GE01 Python, Pair Programming and Version Control

Effort: Collaborative Assignment [CS3300 Academic Integrity](#) (Pairs)

REQUIREMENT: At least 20 minutes of pair programming with someone else.

Points: 40 (see rubric in canvas)

Deliverables: DO NOT UPLOAD A ZIP FILE and submit word or pdf files.

- **Upload this document with your answers**
- **A screencast video of your pair programming activity**
- **Resume and interview questions**

Due Date: See Canvas

Goals:

- Communicate effectively in a variety of professional contexts within a team, with customers, creating oral or written presentations, and technical documents.
- Devotion to lifelong learning: Prepare to learn on their own whatever is required to stay current in their chosen profession, for example, learning new programming languages, algorithms, developmental methodologies, etc.
- Utilize pair programming to begin learning python.

Names of the person you collaborated

Tyler Andrews, Zach Snyder

Description: Learning how to learn new technologies. This is not about getting everything working perfectly the first time but collaborating, communicating, finding resources and problem solving with others. Most of all do not panic if you run into issues. Note the issues and how you resolved them.

Think about what information is helpful to have for the next time you do this.

Find 4 or more resources that could be valuable for a new person getting started with python and version control.

Brief description	**Resource**
Describes how to use the command line to install git on your computer	https://www.howtogeek.com/832083/how-to-install-git-on-windows/
Help me set up the repository	https://docs.github.com/en/get-started/getting-started-with-git/about-remote-repositories
Taught me how to use curl, ultimately I believe this failed and a different attempt overwrote it but	https://stackoverflow.com/questions/2423777/is-it-possible-to-create-a-remote-repo-on-github-from-the-cli-without-opening-br

I still want to include it.	
Ultimately I returned to github docs a lot, they seem to have almost all questions answered here.	https://docs.github.com/en

Start exploring git, github, command line, and python in a virtual environment.

1 Python and IDE

Set up your python and IDE for your python development.

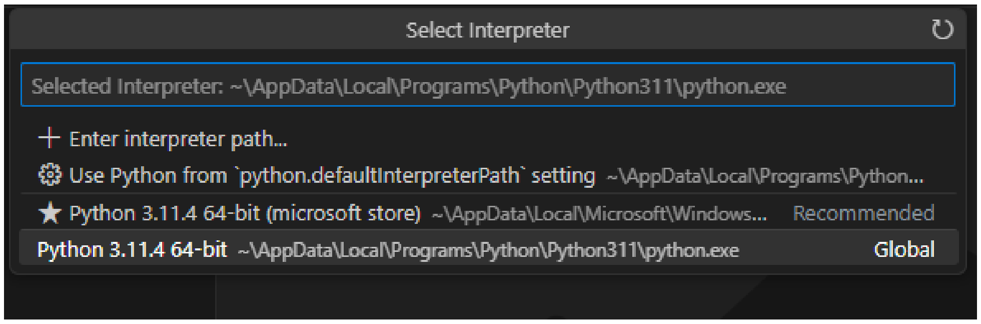
Install Python

1. Open the command window and check your python version to see if you have it installed.
2. Install python version 3.11 Download Python. If on windows and have older version of python you should uninstall first : How to Uninstall Python

Install VS Code IDE

You can use a different IDE but this is what I will be using in my lectures. This has nice tools to integrate with python, django and databases.
https://code.visualstudio.com/download

1. Configure the Python interpreter: In Visual Studio Code, open the Command Palette by pressing `Ctrl+Shift+P` (Windows/Linux) or `Cmd+Shift+P` (Mac). Search for "Python: Select Interpreter" and choose the Python interpreter associated with your virtual environment (e.g., `myenv`).



