```python
import pytest
from classes_def import Text, Sent, LexEntry

def test_sentences():
    text = Text("Hello, this is an example text. It has multiple sentences.")
    sentences = text.split_into_sent()
    sent = Sent(sentences)

    print("Sentences:")
    for i in range(len(sentences)):
        try:
            print(f"Sentence {i}: {sent.get_sentence(i)}")
        except IndexError as e:
            print(e)

    assert sentences == ["Hello, this is an example text.",
                         "It has multiple sentences."], f"Expected sentences did not match: {sentences}"


def test_lex_entries():
    text = Text("Hello, this is an example text. It has multiple sentences.")
    sentences = text.split_into_sent()
    sent_chunk = Sent(sentences)
    for sentence in sentences:
        lex_entry = LexEntry(sentence)
        words = lex_entry.split_into_words()
        print(f"\nWords in '{sentence}':")
        for word in words:
            print(f"Word: {word}, Length: {len(word)}, Index: {lex_entry.get_word_index(word)}")

    first_sentence_lex = LexEntry(sentences[0])
    assert first_sentence_lex.split_into_words() == ["hello", "this", "is", "an", "example",

"text"], f"Expected words did not match: {
```

```
first_sentence_lex.split_into_words()}"

    second_sentence_lex = LexEntry(sentences[1])
    assert second_sentence_lex.split_into_words() == ["it", "has", "multiple",
 "sentences"], f"Expected words did not match: {second_sentence_lex.split_into_words()}"

def test_lemma():
    text = Text("Hello, this is an example text. It has multiple sentences.")
    sentences = text.split_into_sent()
    sent = Sent(sentences)

    for sentence in sentences:
        lex_entry = LexEntry(sentence)
        words = lex_entry.split_into_words()

        for word in words:
            pos = lex_entry.get_pos(word)
            lemma = lex_entry.lemmatize_word(word)

            # Print or assert as needed for testing
            print(f"Word: {word}, POS: {pos}, Lemma: {lemma}")

            # Example assertion (adjust as needed based on your expected outcomes)
            # This example assumes that 'example' should be lemmatized to 'example'
            if word == 'example':
                assert lemma == 'example', f"Expected 'example', but got '{lemma}'"
            elif word == 'has':
                assert lemma == 'have', f"Expected 'have', but got '{lemma}'"

if __name__ == "__main__":
    pytest.main()
```