Assignment 11 - Git Branching

TW Lab 2025

In today's lab, you will explore some more Git commands used to work with branches.

Instructions

Complete each part of the assignment step by step, following the provided commands and tasks. After executing each command, observe the output and take a screenshot. At the end of the assignment, submit a document file (.docx or .pdf) that contains the terminal screenshots along with a brief description of the observed outputs.

You will require a git repository which has been initialized with a few commits. If you do not have access to your previous work, create a new repository and commit your vector addition code (vecadd.py) along with a readme file. Now make some changes to the readme and create another commit.

1. Observe the "HEAD" pointer. Check where the "HEAD" pointer is currently point ing by running:
   git log --oneline
2. Suppose you wish to create a faster version of your vector dot-product algorithm. Since your old algorithm is stable, you decide not to disturb that and thus, store this fast version separately in a new branch. Use the following command to create a branch named "fast": git branch fast
   Verify the branch creation by listing all branches using: git branch
3. You then decide to use a parallel algorithm to implement a fast dot-product. So rename the "fast" branch to "parallel" using an appropriate command.
4. Switch to the "parallel" branch. Also, verify that you have switched branches using: git status
5. Now, modify your dot-product code in the repository to include parallelism and commit your changes. Observe the output of git log. What difference do you see between "main/master" and parallel?
6. Meanwhile, your teammate, who was using your stable dot-product code (from "main/master" branch), reported a bug. Switch to "main/master" and modify one line in your old code to create a commit.

   Capture the output of git log --oneline --all --graph and git status, show ing that the working directory is clean and the fix is committed in the "main/master" branch.

7. You are extremely happy with the performance of the parallel version of your dot

product code. So you decide to merge those changes into “main/master” so that your teammate can make use of it.

To merge the “parallel” branch into the “main/master” branch use: git checkout main/master
git merge parallel
If there are no conflicts, the merge should be successful. However, if there are conflicting changes, Git will notify you of a merge conflict.

8. In the 5th and 6th exercise, you created two commits, one on each branch. If you modified the same lines of code in both these commits, git would not know how to merge them and hence raise a CONFLICT. In such cases, git copies both the sections with some demarcations. (<<<<<, =====, >>>>>) and leaves the file in a “both modified” state and expects that you will manually restructure the code. Once complete, you can add the file and commit to complete the merge process.

In case you are facing a merge conflict after exercise 7, resolve it using the method described above. In case you did not see a merge conflict, create two such commits (one on each branch) where you modify the same line of code, and then try to merge the “parallel” branch in “main/master” again. Once you see a conflict, check git status and open your file in an editor. Restructure the code as you wish and finally remove the demarcations. Finally, commit the file.

9. Since we have successfully merged the code into “main/master”, we no longer need the “parallel” branch. Delete the “parallel” branch.

10. Sometimes we need to checkout another branch or a previous commit while our current working directory may not be clean (i.e. it may contain some modified files). In such cases, git provides us a utility to temporarily push our changes on a stack and later recover them.

Explore the git stash command. Use the help command to read a description and sample usage: git help stash. You can use the help command with all other git commands as well.

Submission

Upload the following to Google Classroom:

1. Generated PDF file

2. The source file

Please ensure that the document is well-organized, with clear headings for each task. 2