Nick & Kelly

1. Pick a goal
   1. Try to be as explicit and clear as possible for what you are asking the participants to do
      1. I have tried to be (what felt to me) painfully clear but I still was falling short making it clear to some people.
      2. People will always tend to notice or ask about things that are not the focus of the exercise, being specific about the goal may help. Also setting up a simplified scenario can be helpful (although requires much more prep time)
2. Prepare the activity.
   1. Create a minimal working example
      1. Working directly with your own raw code can be challenging
         1. It is likely filled with code specific to you/your project
         2. It can take a lot of time to explain details and working of code
         3. Code can have multiple steps that are needed to run to get to script focus
         4. Using a whole script opens up the risk of getting off topic
         5. If needed, Prepare a small example dataset
3. Keep It Simple Silly (KISS)
   1. There is never enough time
      1. I typically have multiple activities planned, but rarely get through them all
      2. Modularize the activity so that each can be its own and if accomplished you are able to move onto the next
   2. One activity per code review is typically sufficient. small, digestible in an hour (<100 lines)
      1. This will take up most of the time
         1. Intro - 10 min
         2. Think/pair - 30 min
         3. Share in group - 10 min
         4. Debrief - 5 min
4. Consider the audience
   1. Practice exercise
      1. When in doubt try it out!
         1. Run your plan by someone prior to lab meeting to make sure your request/goal is understandable
   2. Participants value (based on feedback)
      1. Group work
         1. Helpful to write code, see others approach or work through problem
         2. Smaller groups 2-3 to make sure everyone is able to talk
      2. Short exercises (not rushed)
      3. Help/guidance
         1. Clear explanation of the problem, code, and goal
         2. Hints or pointing towards resources for those new to specific task
      4. Tasks
         1. Short (not rushed)
         2. Simplified (so focus can be on goal and not figuring things out)
         3. If complex task, split into digestible parts
5. Share the code
   1. Preferably several days in advance
   2. Determine dependencies
   3. Communicate goals
6. Use think/pair/share
   1. Pair to mix up experience
   2. Gives individuals time to think on their own, discuss their ideas and learn from each other and then present unique idea to all, instead of getting one answer to a question
   3. This allows pairing of different experiences and greater exchange of ideas
7. Reciprocate respect
   1. Facilitation
      1. Lead the discussion
      2. Have an action plan: Focus the conversation
      3. Balance expertise in groups
      4. Attempt to purposefully engage early learners
      5. Avoid letting (senior) individuals dominate
   2. Attitude:
      1. Both:
         1. Be humble
         2. Be willing to incorporate feedback
         3. Ask clarifying questions and note *why* you need the clarification
      2. Participants:
         1. Remember that this isn’t a competition
         2. Focus on the speaker’s goals
         3. Avoid distractions and engage with the presenter
         4. Allow others to contribute
         5. Give feedback & coding suggestions at a novice explanation level
         6. Give compliments
8. Feedback / debrief