and there are different participants at the whiteboard (some are very shy and we are trying to help them at least come to a whiteboard and write something very simple). After switching groups, mentors should ask someone who has not already presented a solution to do the new problem.

- Sometimes the interviewee will be able to progress smoothly through the solution. Often, however, they will reach one or more sticking points. Some potential problems/courses of action are:
    - Lack of confidence in the solution before starting to whiteboard: Ask them to explain the solution before writing on the whiteboard and we brainstorm a bit. If it's an easy/short problem, pick the person who hasn't spoken or spoke very little in the brainstorm session to write on the whiteboard. If it's a difficult problem and lengthy one, ask the person who has spoken the most to write it.
    - Lack of confidence in the solution when already up near the whiteboard enthusiastic to write code, although not with an optimal solution: Ask them to complete it. Otherwise someone else has an opportunity to write a better solution.
    - Struggling with the programming language constructs. Many of them get stuck on a syntax issue: At that point, tell them not to worry about the language but write pseudocode to describe the steps. During feedback, mention about focusing on writing code without an IDE.
    - Trying to optimize code while writing the solution: Tell them to complete the code and then rewrite with the optimal solution. Once they complete the initial solution, instead ask the person who pointed out the optimal solution or agreed with the optimal solution in the group to write it. This way everyone gets a chance to write code on the whiteboard.

- One of the most common areas for improvement is to continue talking while writing on the whiteboard. Prompt the interviewee to talk through the code. Ask them to explain each line to understand how they're thinking about the solution. Speaking while writing helps the mentor understand why they're stuck so the mentor can ask questions or offer suggestions that will help them arrive at the next step faster. In an actual interview, it also keeps the interviewer engaged and gives insight into the interviewee's thought process which is what they (should be) really looking for from the programming exercise. One mentor phrased it as "practice writing code while explaining it to someone or if no one is around, to a rubber duck <https://en.wikipedia.org/wiki/Rubber_duck_debugging> :-) "