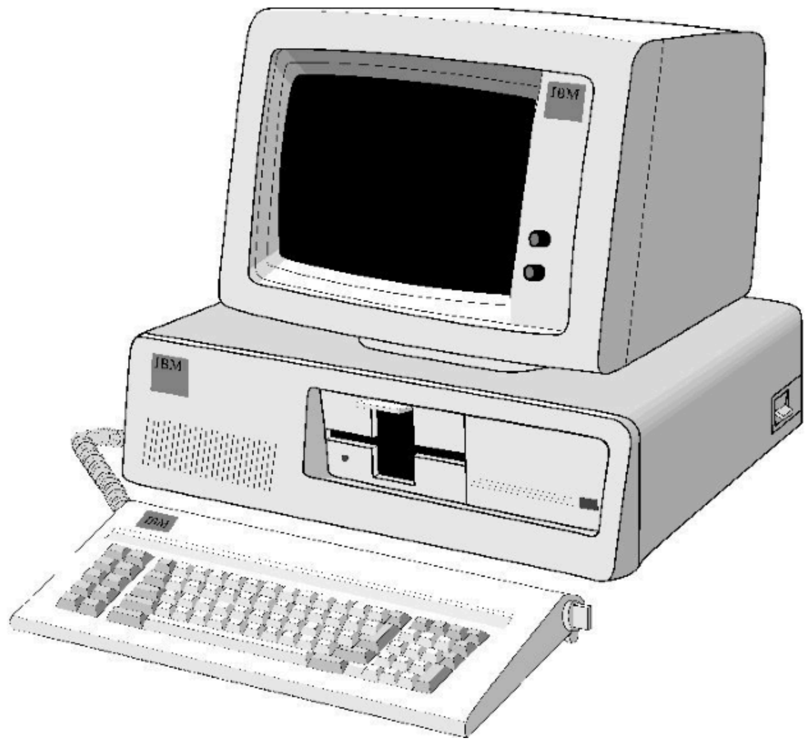
GIT / GITHUB

8.28.2020

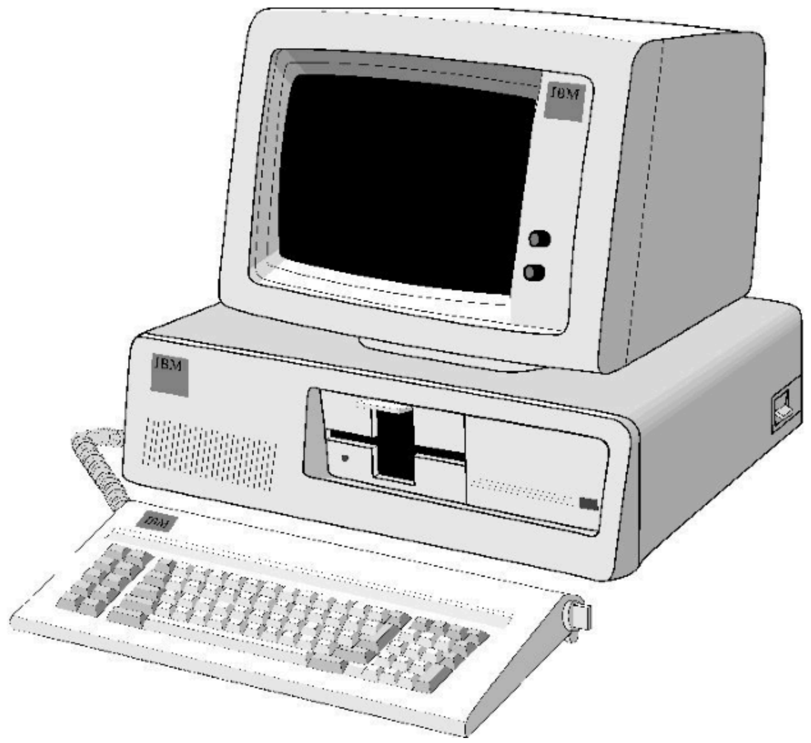
“SCRIPTS” VS. “SOFTWARE”

* SCRIPT - artisanal, organic code lovingly passed from person to person, gently modified by each, growing cruftier and less understandable every year, with no way to know who made what changes or why and no way to combine updates you made with updates someone else made



