This assignment gives up to 15 points. You should answer the first two questions in plain text (via Moodle), and upload a program assignment01.py as a solution two the third task. The program should run as is.

Look at the following FST.

```
import pynini as np
fst = (pn.a("a") | pn.a("e")) + (pn.t("a", "0").closure() | pn.t(pn.a("i").plus,
pn.a("0").star)) + pn.a("x")
fst.optimize()
```

1. What is the input language of this FST? Give a regular expression that is equivalent to it's input language. (1 point)
2. Give inputs that are mapped by fst to 0 outputs, 1 output, 2 outputs, and more than 2 outputs. (4*0.5 = 2 points)
3. Make program that maps numbers in text to English words (up to 999). Use FSTs to do that. Use provided assignment01_template.py as starter.

It's a good idea to implement this with two FSTs. The first FST will factorize a number into into sequences annotated with powers of ten:

```
0    -> 0
1    -> 1
10   -> 1^
23   -> 2^ 3
203 ->  2^^ 3
```

A second FST will convert the factorized form into words:

```
0      -> zero
1      -> one
1^     -> ten
2^ 3   -> twenty three
2^^ 3  -> two hundred and three
```

Finally, you should also handle decimals. Decimals should be verbalized digit-by-digit: 2.569 whould be converted into "two point five six nine".

The program should read a digit from stdin, and provide two outputs: factorized form and final form (a template for this is provided).

E.g.

```
./assignment01.py
Please enter a number (Ctrl-D to exit): 112
Result in factorized form
1^^ 1^ 2
Result in words
```

```
['hundred twelve', 'one hundred twelve']
```

Scoring:

- Factorization works: 3 points
- Integers are converted correctly: 5 points
- Decimals are converted correctly: 2 points
- Your program should have a command line option "-r" that inverts the program's behaviour by allowing conversion from words to numbers ("one" -> "1"): 2 points

Note that for some numbers, many outputs are allowed, e.g. "102" can be verbalized both into "one hundred and two" and "hundred and two". Your program should output all possible forms.