Advanced Git Local Worksheet

You have now had a fair amount of practice using `git` by yourself; likely without too many issues. As multiple people start to work on a single repository, however, things start to get complicated. This worksheet is intended to give you a better mental model of what is happening when you do things like `git merge`, `git checkout`, and `git rebase`.

Problem 0: Introduction Sequence

Go to https://learngitbranching.js.org/ and complete **all four** "levels" of the `Introduction Sequence` track.

1. Take a screenshot of checkmarks for **all four** "levels" and paste it in the space below:

   ```

   ```

2. The game does `git commit` rather than the more realistic `git commit -m "message"`. Why do you think they made that decision?

   >

3. `git commit` adds a "commit node" to your repository graph. Explain why it is important to know what branch you're on when you do `git commit`.

   >

4. What is the difference between `git rebase` and `git merge`?

   >

Problem 1: Ramping Up

Go to https://learngitbranching.js.org/ and complete the **first two** “levels” in the `Ramping Up` track.

1. Take a screenshot of checkmarks for **first two** “levels” and paste it in the space below:

   ```

   ```

2. What does `HEAD` mean?

   >

3. What is the difference between `HEAD` pointing to a commit and `HEAD` pointing to a branch?

   >

4. What is the difference between `HEAD` pointing to a commit and `HEAD` pointing to a branch if you do `git commit`?

   >

Problem 2: Moving Work Around

Go to https://learngitbranching.js.org/ and complete **just the first** “level” in the `Moving Work Around` track.

1. Take a screenshot of checkmarks for **just the first** “level” and paste it in the space below:

   ```

   ```

2. How are `git cherry-pick` and `git rebase` similar?

   >

3. `git` is a VERY complex system that takes a long time to learn. What has been the hardest part about `git` for you to understand?

   >

Problem 3: Reflection

1. How did using the visualizations in `learngitbranching.js.org` change or clarify your mental model of how `git` works? Give a specific example of something that became clearer.

   >

2. Thinking ahead to working on your group project with multiple people committing to the same repository, what potential `git` challenges (e.g., merge conflicts, confusion about branches, accidental rebases) are you most concerned your group might face, and why?

   >

3. Did you notice any obvious errors with this worksheet that should be fixed for future students? If so, what were they?

   >

4. Were there things in the worksheet that were overly confusing and that you think could be improved? If so, what were they?

   >