Workout Of (the) Day (WOD)

Write a Python program called `wod.py` that will create a Workout Of (the) Day (WOD) from a list of exercises provided in CSV format (default `wod.csv`). Accept a `-n|--num_exercises` argument (default 4) to determine the sample size from your exercise list. Also accept a `-e|--easy` flag to indicate that the reps should be cut in half. Finally accept a `-s|--seed` argument to pass to `random.seed` for testing purposes. You should use the `tabulate` module to format the output as expected.

The input file should be comma-separated values with headers for "exercise" and "reps," e.g.:

```
$ tablify.py wod.csv
+---------------+--------+
| exercise      | reps   |
|---------------+--------|
| Burpees       | 20-50  |
| Situps        | 40-100 |
| Pushups       | 25-75  |
| Squats        | 20-50  |
| Pullups       | 10-30  |
| HSPU          | 5-20   |
| Lunges        | 20-40  |
| Plank         | 30-60  |
| Jumprope      | 50-100 |
| Jumping Jacks | 25-75  |
| Crunches      | 20-30  |
| Dips          | 10-30  |
+---------------+--------+
```

You should use the range of reps to choose a random integer value in that range.

Expected Behavior

```
$ ./wod.py -h
usage: wod.py [-h] [-f str] [-s int] [-n int] [-e]

Create Workout Of (the) Day (WOD)

optional arguments:
  -h, --help            show this help message and exit
  -f str, --file str    CSV input file of exercises (default: wod.csv)
  -s int, --seed int    Random seed (default: None)
  -n int, --num_exercises int
```

```
                        Number of exercises (default: 4)
  -e, --easy            Make it easy (default: False)
$ ./wod.py
Exercise      Reps
----------  ------
Crunches        26
HSPU             9
Squats          43
Pushups         36
$ ./wod.py -s 1
Exercise        Reps
-------------  ------
Pushups           32
Jumping Jacks     56
Situps            88
Pullups           24
$ ./wod.py -s 1 -e
Exercise        Reps
-------------  ------
Pushups           15
Jumping Jacks     27
Situps            44
Pullups           12
$ ./wod.py -f wod2.csv -n 5
Exercise               Reps
--------------------  ------
Erstwhile Lunges          9
Existential Earflaps     32
Rock Squats              21
Squatting Chinups        49
Flapping Leg Raises      17
```

Discussion

It's recommended you use the `csv.DictReader` module to parse the CSV files. You will then need to split the "reps" fields like "20-50" into a low and high values that are coerced into integer values. For the purposes of this exercise, you can assume the CSV files you are given will have the correct headers and the fields will be correctly formatted.

You should use the `random` module to select a sample of exercises, e.g.:

```
>>> import random
>>> random.sample(range(10), k=3)
[1, 6, 4]
```

```
>>> random.sample(range(10), k=3)
[8, 5, 6]
```

So first focus on parsing the input CSV into something you can `sample`, like a list or a dictionary. I chose to create a data structure that is a list of tuples containing the name of the exercise, the low range, and the high range for the reps:

```
[('Burpees', 20, 50),
 ('Situps', 40, 100),
 ('Pushups', 25, 75),
 ('Squats', 20, 50),
 ('Pullups', 10, 30),
 ('HSPU', 5, 20),
 ('Lunges', 20, 40),
 ('Plank', 30, 60),
 ('Jumprope', 50, 100),
 ('Jumping Jacks', 25, 75),
 ('Crunches', 20, 30),
 ('Dips', 10, 30)]
```

Then I can get a random rep value using `random.randint`, e.g.:

```
>>> random.randint(5, 10)
6
>>> random.randint(5, 10)
8
```

Test Suite

A passing test suite looks like the following:

```
$ make test
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/python/practical_python_for_data_science/ch08-python-parsing/ex
plugins: remotedata-0.3.1, openfiles-0.3.2, doctestplus-0.2.0, arraydiff-0.3
collected 5 items

test.py::test_usage PASSED                                                  [ 20%]
test.py::test_runs01 PASSED                                                 [ 40%]
test.py::test_runs02 PASSED                                                 [ 60%]
test.py::test_runs03 PASSED                                                 [ 80%]
test.py::test_runs04 PASSED                                                 [100%]
```

```
=========================== 5 passed in 0.50 seconds ===========================
```