One Two Test Experiment

Before participants arrive

Boot the computers in Ubuntu

If the computers are booted in Windows, restart the computer and press **F12** when you see the screen with the Dell logo. On the boot menu, select the drive with the name **ubuntu**. The system will boot in Ubuntu.

left	middle	right
7	**14**	**21**
6	13	20
5	**12**	**19**
4	11	18
3	**10**	**17**
2	9	16
1	8	15

Assign participants to computers

1. From the "Programming Lab Appointments" Google Calendar, find out who is coming, and write down their names and emails in the "one-two-test-subj-info" sheet in the Lupyan Lab Google Drive.
2. Look up the "go to" language for these participants in the exercise-lab/programming-languages-screening GitHub repo.
3. Record their "go to" language in the subj info sheet.

Setting up for a participant

To set up for a participant, you need to be logged in to the lupyanlab account. Then complete the following steps:

1. Open a terminal (shortcut: Ctrl+Alt+t) and navigate to the correct experiment directory.

```
cd ~/experiments/one-two-test
```

2. Activate the right version of python in the terminal session

```
pipenv shell
```

3. Install the experiment for a participant.

```
python run.py --help
python run.py --username pierce \
              --language java   \
              --problem hello-world saddle-points
python run.py -u pierce -l java -p hello-world saddle-points
```

Summary

```
cd ~/experiment/pilot
pipenv shell
python run.py -u [username] -l [language] -p [problem1] [prob
```

After the run.py script completes, log out of the lupyanlab account, and log back in as the participant.

Log in for the participant

1. Log in to their account for them, using their usernames as their passwords.

Note: You may have to click through a welcome screen that pops up because it's the first time this user has logged on.

2. Open a navigator window (click on the Files app in the sidebar) and navigate to the problems directory.
3. Open each of the problems in the expected IDE.

Java: Eclipse

Using the app launcher (lower left corner of the Ubuntu GUI), open the Eclipse app. Then import the problem as a Gradle project from the Eclipse app File menu.

```
File > Import > Gradle project
```

Python: PyCharm

Using the app launcher (lower left corner of the Ubuntu GUI), open the PyCharm app. Then open the problem as a Python project from the PyCharm app File menu.

When the participants arrives

Consent forms

Give each participant a consent form to read and sign.

Explain what will happen in the experiment

As you know, this is a research project on problem solving in different programming languages. In this experiment, you will work on two problems. The first one is a "hello world" warm up problem. All you need to do is get the program to print "Hello World". It's a warm up problem because we want you to get comfortable running the automated unit tests that we will use to verify that your solution is correct. Once you've finished the hello world problem, let us know, and we'll let you move on to the second problem, called "the saddle points problem." The saddle points problem is done in two parts. After you finish part 1, let one of us know, and we will come over and get you started on part 2.

In this experiment, we are interested in how easily people can pick up and continue code that was written by someone else. So that means we want you to use informative variable names and include comments wherever you think it might be helpful for someone else reading your code.

You should also know that it's ok to use the internet to Google things, but please don't try to find exact solutions to the problem you are solving. It will be really easy for us to tell if you've just copied and pasted in an entire solution you found somewhere else on the web, but if you need to look up the docs for a builtin function or something, go right ahead.

Now we will get you set up at your computer, show you how to run the tests, and let you get started on the first problem. Once you've finished the first problem, we will come over and you can show us that you can run the tests and that the tests pass. Then you can start on the second problem.

Show each participant how to run the tests

Java: Eclipse

Run the tests by opening the test file and pressing the green "Play" button in the menu bar. Run the file as a JUnit test.

OR: run the tests from the command line by navigating to the problem directory and running `gradle test`:

```
cd ~/problems-java/saddle-points/
gradle test
```

Python: PyCharm

Run the tests by opening the test file and selecting "Run 'pytest for test_file_name.py' ".

OR: run the tests from the command line by navigating to the problem directory and running `pytest`:

```
cd ~/problems-python/saddle-points/
pytest
```

Switching from part1 to part2

When a participant has finished part1, open a terminal window (Ctrl+Alt+T) and follow these steps to start them on part2.

```
# Navigate to the problem directory
cd ~/problems-[language]/saddle-points

# See if there are any uncommited changes
git status

# If you see RED when you run `git status`, run the followin
git add . && git commit -m "Finished part1"

# If you don't see RED or after you have committed, run the 
git tag part1done
git merge -m "Add part2" origin/part2

# Open up the new part2 tests file for the participant
```

After a participant is done

Push their changes to GitHub.

```
# Navigate to the problem directory
cd ~/problems-[language]/saddle-points

# See if there are any uncommited changes
git status

# If you see RED when you run `git status`, run the followin
git add . && git commit -m "End of experiment"

# If you don't see RED or after you have committed, run the 
git push
```

Teardown the experiment.

1. Log out of the participant's account, and log back in as lupyanlab.
2. Navigate to the experiment directory, and run the stop.py script.

```
cd ~/experiments/one-two-test
pipenv shell
python stop.py -u [username]
```