Lab-5: GIT

Members:
Prabhjot Kaur - B00843735

Part-1:

1. **How many constants were added between versions 4 and 5?**
    a. Four constants were added between versions 4 and 5:
        i. private static final int BOLD = 3;
        ii. private static final int UNDERLINE = 4;
        iii. private static final int ITALICIZE = 5;
        iv. private static final int LISTITEM = 6;
2. **In which version was the method "main" separated into its own file?**
    a. In version 4:
        i. Commit id: commit 46a3927d7e77bd5649afa0f4ef5f0e27d11efc27
        ii. Commit message: "Split main() from the rest of the code. Include code to split the body into paragraphs / lists".
3. **How many versions of the code have been checked in?**
    a. Five versions of the code have been checked in already as five commits were made already.
        i. commit b0590e1936d3fb47272ea74623e7f9b59866ae60
        ii. commit 81e41a2fb4e99eac7ab2807f9661f7ba54bdd076
        iii. commit 18c11a776282dbcd2e8eac74cbdcc8a954be7232
        iv. commit 46a3927d7e77bd5649afa0f4ef5f0e27d11efc27
        v. commit cc797b54956173cdc28eb743331c787f22a2b253
4. **How can you tell? In which version did the external documentation file appear in the repository?**
    a. External documentation file 'External_documentation.txt' was added commit - 18c11a776282dbcd2e8eac74cbdcc8a954be7232 in version-3
    b. Got to know about this by running '*git ls-tree --name-only –r <commit id>*' for each commit which lists the files added or modified for that commit

Part-3:

5. **Which imports were added between versions 9 and 10?**
    a. Two imports were added:
        i. import java.util.HashMap;
        ii. import java.util.HashSet;

Broadening Questions:

1. **When working alone on a project, how frequently should you commit your code to a version control system? Explain why**
    - When working alone it's important to do version control at that time as well.
    - But frequency of code commits does not need to be that high.
    - You should commit changes that include some major code change especially logical change in case you plan of switching to the old logic again.
    - You should commit changes where you are adding some new functionality.
    - You can avoid checking in every minor change like changing some syntax, code cleaning, indentation, new imports.
    - All these changes cab be clubbed into one change and can be committed at the end.
2. **When working in a team, how frequently should you commit your code to a version control system? Explain why**
    - When working in a team frequency of code commits to version control system should be high.
    - Even the small changes can affect the code and can sometimes become the blocker when you try to merge everyone's changes at the end.
    - So, it's important that every change made by each person should be committed into version control.
    - Having said that, that doesn't mean for every one-liner every team member should raise a commit.
    - There should be a defined change with some meaningful impact and message so that git/version control system won't be dumped with changes and it will be easy to backtrack the changes.
3. **Why might you create branches for your project in your version control system?**
    - There can be different reasons why we need branches in the project.
    - First one being, we always need master branch to be clean and bug free, so only that code should be available in master branch that is tested and verified and in good working condition.
    - If you are experimenting with some new logic and want to keep track of those changes then branch can come handy for that.
    - Apart from this, branches can also be used to keep production level code to be separate from development code.
    - Branches can also be used to maintain a release cycle, where every branch contains new set of features and improvements for that release.

References:

1. Lab-5 slides
2. https://stackoverflow.com/