Select Interpreter

Selected Interpreter: ~\AppData\Local\Programs\Python\Python311\python.exe

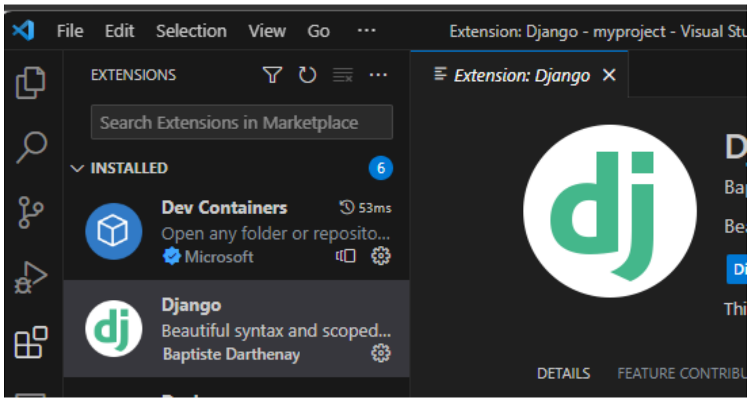
+ Enter interpreter path...

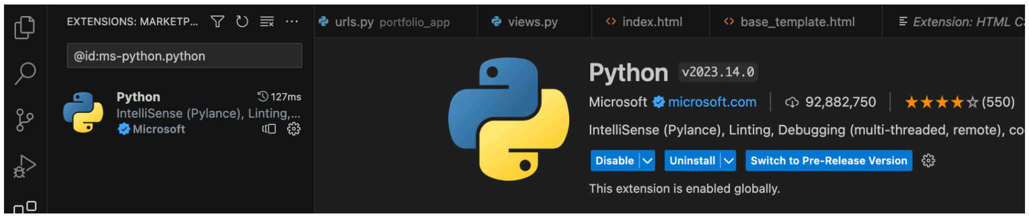
Use Python from `python.defaultInterpreterPath` setting ~\AppData\Local\Programs\Python...

★ Python 3.11.4 64-bit (microsoft store) ~\AppData\Local\Microsoft\Windows... Recommended

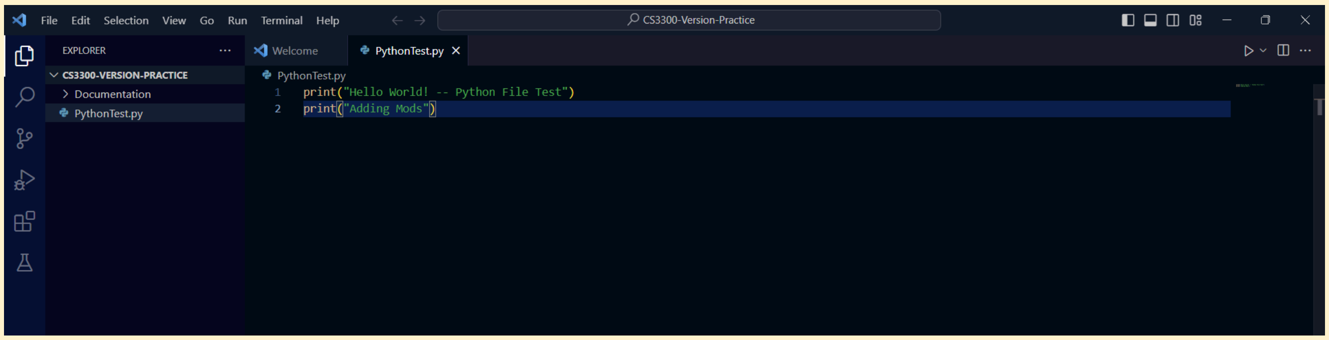
Python 3.11.4 64-bit ~\AppData\Local\Programs\Python\Python311\python.exe Global

2. Install the Django extension developed by Baptiste Darthenay: In Visual Studio Code, go to the Extensions view and search for the "Django" extension. Install it to benefit from Django-specific features and enhancements for what we will be doing later.


