Lab Number Three: Subarray Operators

Consider the following version of the LDA generative process, that records the words in each document as well as which topic produced which word:

```
import numpy as np

# there are 2000 words in the corpus
alpha = np.full (2000, .1)

# there are 100 topics
beta = np.full (100, .1)

# this gets us the probabilty of each word happening in each of the 100
topics
wordsInTopic = np.random.dirichlet (alpha, 100)

# produced [doc, topic, word] gives us the number of times that the given
word was
# produced by the given topic in the given doc
produced = np.zeros ((50, 100, 2000))

# generate each doc
for doc in range (0, 50):
        #
        # get the topic probabilities for this doc
        topicsInDoc = np.random.dirichlet (beta)
        #
        # assign each of the 2000 words in this doc to a topic
        wordsToTopic = np.random.multinomial (2000, topicsInDoc)
        #
        # and generate each of the 2000 words
        for topic in range (0, 100):
                produced[doc, topic] = np.random.multinomial
(wordsToTopic[topic], wordsInTopic[topic])
```

As described in the comments, `produced [doc, topic, word]` gives the number of times that the given word was produced by the given topic in the given doc. You need to complete the five tasks where we have not given an answer, and then show your answers in order to get checked off:

1. Write a line of code that computes the number of words produced by topic 17 in document 18. (the answer: `produced[18,17,:].sum ()`)
2. Write a line of code that computes the number of words produced by topic 17 thru 45 in document 18.
3. Write a line of code that computes the number of words in the entire corpus.
4. Write a line of code that computes the number of words in the entire corpus produced by topic 17 .
5. Write a line of code that computes the number of words in the entire corpus produced by topic 17 or topic 23. (the answer: `produced[:,np.array([17,23]),:].sum ()`)

6. Write a line of code that computes the number of words in the entire corpus produced by even numbered topics. (the answer: `produced[:,np.arange(0,100,2),:].sum ()`)
7. Write a line of code that computes the number of each word produced by topic 15.
8. Write a line of code that computes the topic responsible for the most instances of each word in the corpus.
9. Write a line of code that for each topic, computes the max number of occurrences (summed over all documents) of any word that it was responsible for. (nasty! One answer, though it is possible to compute up with a simpler answer: `produced[:,np.arange(0,100,1),produced.sum (0).argmax (1)].sum(0)`)