```python
import nltk
from nltk.corpus import wordnet
from nltk.stem import WordNetLemmatizer
from nltk import pos_tag
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
from nltk.tokenize import sent_tokenize,
word_tokenize
import string

class Text:
    def __init__(self, text):
        self.text = text

    def split_into_sent(self):
        sentences = sent_tokenize(self.text)
        return sentences

    def __str__(self):
        return self.text

class Sent:
    def __init__(self, sentences):
        self.sentences = [sentence for sentence in
sentences]

    def get_sentence(self, index):
        if index < 0 or index >= len(self.sentences):
            raise IndexError("Index out of range")
        return self.sentences[index]

    def __str__(self):
        return ' '.join(self.sentences)

class LexEntry:
    wnl = WordNetLemmatizer()

    pos = {"noun", "pronoun", "verb", "adjective", "
adverb", "preposition", "determiner", "conjunction",
           "interjection", }

```

```
39      def __init__(self, sentence):
40          self.wnl = WordNetLemmatizer()
41          self.lex_entries = self.process_text(sentence
    ).split()
42
43      def process_text(self, text):
44          translator = str.maketrans('', '', string.
    punctuation)
45          cleaned_text = text.translate(translator)
46          return cleaned_text.lower()
47
48      def split_into_words(self):
49          return self.lex_entries
50
51      def get_word_len(self):
52          return [len(word) for word in self.
    lex_entries]
53
54      def get_word_index(self, word):
55          try:
56              return self.lex_entries.index(word)
57          except ValueError:
58              return -1
59
60      def get_pos(self, word):
61          tag = pos_tag([word])[0][1]
62          if tag.startswith('J'):
63              return wordnet.ADJ
64          elif tag.startswith('V'):
65              return wordnet.VERB
66          elif tag.startswith('N'):
67              return wordnet.NOUN
68          elif tag.startswith('R'):
69              return wordnet.ADV
70          else:
71              return wordnet.NOUN  # Default to noun if
     not found
72
73      def lemmatize_word(self, word):
74          pos = self.get_pos(word)
75          return self.wnl.lemmatize(word, pos)
```

```
76
77 text = Text("Hello world. This is a test sentence.")
78 sentences = text.split_into_sent()
79 print(sentences)
80
81 sent = Sent(sentences)
82 print(sent.get_sentence(1))
83
84 lex_entry = LexEntry(sent.get_sentence(1))
85 print(lex_entry.split_into_words())
86 print(lex_entry.get_word_len())
87 print(lex_entry.get_word_index("test"))
88 print(lex_entry.lemmatize_word("favoring"))
```