Problem context

An English document has been corrupted in an unusual manner. Each alphabetic word (no numbers, but apostrophes allowed) has had every other character replaced with a # character. For example, a sentence like

```
I'm here. It's a cat.
```

would appear in the document as

```
I#m h#r#. I#'# a c#t.
```

We have extracted the tokens from this corrupted document and have provided the list of tokens to you as `corrupted_tokens.txt`. Note that the order of tokens in the list is the same as how they appeared in the original document, and that surrounding punctuation like `.` and `,` have been removed.

As an example, if the uncorrupted document is:

```
I'm here. It's a cat.
```

the its corresponding corruption followed by tokenization would be:

```
I#m
h#r#
I#'#
a
c#t
```

In [1]:
```
# read in corrupted tokens

CORRUPTED_TOKENS = []

with open('corrupted_tokens.txt', 'r', encoding='utf-8') as f:
    for line in f:
        tok = line.strip()
        CORRUPTED_TOKENS.append(tok)

# print a sample of the corrupted tokens
print('num corrupted tokens =', len(CORRUPTED_TOKENS))
print('first 20 corrupted tokens ->')
for tok in CORRUPTED_TOKENS[:20]:
    print(' ', tok)
```

```
num corrupted tokens = 29293
first 20 corrupted tokens ->
  T#a#
  n#t#
  r#c#l#s
  a#
  e#p#r#e#c#
  T#e
  p#s#e#g#r#
  w#r#
  s#n#
  f#r
  t#
  c#m#
  u#
  i#
  t#e
  b#w
  a#d
  s#e
  a
  f#n#
```

Task

Try to recover the corrupted tokens as best you can and write your guess of the original tokens to a file `recovered_tokens.txt`, one token per line. Each line in `recovered_tokens.txt` should match its corresponding line in `corrupted_tokens.txt`.

To enable you to do this, you are given a tokenization of an uncorrupted training document, i.e. the `training_tokens.txt` file. This file also one token per line and its tokens appear in the same order as in the uncorrupted training document. You can use this along with whatever external material you deem necessary to inform your recovery of the corrupted tokens.

You may use whatever language and/or tools you deem necessary.

Please submit your solution consisting of:

1. your `recovered_tokens.txt` file. This should contain the same number of tokens/lines as `corrupted_tokens.txt`. Additionally, case is not important for the recovered tokens, i.e. we treat `Prince` the same as `prince` the same as `PRINCE` .
2. your code for your attempt in recovering the corrupted tokens
3. a short write-up of your methodology

In [2]:
```
TRAINING_TOKENS = []
with open('training_tokens.txt', 'r', encoding='utf-8') as f:
    for line in f:
        tok = line.strip()
        TRAINING_TOKENS.append(tok)

# print a sample of the training tokens
print('num training tokens =', len(TRAINING_TOKENS))
print('first 20 training tokens ->')
for tok in TRAINING_TOKENS[:20]:
    print(' ', tok)
```

```
num training tokens = 406223
first 20 training tokens ->
  PREFACE
  Most
  of
  the
  adventures
  recorded
  in
  this
  book
  really
  occurred
  one
  or
  two
  were
  experiences
  of
  my
  own
  the
```

Example solution

As an example for how the `recovered_tokens.txt` should look like, here's a naive example solution that attempts recovery by replacing all `#` characters with `e` and generates a `recovered_tokens_EXAMPLE.txt` file with all lower-case tokens.

In [3]:
```
# example predictor that lower-cases a corrupted token and attempts reco
very by replacing '#' with 'e'
def dummy_predictor(corrupted_token):
    return corrupted_token.lower().replace('#', 'e')
```

In [4]:
```
# recover a guess of the original tokens using the dummy predictor
RECOVERED_TOKENS_EXAMPLE = []

for tok in CORRUPTED_TOKENS:
    recovered_tok = dummy_predictor(tok) if len(tok) > 1 else tok
    RECOVERED_TOKENS_EXAMPLE.append(recovered_tok)

# print a sample of the recovered tokens
print('num recovered tokens (example) =', len(RECOVERED_TOKENS_EXAMPLE))
print('first 20 recovered tokens (example) ->')
for tok in RECOVERED_TOKENS_EXAMPLE[:20]:
    print(' ', tok)

# write recovered tokens file from our guess of the original tokens
with open('recovered_tokens_EXAMPLE.txt', 'w', encoding='utf-8') as f:
    for tok in RECOVERED_TOKENS_EXAMPLE:
        f.write('%s\n' % tok)
```

```
num recovered tokens (example) = 29293
first 20 recovered tokens (example) ->
  teae
  nete
  receles
  ae
  eepereeece
  tee
  peseeegere
  were
  sene
  fer
  te
  ceme
  ue
  ie
  tee
  bew
  aed
  see
  a
  fene
```