File Edit Selection View Go ···
Extension: Django - myproject - Visual Stu
EXTENSIONS
Search Extensions in Marketplace
INSTALLED 6
Dev Containers 53ms
Open any folder or reposito...
Microsoft
Django
Beautiful syntax and scoped...
Baptiste Darthenay
Extension: Django
DETAILS
FEATURE CONTRIBU



3. You can use this to edit your python file for practice.
4. Take a screenshot of the ide you have set up and the python file from the repository once you edit it below.



```
print("Hello World! -- Python File Test")
print("Adding Mods")
```

2 Pair Programming

Goal: Improve software quality by having multiple people develop the same code.
Setup:

- One shared computer, alternate roles
- Driver: Enters code while vocalizing work
- Observer: Reviews each line as it's typed, acts as safety net + suggest next steps

Effects:

- Cooperative, a lot of talking! + Increases likelihood that task is completed correctly
- Also transfers knowledge between pairs

Start learning the basics by going through Hello, World! - Free Interactive Python Tutorial by following the instructions below.

- You should spend at least 20 minutes pair programming
- Choose video screen-recording software that you can use to capture your discussion and screen. (such as https://obsproject.com/)

Where it says exercise code: that means for that section you are doing the exercise at the end of the information.

- Do not copy the solution code. Instead copy your code and paste below. Add any notes that would be helpful.
- Do not worry if you do not finish all the parts when pair programming but you should start pair programming and videoing with lists.
- Complete on your own after the pair programming ends.

Scan the following sections before pair programming. Take turns summarizing each section to the other. Add any brief notes or examples.

Hello, World!

print(“Hello World”)

Python uses print for basic output, indentation is very important in python and is used to separate blocks of code

Variables and Types

In python you dont have to specify what type of variable it is, python will figure it out for you from the inputted value

mystring = "hello"

myfloat = 10.0

myint = 20

Lists Review and complete exercise code:

```
numbers.append(1)
numbers.append(2)
numbers.append(3)

strings.append("hello")
strings.append("world")

second_name = names[1]
```

Basic Operators Review and complete exercise code:

```
x_list = [x, x, x, x, x, x, x, x, x, x]
y_list = [y, y, y, y, y, y, y, y, y, y]
big_list = x_list + y_list
```

Scan the following sections. Take turns summarizing each section to the other. Add any brief notes or examples.

Basic Operators

Python has some unique operators past normal operators like ** which is power and operators that can be used on strings and lists

String Formatting

```
data = ("John", "Doe", 53.44)

format_string = "Hello %s %s. Your current balance is $%s."
```

Formatting strings in python is very similar to C or C++ but much much more simple

Basic String Operations

Python has a very large database of operations that can be performed on strings which might be helpful for inputted data!

This specific tutorial has additional commands at the very bottom that my team missed, and it made the exercise much more difficult than anticipated

Conditions

```
number = 16

second_number = 0

first_array = [1,0,0]

second_array = [1,2]
```

Python conditions are very similar to other languages besides the fact that you dont need () on conditions and most conditions can be typed instead of using a symbol

Loops

```
for number in numbers:

    if number == 237:

        break

    if number %2 == 0:

        print(number)
```

Loops in python are also very similar to other languages but are much easier to implement.

Functions Review and complete exercise code:

```
def list_benefits():
    return ["More organized code", "More readable code", "Easier code reuse", "Allowing programmers to share and connect code together"]

# Modify this function to concatenate to each benefit - " is a benefit of functions!"
def build_sentence(benefit):
    return ("%s is a benefit of functions" % benefit)
```

Classes and Objects Review and complete exercise code:

```
car1 = Vehicle()
car2 = Vehicle()
car1.name = "Fer"
car1.kind = "convertible"
car1.color = "red"
car1.value = 60000.00
car2.name = "Jump"
car1.kind = "van"
car1.color = "blue"
car1.value = 10000.00
```

