EXERCISE #1: HOW GIT SAVES FILES (15 – 20 MINUTES)

COMMANDS CHEAT SHEET:
https://github.com/emilyxxie/gits_guts_commands

1. create a new folder and initialize git on it
2. use a text editor to create & save a new file. write a 1-line bio in it
3. save the file to your git database using plumbing (see cheat sheet)
4. use resulting hash to inspect the file using a plumbing command (cheat sheet)
5. edit the file: underneath your bio, add 1 additional line about how you love coding & then save the file to your database
6. list the contents of your objects folder - you should see two items. you now have two versions of your file.
7. inspect the resulting object. HOWEVER, beforehand, turn to your neighbor, and discuss what you think you will see.

EXERCISE #2: MAKING TREE OBJECTS (10 – 15 MINUTES)

COMMANDS CHEAT SHEET: https://github.com/emilyxxie/gits_guts_commands

1. add the file you created to the index using plumbing
2. "ls" your .git folder – you should now see an index appear
3. examine the contents of your index file, aka staging
4. run a porcelain git status
5. write your tree
6. list all objects in your database

IF YOU FINISH EARLY – CHALLENGE: TREES IN TREES (5 MINUTES)

1. try making a subdirectory in your repo. inside, create a new file with content
2. add the file to your index (aka staging area)
3. write your tree & inspect the resulting tree object. you'll see a reference to another tree

EXERCISE #3: COMMIT OBJECTS (15 – 20 MINUTES)

COMMANDS CHEAT SHEET: https://github.com/emilyxxie/gits_guts_commands

1. create a commit object using the recently created tree
2. take a look at your git database
3. inspect the commit object
4. change your existing files or add a new one
5. update your index and write your tree
6. create another commit object, chaining it to the one previous
7. inspect this new commit object
8. run git log --stat [commit hash]

CHALLENGE DISCUSSION – IF YOU FINISH EARLY: (5 MINS)

1. Have you ever done a "git commit --amend" to edit your commit message?
2. If so, discuss with a neighbor or someone else who's finished: what do you think will happen when you amend a commit? Will a new commit object be created, or will we have simply edited the pre-existing commit object for the commit in question. Why?

EXERCISE #4: GIT REFERENCES (20 – 25 MINUTES)

COMMANDS CHEAT SHEET:
https://github.com/emilyxxie/gits_guts_commands

1. list everything in your git references folder. you'll see nothing because you don't have any branches.
2. use a plumbing command to manually create a branch
3. list your refs folder again to see the result
4. cat your master branch (remember, you can cat because branches are just text files)
5. run a porcelain command to create & checkout to a new branch (git checkout -b)
6. list your .git/refs/heads folder again
7. now cat the new branch to inspect the commit hash
8. now edit some files in your new branch. add and commit the changes using porcelain like you would normally. now, cat the branch again.

CHALLENGE – IF YOU FINISH EARLY: (5 MINS)

1. Remember how git keeps track of what branch you're on via .git/HEAD? cat the file to look at it. what if you change this file to point to another branch? Discuss w/ neighbor, then implement / explore. run a git status and a git log. have fun, break stuff, mess around
2. What would happen if you just go in and change a branch file in .git/refs/heads so that the commit is different? Discuss with a neighbor, and then implement / explore / break stuff / have fun messing around.