“SCRIPTS” VS. “SOFTWARE”

* analysis_test
* analysis_test2
* analysis_test2_fixed
* analysis_test_3
* analysis_final
* analysis_final_fixed
* analysis_final_new
* analysis_final_new2
* analysis_final_new2_BROKEN



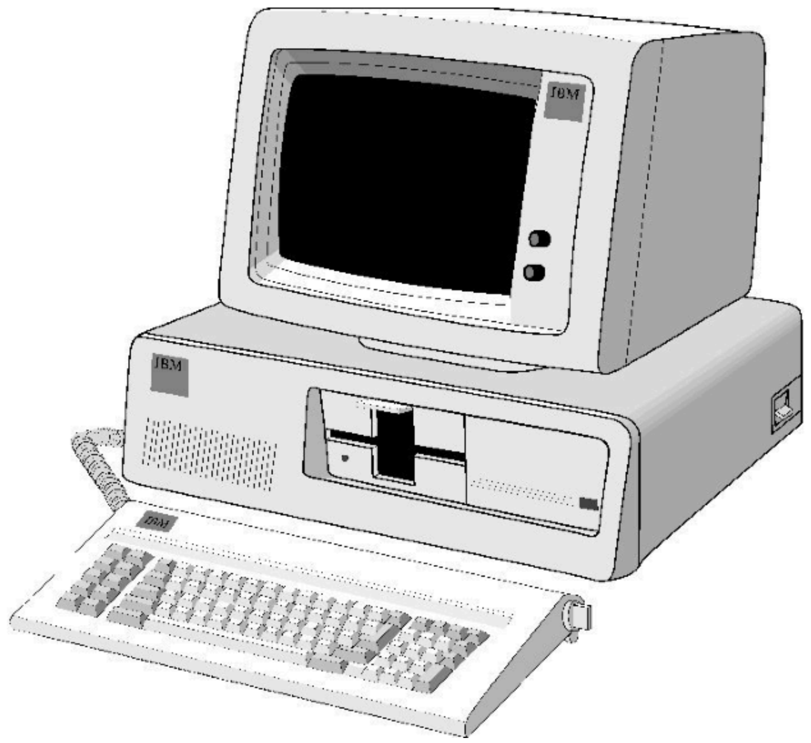
“SCRIPTS” VS. “SOFTWARE”

* I created analysis.py and gave it to some coworkers

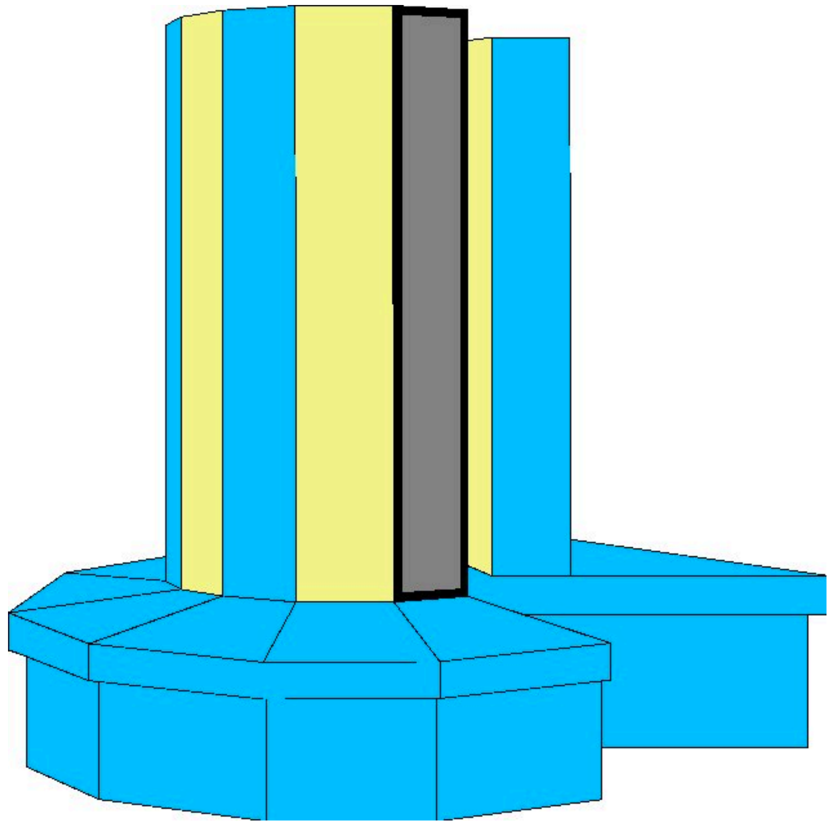
* Adora added some things => analysis_adora.py