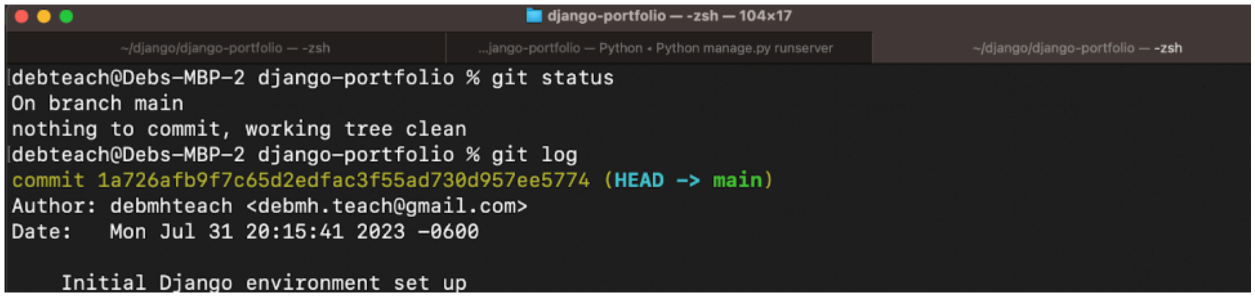
Dictionaries Review and complete exercise code:

```
phonebook["Jake"] = 938273443
del phonebook["Jill"]
```

3 Version Control

Set-up git and github repository

Use the command line tool of your preference in your environment. I ended up using command prompt on my windows but also have used windows powershell.I use the generic command tool on my mac. Here is an example of using the default command prompt



django-portfolio — -zsh — 104×17
~/django/django-portfolio — -zsh
...jango-portfolio — Python ◂ Python manage.py runserver
~/django/django-portfolio — -zsh

```
debteach@Debs-MBP-2 django-portfolio % git status
On branch main
nothing to commit, working tree clean
debteach@Debs-MBP-2 django-portfolio % git log
commit 1a726afb9f7c65d2edfac3f55ad730d957ee5774 (HEAD -> main)
Author: debmhteach <debmh.teach@gmail.com>
Date:   Mon Jul 31 20:15:41 2023 -0600

    Initial Django environment set up
```

Research

- What is git and github? What does git provide? What does github provide?
  - Git is a local version control software whereas github is an online tool used to host repositories
- How can you create a github repository from a local folder?
  - By going to the local folder and running the command git init in a command window
- What documentation could be useful to help understand the commands?
  - A large database of commonly used commands and their uses

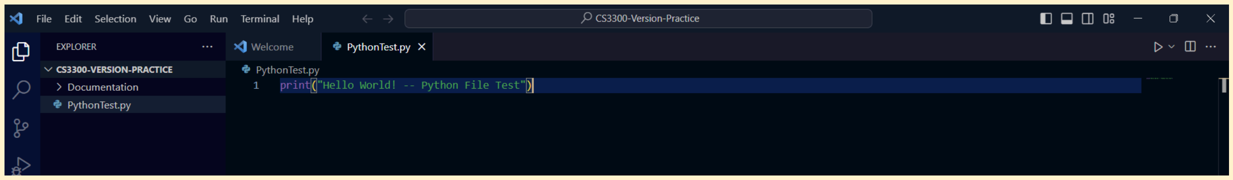
Include resources in the table above.

1. Create a python file in a local folder cs3300-version-practice
2. Create a folder called documentation in cs3300-version-practice that contains this document.
3. Create a github account if you do not have one.
4. Create a github repository that is public from the local folder.

Explain what you did and the commands you used.

To get git started I first had to download git, which I did through the command line using a command I found with my first resource: winget install --id Git.Git -e --source winget
Initially this wasn't enough and for whatever reason the command "git" wouldn't work. Since then this issue resolved itself (might have needed a restart)
Once Working I simply typed "git" and that told me to use the command "git init" to initialize a repository in my workspace.
Next I set up a repository through git hub to save my files using curl:
curl.exe -u 'Username' https://api.github.com/user/repos -d '{"name":"nameofrepo"}'
Once I ran this I realized I needed to authorize git to run through curl so I had to install git hub CLI with winget:
`winget install --id GitHub.cli`
This also required a restart. After restarting I decided to try a new command as I ran into thousands of errors with the curl.
gh repo create --private my-project
This created a repository for me, now I could link the two with:
git remote add origin git@github.com:USER/REPO.git
Finally I ran:
git push origin master
This seemed to fail, but ultimately the two connected so I called it good!

