Homework 1

1) Get an account on GitHub or BitBucket (or some other equivalent service) and use Git to track your code, and push to this repository.

- get a 'private' repository if you'd like, to keep the internet from sniffing you out

- if you have a private repo, find Isaac on GitHub or BitBucket, and share your repo with him (account: ishivvers)

- while writing your solution, make at least 5 separate commits with real comments (we will check history)

- push your final version before 5:00pm Sunday, and send Isaac an email explaining exactly how to clone it

- from here on out, log your homework in the same Git repo, and we will run a script that pulls the repository every Sunday at 5:00. This is how your future homeworks will be submitted!

Homework 1

2) Write a package called 'CalCalc', with a method called 'calculate' that evaluates any string passed to it, and can be used from either the command line (using argparse with reasonable flags) or imported within Python. Feel free to use something like eval(), but be aware of some of the nasty things it can do, and make sure it doesn't have too much power: http://nedbatchelder.com/blog/201206/eval_really_is_dangerous.html

EXAMPLE:

```
$ python CalCalc.py -s '34*28'
$  952
---- AND, from within Python ----
>>> from CalCalc import calculate
>>> calculate('34*20')
>>> 952
```

2a) To make this more awesome, have your function interact with the Wolfram|Alpha API to ask it what it thinks of the difficult questions. NOTE: this added bit will only be worth 1 point out of 12, but it makes the program way way way more awesome.
To make this work, experiment with urllib2 and a URL like this:
'`http://api.wolframalpha.com/v2/query?input=XXXXX&appid=UAGAWR-3X6Y8W777Q`'
where you replace the XXXXX with what you want to know. NOTE: the '`&appid=UAGAWR-3X6Y8W777Q`' part is vital; it is a W|A AppID I signed up for the class. Feel free to use that one, or you can get your own, and read more about the API, here: http://products.wolframalpha.com/api/
And you can explore how it works here: http://products.wolframalpha.com/api/explorer.html

EXAMPLE:

```
$ python CalCalc.py 'mass of the moon in kg'
$  7.3459e+22
---- AND, from within Python ----
>>> from CalCalc import calculate
>>> calculate('mass of the moon in kg',  return_float=True) * 10
>>> 7.3459e+23
```

Homework 1

3) Create a proper setup.py and turn your module into a proper Python Distribution, so that we can install it and use it.

Folder Hierarchy:

```
Your_Homework1_Folder/CalCalc
Your_Homework1_Folder/CalCalc/setup.py
Your_Homework1_Folder/CalCalc/__init.py__
Your_Homework1_Folder/CalCalc/README.txt
Your_Homework1_Folder/CalCalc/CalCalc.py
```

Include at least 5 test functions in `CalCalc.py`, and test with `nosetest`, to make sure it behaves the way you think it should.

EXAMPLE:

```
CalCalc.py:
[...]
def calculate([...]):
    [...]
def test_1():
    assert abs(4. - calculate('2**2')) < .001
```

When grading, I will create a virtual environment and attempt to install your module by running:

```
python setup.py install
```

And then I will test it using `nosetest`, and I will attempt to run it from the command line and by importing it in Python. Explain everything in your README.txt

You will be graded on: Git usage, nosetest, packaging your code, and your code itself