* Bow also added some things => analysis_bow.py

* How can Bow use Adora’s changes, or vice versa?

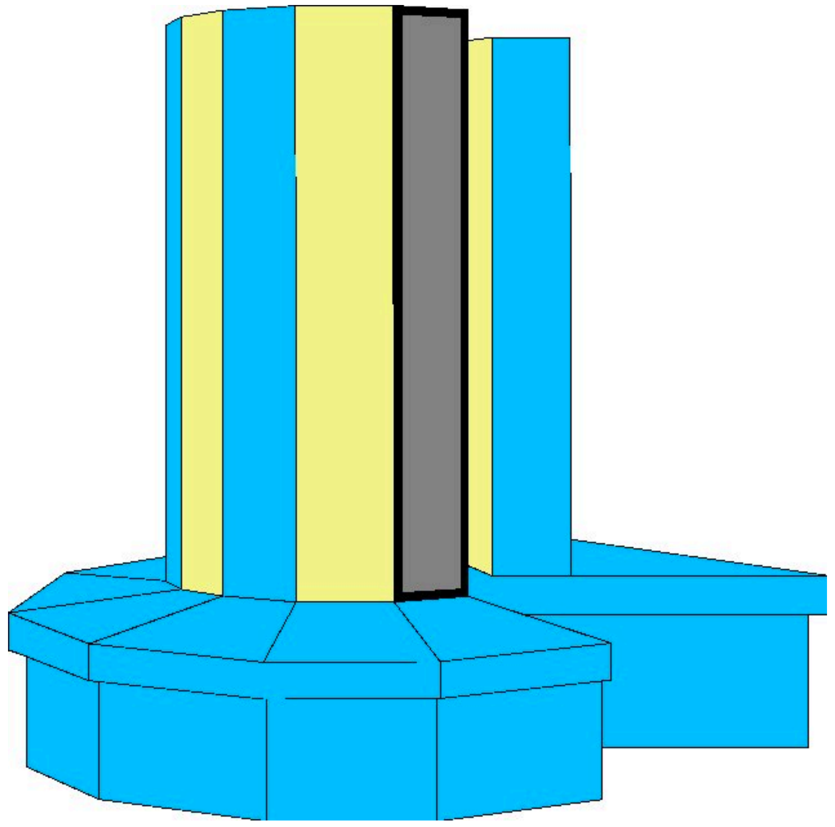


“SCRIPTS” VS. “SOFTWARE”



* (OPEN SOURCE) SOFTWARE - monolithic, standardized code shared by many, contributed to by many

“SCRIPTS” VS. “SOFTWARE”



* analysis
* analysis_utils
* README

=

VERSION CONTROL

VERSION CONTROL: GIT

GIT

Linus Torvalds

* Tracks changes to code

* Lets you share those changes with other ppl

* Lets other ppl share changes with you

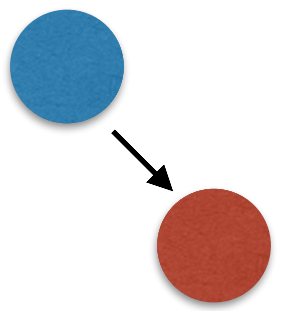
* Lets you combine your changes with other ppls changes

GIT DEMO: TRACKING YOUR CHANGES

* `git init` - *create repository*
* *[create test.py using sublime text]*
* `git status` - *see what you've changed*
* `git add test.py` - *start tracking test.py*
* `git commit -m “initial commit”`
* `git log` - *see list of commits*