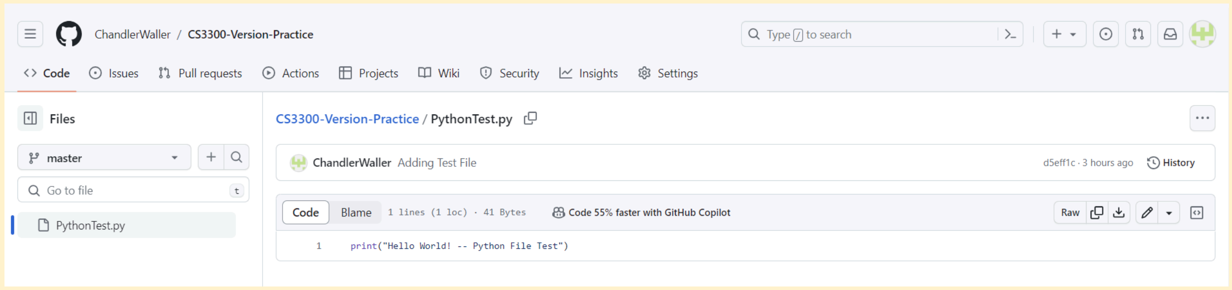
Paste a screenshot of your local directory code



```
print("Hello World! -- Python File Test")
```

Local code

Paste a screenshot of your github repository code


ChandlerWaller / CS3300-Version-Practice
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Files
master
Go to file
PythonTest.py
CS3300-Version-Practice / PythonTest.py
ChandlerWaller Adding Test File
d5eff1c · 3 hours ago
History
Code
Blame
1 lines (1 loc) · 41 Bytes
Code 55% faster with GitHub Copilot
Raw

```
print("Hello World! -- Python File Test")
```

Code on online repository

Paste the url to you github repository code

https://github.com/ChandlerWaller/CS3300-Version-Practice.git

5. You may need to generate an SSH Key pair to configure remote access to your repositories. Github's instructions for this process can be found here.
6. You may need to set

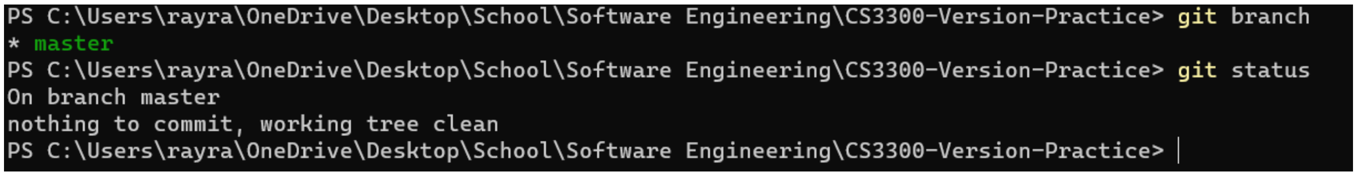
   git config --global user.email "you@email"   (email associated with repository)

   git config --global user.name "Your Name

Add, Commit, Push Practice

1. You can just work with updating a python file.
   1. Check the git branch and status

```
git branch
git status
```



```
PS C:\Users\rayra\OneDrive\Desktop\School\Software Engineering\CS3300-Version-Practice> git branch
* master
PS C:\Users\rayra\OneDrive\Desktop\School\Software Engineering\CS3300-Version-Practice> git status
On branch master
nothing to commit, working tree clean
PS C:\Users\rayra\OneDrive\Desktop\School\Software Engineering\CS3300-Version-Practice> 
```

