Python Regular Expressions Date Parsing

Write a Python program called "dates.py" that takes as a single, positional argument a string and attempt to parse it as one of the given date formats. If given no argument, it should print a usage statement. It does not need to respond to `-h|--help`, so you could use `new_py.py` without the argparse flag.

```
$ ./dates.py
Usage: dates.py DATE
```

If you are able to match one of the acceptable format strings below, print the date in a standard "YYYY-MM-DD" format. If only given year and month, e.g. "12/06," use "1" as the day.

These are the formats you will be given:

```
$ cat eg_dates.txt
2012-03-09T08:59
2012-03-09T08:59:03
2017-06-16Z
2015-01
2015-01/2015-02
2015-01-03/2015-02-14
20100910
12/06
2/14
2/14-12/15
2017-06-16Z
Dec-2015
Dec, 2015
March-2017
April, 2017
```

Here is the expected output:

```
$ while read -r DATE; do ./dates.py "$DATE"; done < eg_dates.txt
2012-03-09
2012-03-09
2017-06-16
2015-01-01
2015-01-01
2015-01-03
2010-09-10
2006-12-01
2014-02-01
2014-02-01
2017-06-16
2015-12-01
```

```
2015-12-01
2017-03-01
2017-04-01
```

If you are unable to parse the argument, print "No match":

```
$ ./dates.py foo
No match
$ ./dates.py 1999.12.31
No match
```

While there are date parsing modules, I do not want you to use those in your code. Please write your own regular expressions and parsing code.

Test Suite

A passing test suite looks like this:

```
$ make test
python3 -m pytest -v test.py
============================= test session starts ==============================
platform darwin -- Python 3.6.8, pytest-4.2.0, py-1.7.0, pluggy-0.8.1 -- /anaconda3/bin/pyth
cachedir: .pytest_cache
rootdir: /Users/kyclark/work/worked_examples/regex, inifile:
plugins: remotedata-0.3.1, openfiles-0.3.2, doctestplus-0.2.0, arraydiff-0.3
collected 13 items

test.py::test_usage PASSED                                               [  7%]
test.py::test_00 PASSED                                                  [ 15%]
test.py::test_01 PASSED                                                  [ 23%]
test.py::test_02 PASSED                                                  [ 30%]
test.py::test_03 PASSED                                                  [ 38%]
test.py::test_04 PASSED                                                  [ 46%]
test.py::test_05 PASSED                                                  [ 53%]
test.py::test_06 PASSED                                                  [ 61%]
test.py::test_07 PASSED                                                  [ 69%]
test.py::test_08 PASSED                                                  [ 76%]
test.py::test_09 PASSED                                                  [ 84%]
test.py::test_10 PASSED                                                  [ 92%]
test.py::test_bad_input PASSED                                           [100%]

========================== 13 passed in 4.88 seconds ===========================
```