Wentworth Institute of Technology
COMP3071 – Intro To Programming Languages
Fall 2016, Kreimendahl

Lab Assignment 3 – Due Nov 5 at 11:59PM

This lab assignment introduces the programming language Prolog, a logical programming language.

1. Install Software

First, download and install a common compiler for Prolog, SWI-Prolog, from http://www.swi-prolog.org/download/devel. Note that this is the developer version, not the stable version. The stable version won't run on newer Macs.

2. GitLab

We are going to use GitLab[1] to distribute starter code, as well as for you to submit this lab assignment. GitLab is one of several popular web-based platforms (e.g. GitHub, BitBucket) that provide *version control*[2] services and software.

3. Complete the Assignment

The steps up till now you only need to do once. By contrast, you will follow the remaining steps for each assignment for the rest of the semester. Before you begin, here is the big picture of the process you will complete for each assignment:

1. Visit the assignment **repository**
2. **Fork** the repository (i.e. make a *private* copy that is owned by your account)
3. Import or download the repository
4. Write your code, saving your work frequently (**commit** and **push**)
5. Confirm your final submission is accessible to the instructor
6. Set appropriate **access** to your forked repository (add the instructor)

3.1. Visit the Assignment Repository

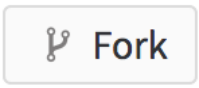
To begin, visit the website for this assignment's repository (a link will be listed under "Your Projects" when you visit the GitLab main dashboard – accessible by clicking the Wentworth logo at the top-middle of any page). https://gitlab.com/comp3071fk/la3

3.2. Fork the Assignment Repository

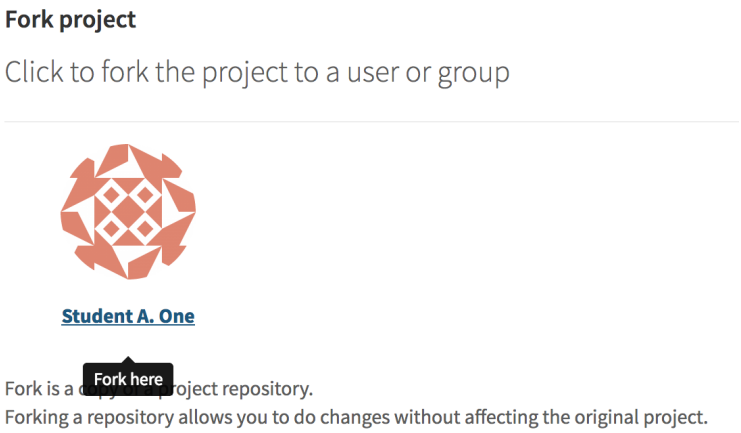
On the repository website, click the "Fork" button – this will create a copy of the repository that your account will own (this will be where you will submit your work).

---

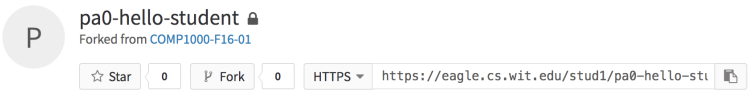
[1] https://gitlab.com
[2] https://en.wikipedia.org/wiki/Revision_control



The resulting page will ask which user/project should own the forked repository – click the box that represents your user.



If successful, you will be brought to the website of your forked repository (note that it will include a new button showing the repository from which it was forked).



-

3.3. Download/Import files

You now have a private copy of the assignment in your repository on GitLab's servers – it's time to bring this to your computer, so you can write some Prolog code! There are graphical front ends to git, or you can use git itself from the command line.

3.4. Written Code

See section 4 for specifications about this assignment.

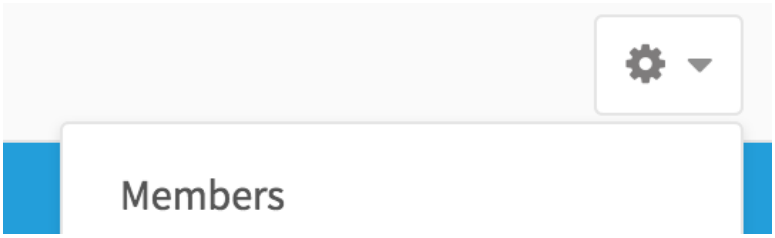
3.5. Confirm Submission

Now complete this assignment, as you will EVERY assignment in this class:

- Save, commit, and push all code changes
- Confirm the latest code is visible via the "Files" section of your repository website
- Add instructor access to your project (See section below)

3.6. Set Repository Access

The lock icon next to the name of the repository means that no one can access your repository unless you provide explicit access. To grant access, click the "Members" link from the "gears" (i.e. settings) menu:



On the resulting page, find your instructor (kreimendahlf) in the **People** field, set **Project Access** to Developer, and click the "Add users to project" button. If this succeeds, you should see your user and the instructor, tagged as a Developer. Submission consists of you adding your instructor to the project.

Remember

- If your repository is shared with any additional users, we will assume that you were cheating by letting other students see your work.
- If your instructor does not have Developer access, your work cannot be graded and you will receive a 0.

4. Specifications

The purpose of this lab is to familiarize with syntax and some simple functionality in Prolog

4.1. Language

This assignment should be completed in Prolog. Future assignments will be written in different languages, but this one must be completed in Prolog, using `SWI-Prolog` and any text editor that you are comfortable with. Please add sufficient comments to your code so that it's clear to a smart but uninformed reader what your intention is with a code block.

4.2. PA3a: your own knowledge base

For the first part, implement your own knowledge base and provide example queries on that knowledge base. Write your clauses in `PA3a.pl`. Provide at least 6 facts and 4 rules and one conjunction in a rule. Provide at least three non-trivial queries in the comment below your clauses. (Non-trivial means they are not querying explicit facts.)

Your knowledge base can entail any information you would like, with any relationships between the atoms that makes sense. You can use variables if you want, but you aren't required to.

4.3. PA3b: Northern German trains

For the second part, refer to `PA3b.pl`, which has a bunch of facts already in it. The goal is to be able to query if two stations are connected to each other via any route using the `connected/2` predicate, and we will need several additions to do that.

First of all, write a rule called `adjacent_/2` that generalizes the adjacent stations we have. If A is `adjacent` to station B, then station A is `adjacent_` to station B. Also, station B is `adjacent_` to station A. That way we can go either way between adjacent stations. The reason we don't just extend the predicate

`adjacent` with variable definitions is because Prolog attempts unification from the top to bottom of the rule list and it will unify two stations in an infinite loop if we kept with the same predicate name.

Secondly, define a `connected/2` predicate that says if two stations are adjacent, they are connected.

Third, add another rule for the `connected/2` predicate that says if two stations are connected via a third station, they are connected to each other. With those three predicates, we should be able to query if any two stations are connected to each other!

4.4. Testing

Test your functions by loading `la3a.pl` into `SWI-Prolog` by going to File->Consult… and navigating to the file you want to load. Verify that queries work the way you expect.