2. Update the file. Before you can commit the version you must add the new file to the index (the staging area)

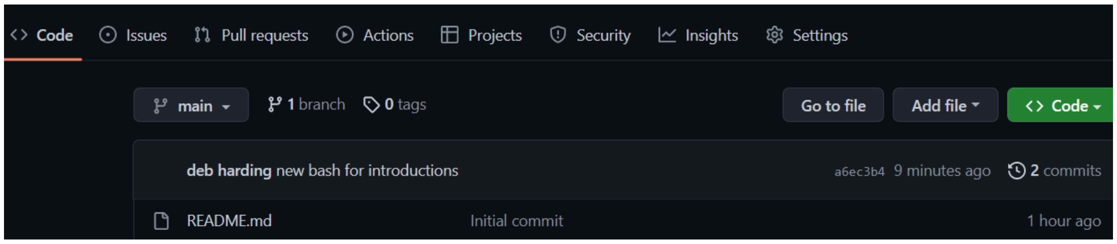
git add .
git status
Finished this

3. Record changes to the local repository with a description but first you might need to include the author identity. Then check the status

git commit -m ‘add description’
git status
Finished this

4. You will add your code, commit and push. Then explore the repository on the remote server, github
   git push
   git status

Code already in repository



deb harding new bash for introductions — a6ec3b4 9 minutes ago — 2 commits

README.md — Initial commit — 1 hour ago

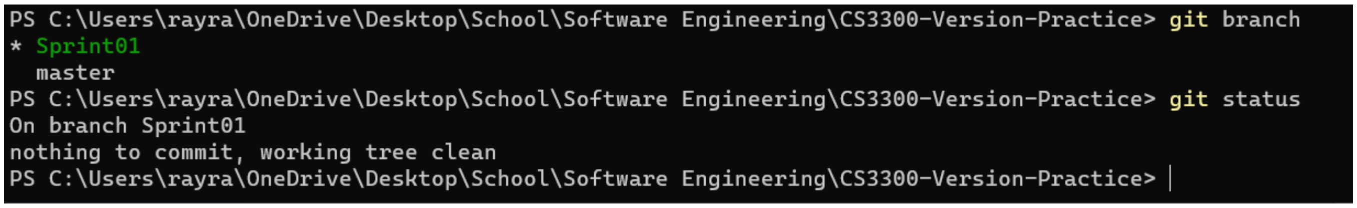
Branching

1. From the command line in your repository on your computer check the log and what branch you are on.
2. Create a branch called sprint01 and check the log and branch
   Copy and paste the commands you used

Git checkout -b Sprint01

3. Switch to sprint01 branch to check out code:

git checkout 'sprint01'
git branch
git status



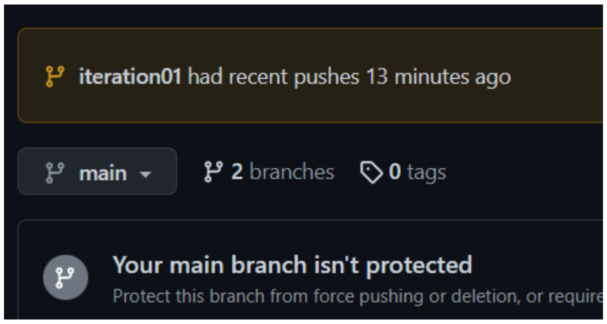
```
PS C:\Users\rayra\OneDrive\Desktop\School\Software Engineering\CS3300-Version-Practice> git branch
* Sprint01
  master
PS C:\Users\rayra\OneDrive\Desktop\School\Software Engineering\CS3300-Version-Practice> git status
On branch Sprint01
nothing to commit, working tree clean
PS C:\Users\rayra\OneDrive\Desktop\School\Software Engineering\CS3300-Version-Practice> |
```

4. Modify python file and Add the file to the staging area and update the version in your local directory.
   Copy and paste the command(s) you used

```
Git status
Git add .
Git commit -m "Changes made"
```

5. Share the changes with the remote repository on the new sprint01 branch. Go to your github and you will see you now have two branches. Click to view the branches. Now others working on the branch could pull your updates from the sprinto1 branch.

   git push --set-upstream origin sprint01
   git status
   git log



iteration01 had recent pushes 13 minutes ago

Your main branch isn't protected
Protect this branch from force pushing or deletion, or require

6. Switch to the main branch and update the remote main branch repository with the change from sprint01 branch. Then go to github to see the versioning.

