Hamming Distance

The Hamming Distance is the "edit distance" between two strings. For example, the distance between "foo" and "boo" is 1 (change the "f" to "b"):

```
f o o
b o o
1 0 0 = 1
```

While the distance from "foo" to "faa" is 2 ("oo" to "aa"):

```
f o o
f a a
0 1 1 = 2
```

The distance from "foo" to "foobar" is 3 because we would have to add or subtract 3 letters to make either string into the other.

```
f o o
f o o b a r
0 0 0 1 1 1 = 3
```

For the purposes of our exercise, we will always start at the left side of both strings. That is, "room" and "broom" might seem like they are one edit apart (add a "b" to "room"), but their distance should be calculated as 4:

```
r o o m
b r o o m
1 1 0 1 1 = 4
```

Write a Python program called `hamming.py` that a single position argument which is a readable file that will contain pairs of words like so:

```
$ cat input2.txt
foo boo
foo faa
foo foobar
```

For each line of input, the program should print the Hamming distance in a field 8 characters wide followed by a colon and the two words each in a field 20 characters wide:

```
$ ./hamming.py input2.txt
       1:foo                 boo
       2:foo                 faa
       3:foo                 foobar
```

The program should also accept a `-m` or `--min` option that is a minimum integer value of the Hamming distance to print a pair:

```
$ ./hamming.py input2.txt -m 2
       2:foo                 faa
       3:foo                 foobar
```

The distance can also be used to find the number of point mutations between, say, two strands of DNA:

```
$ ./hamming.py input3.txt
       9:TAGGGCAATCATCCGAG   ACCGTCAGTAATGCTAC
      10:TAGGGCAATCATCCGG    ACCGTCAGTAATGCTAC
```

These mutations are sometimes called "Single Nucleotide Polymorphisms" (SNP, prounouced "snip") or "Single Nucleotide Varations" (SNV).

A passing test suite looks like this:

```
$ make test
pytest -xv test.py
============================= test session starts ==============================
...
collected 8 items

test.py::test_exists PASSED                                              [ 12%]
test.py::test_usage PASSED                                               [ 25%]
test.py::test_bad_file PASSED                                            [ 37%]
test.py::test_input1 PASSED                                              [ 50%]
test.py::test_input2 PASSED                                              [ 62%]
test.py::test_input3 PASSED                                              [ 75%]
test.py::test_input2_min2 PASSED                                         [ 87%]
test.py::test_input3_min10 PASSED                                        [100%]

============================== 8 passed in 0.42s ===============================
```