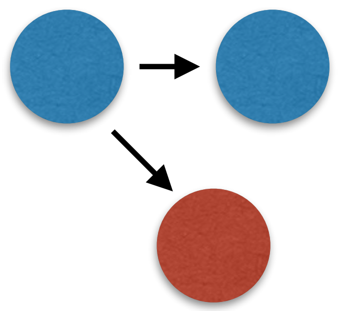
GIT DEMO: BRANCHES

* git branch major_rehaul - create a new branch
* git branch - *list all the branches*
* git checkout major_rehaul - *switch branches*
* *[edit test.py]*
* git add test.py
* git commit -m “wow such changes”
* git log



GIT DEMO: BRANCHES

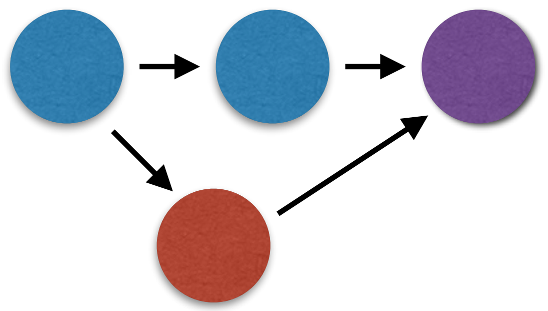
* git checkout master *- switch back to original branch*
* *[edit test.py again]*
* git add test.py
* git commit -m “a different change!”
* git log



GIT DEMO: BRANCHES

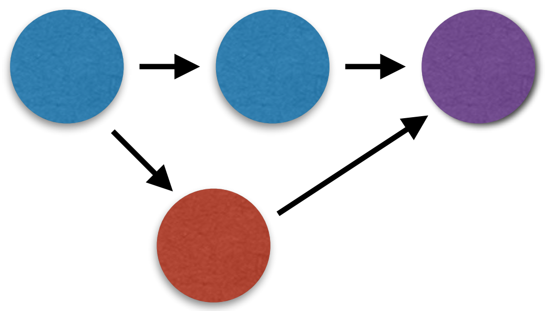
```
* git merge major_rehaul
```

* *[view test.py]*



GIT DEMO: MERGING

* *If auto-merge successful:*
    * *[save the merge message, all is good!]*
    * git log
* *If auto-merge fails:*
    * *[edit test.py, fix conflict, add, & commit]*
    * *or:* git merge --abort *- undo!*

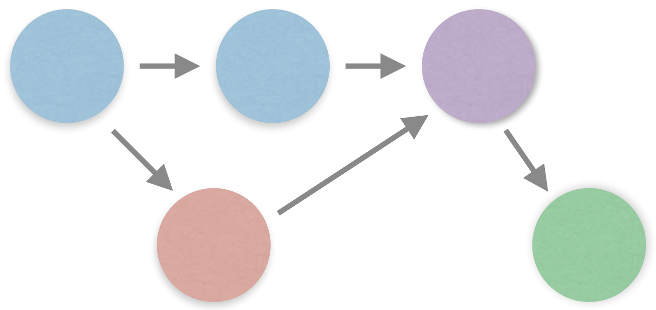


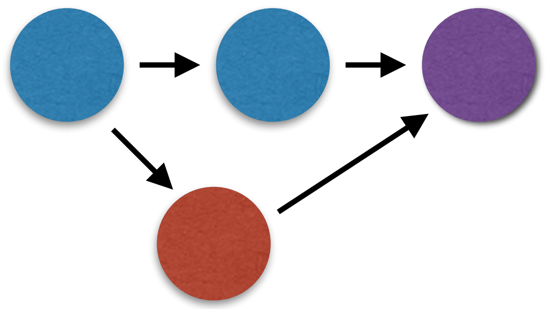
HOW DO YOU SHARE?

* git clone demo_repo demo_repo_2

* *[create new branch in demo_repo_2]*

* *[edit test.py, commit]*





HOW DO YOU SHARE?

