Ben Jacobs
Home-baked Version Control
Pros: Easy to create and manage, all files are stored on your system, all files can be accessed locally
Cons: You jeopardize losing all progress if you don't backup your workstation, takes up memory storage, you work independently of other people on the project

Autocratic Version Control
Pros: Servers are able to take and store backup copies of files, easier to collaborate on project
Cons: Conflicts can arise when merging

Centralized Version Control
Pros: Projects are mainstreamed to help productivity
Cons: Commits are slower, if the main sever crashes then versions will be lost

Distributed Version Control
Pros: Flexibility for remote work, edit local files, fix files remotely then merge, branches
Cons: Slower development, potential security issues

Section/Page 2:
Arrows:

1: Edit the file (vim *filename*)
2: Stage the file (git add *filename)*
3: Commit (git commit)
4: Remove the file (git rm *filename)*
5: Commit (git commit)
6: Unmodified and in master repository
7: Copies file to remote repository
8: git pull (branch)
9: git add .
10: git push origin your-branch
11: git commit
12: Check the current status, break down issues, commit the master independent file then the second merge. If worse comes to worse, you can abort the operation using git merge -–abort.