1. Clone practice repo:
   1. open terminal
   2. cd ~/catkin\_ws/src
   3. git clone https://github.com/waynerobotics/practice.git
2. You need to complete a node, but right now you are on the main branch of the repository. To avoid several people writing code that conflcts with each other, everyone works on their own branch. On that branch feel free to save incremental progress in a process called “committing.” When your assignment is done and you are ready for you code to be “merged” to the main branch, we use a process called a “pull request.”
   1. Change to the repository you just cloned by typing: cd practice
   2. Use git status to observe something: git status
   3. Notice that it says you are on branch main. You don’t want to work in branch main.
   4. Make your own branch and check out that branch:
      1. First it is best practice to “pull” the most recent files from the server (there won’t be ny changes because you just cloned the repo, but if you were making a branch a week or even hours after cloning, someone could have made changes): git pull origin main
      2. Now make your branch: git branch your\_name
      3. “Check out” your branch: git checkout your\_name
      4. Type git status again and it should show you on your branch. This is where you want to work.
3. Your assignment is to complete a simple publisher node of the movement commands you’ll see everywhere in robotics. The same message type moves the turtle in the turtlesim as well as our robot. It doesn’t matter what you make the turtle do, just make it go. In order to make it go, you have to give it a linear velocity command and/or an angular velocity command, bundled into a message of type geometry\_msgs::Twist. This exercise is intended to give you some familiarity with both git and ROS basics. After you first make it work, feel free to fancy up the code if that’s in your ability (add a loop so you can change the values with cin>> and re-run the loop without exiting and recompiling the program, for example).
   1. In the package you just downloaded called practice, open the file practice/src/turtle\_go.cpp and complete it. There is guidance in the file the way and if necessary you can look at the solution.cpp file for the answers.
   2. When the code is complete, go back to your terminal and go to your catkin workspace root with cd ~/catkin\_ws
   3. compile your program with catkin\_make
   4. If there are errors read the output for a pointer to what to fix. Keep fixing errors and running catkin\_make until it compiles.
   5. open two more terminal windows (ctrl-t). In one window run roscore with the command: roscore In the other run the turtlsim with: rosrun turtlesim turtlesim\_node
   6. in your third window run you program with rosrun <package\_name> <node\_name>. You package is called “practice” and your node name is “turtle\_go, so type rosrun practice turtle\_go
   7. Observe how the turtle moves given the 3 values at the top of the program file. Play around with those values, recompile with catkin\_make, and run your node again.
4. Now you’ve written a program (or done enough that you want to preserve it) and it’s time to commit your work
   1. cd ~/catkin\_ws/src/practice
   2. git add --all
   3. git commit -m “type some meaningful notes here..like: turtle\_go node working”
5. When you are done with something enough to call it working, “push” your branch (which we named “your\_name” to the server :
   1. git push origin your\_name
   2. At this point you’ll be asked for your github account name and password. Recently, github switched from passwords and now requires a key. If you need help getting a key reach out to Lloyd or Nnamdi. Otherwise, enter your credentials.
6. Once you’ve pushed your work and you’re satisfied that it’s super-done and ready to be integrated, do a pull request.
   1. Go to the github repository and click on “pull requests”
   2. Click on “new pull request”
   3. On the left, select the branch you want your code to be added to (usually “main” for now)
   4. On the right, make sure your branch is selected
   5. Click “create pull request” and leave some notes. Usually, you will also select some reviewers.. for this assignment, select Lloyd or Nnamdi. This would give us a chance to look over your code and request changes for errors, style, etc. Only after review and approval should code be merged to the main branch. DO NOT PRESS “MERGE” on your own.
7. Relax – well done.