   Copy and paste the commands you used

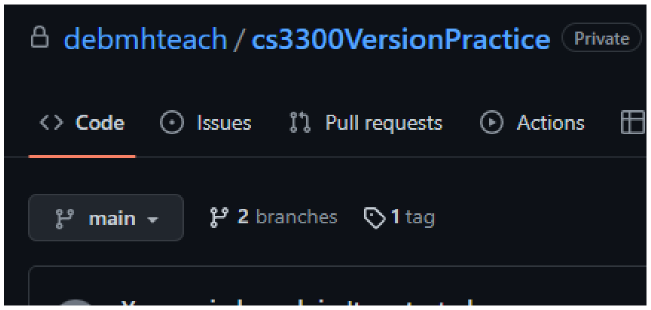
```
Git checkout main
Git merge Sprint01
```

7. Tag the main branch 'v1.0' then view the tag and push to the remote repository. When you go to the remote repository you should see the tag listed.

Copy and paste the commands you used

```
Git tag -a v1.0 -m “First Version Tag”
```

For example



Version Control Concepts

Individually answer each question in your own words, **including any resources you used to help you above.** This will be helpful when you keep technical documentation with your team. **You can use AI to help you understand but answer in your own words.**

3.1 Explain software version control. Address in your description branches, commits, merges, tags.

Software version control is making sure that you can edit, fix, and generally update software with a working “Version” always up. You do so by using a branch to edit the code while the main branch still has working code, then you can commit your changes which save them. Once saved it may be time to test your changes by merging your branch and the testing branch. Finally after everything is done you would move your changes to main and tag it so that you can always return to that version of software using the tag. Thus effectively controlling the version.

3.2 Research what Git is and what its relationship is to software version control. Include how GitHub integrates with git.

Git specifically allows users to save changes to their code on their computer. It sets up a save point for code so to speak. Github integrates with git and literally creates a “hub”, or online storage for the git.

3.2 Explain the following commands and include examples: commit, pull, push, add, clone, status, log, checkout

We start with add
Git add adds any changes to the git
Git commit on the other hand fully commits these changes and makes them the newest version
Git push pushes the local version to github and saves them online as well.
Git pull retrieves the latest online version and adds any changes to your local version

Git clone is used to create an identical repository
Git status checks you local version to a online version to see if there are any differences
Git log shows a log of changes made since the git was created
Git checkout moves to/creates branches

3.3 Explain the difference between a branch and a tag.

A tag is a snapshot of code, or a savepoint while a branch is an entirely different code. In terms of video games a branch is someone's copy of the video game and may not be the same as mine, while a tag is a respawn point in that video game.

3.4 Describe at least three benefits of a version control system and include an example for each that would be related to industry.

Version control allows for ease of undoing changes, using tags we can quickly undo changes made. For example if the intern accidentally commits/pushes non-working changes we can quickly undo this using tags.
Version control also helps prevent changes from crashing in the first place. Using branches we are capable of testing changes before committing them to main. This is helpful in industry because main might be in use by customers already and we don't want to pull our software down for repairs often.
Finally, version control allows for multiple portions of a team to work on different things at the same time. This can be especially helpful in industry as you often want 2 or 3 things being completed at a time.

4 Resume and Interview Questions

Create a document that contains the following parts

Part 1: Create a resume to use to interview to be a full stack developer intern that only includes these sections

1. Summary
2. Skills
3. Relevant Experience

Part 2: Interview questions you would ask to see if someone would be a good fit on your team. Include at least 4 questions.

1. How quickly did you turn in homework last semester in the middle of the semester? (This question will tell me wether or not they are capable of keeping up with due dates)
2. How good are you at texting back friends? Family? Random People? (This will give me a decent idea of what to expect in terms of communication)
3. How good are you with new people? (Trying to get an idea for how open they are extrovert/introvert etc.)
4. Have you ever programmed with someone before? (People who have might be more knowledgable on how to work in a group, especially a group of programmers)