Lab 0

The purpose of this lab is to familiarize you with this term's lab system and to diagnose your programming ability and facility with Python. 6.034 uses Python for all of its labs, and you will be called on to understand the functioning of large systems, as well as to write significant pieces of code yourself.

While coding is not, in itself, a focus of this class, artificial intelligence is a hard subject full of subtleties. As such, it is important that you be able to focus on the problems you are solving, rather than the mechanical code necessary to implement the solution.

Python resources

Some resources to help you knock the rust off of your Python:

- Any of the many good Python handbooks out there, such as:
  - Dive Into Python, for experienced programmers
  - O'Reilly's Learning Python
  - Think Python, for beginning programmers
- The standard Python documentation, at [1] (the Library Reference and the Language Reference are particularly useful, if you know what you're looking for)

Python

There are a number of versions of Python available. This course will use standard Python ("CPython") from http://www.python.org/. If you are running Python on your own computer, you should download and install Python 2.5 or Python 2.6 from http://www.python.org/download/ . All the lab code will require at least version 2.3.

Mac OS X comes with Python 2.3 pre-installed, but the version you can download from python.org has better support for external libraries and a better version of IDLE.

Answering questions

The main file of this lab is called `lab0.py`. Open that file in IDLE. The file contains a lot of incomplete statements that you need to fill in with your solutions.

The first thing to fill in is a multiple choice question. The answer should be extremely easy. Many labs will begin with some simple multiple choice questions to make sure you're on the right track.

Run the tester

Every lab comes with a file called `tester.py`. This file checks your answers to the lab. For problems that ask you to provide a function, the tester will test your function with several different inputs and see if the output is correct. For multiple choice questions, the tester will tell you if your answer was right. Yes, that means that you never need to submit wrong answers to multiple choice questions.

The tester has two modes: "offline" mode (the default), and "online" or "submit" mode. The former runs some basic, self-contained internal tests on your code. It can be run as many times as you would like.

The latter runs some more tests, some of which may be randomly generated, and uploads your code to the 6.034 grader for grading.

You can run the online tester as many times as you want. If your code fails a test, you can submit it and try again. Because you always have the opportunity to fix your bugs, you can only get a 5 on a problem set if it passes all the tests. If your code fails a test, your grade will be 4 or below.

Using IDLE

If you are using IDLE, or if you do not have easy access to a command line (as on Windows), IDLE can run the tester.

Open the `tester.py` file and run it using Run Module or F5. This will run the offline tests for you. When the offline tests pass (or when you're up against a deadline, or when you have questions for the staff) you can

```
test_online()
```

to submit your code and run the online tests.

In fact, it will run the submission and online test just as soon as you pass the offline tests, saving you a few keystrokes.

You should run the tester (and submit!) early and often. Think of it as being like the "Check" button from 6.01. It makes sure you're not losing points unnecessarily. Submitting your code makes it easy for the staff to look at it and help you.

Using the command line

If you realize just how much emacs and/or the command line rock, then you can open your operating system's Terminal or Command Prompt, and `cd` to the directory containing the files for Lab 0. Then, run:

```
python tester.py
```

to run the offline tester, and

```
python tester.py submit
```

to submit your code and run the online tester.

You should run the tester (and submit!) early and often. Think of it as being like the "Check" button from 6.01. It makes sure you're not losing points unnecessarily. Submitting your code makes it easy for the staff to look at it and help you.

Python programming

Now it's time to write some Python.