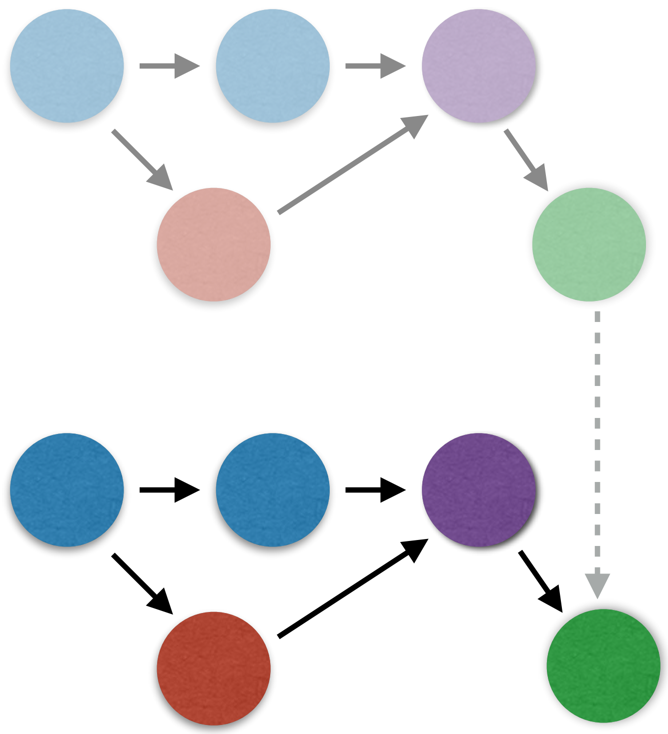
* git push --set-upstream origin new2
  - sends new2 branch to first repo

* *[switch to first repo]*

* git branch

* git checkout new2

* git log

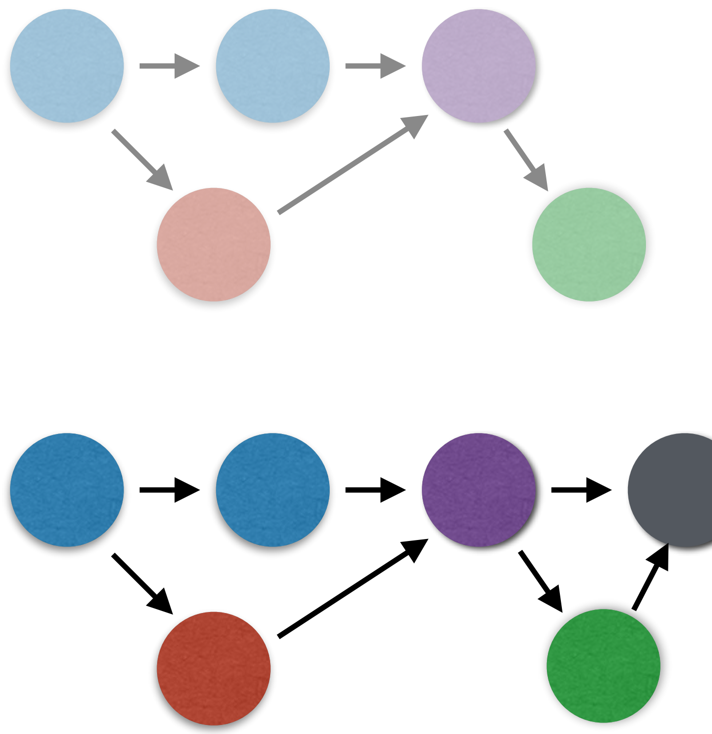


HOW DO YOU SHARE?

```
* git checkout master

* git merge new2

* git log
```



SHARING CODE: GITHUB

GITHUB

* Website that hosts git repositories

* You can **git clone** from github, and **git push** your changes to github

* [demo]

GITHUB

* To contribute on github you need a user account

* ***MINI HOMEWORK BY NEXT WEDNESDAY:***
  make a github user account,
  then “watch” the class github repo
  (https://github.com/alexhuth/ndap-fa2020)

RESOURCES

* **GIT & GITHUB TUTORIAL:** https://www.youtube.com/watch?v=iNP_KmOFqXs

* **GITHUB HELLO WORLD:** https://guides.github.com/activities/hello-world/

* **INTRO TO GIT/GITHUB:** https://www.slideshare.net/akrish/introduction-to-gitgithub-a-beginners-guide

* **MANY MORE:** http://lmgtfy.com/?q=git+github+tutorial

HAPPY WEEKEND!