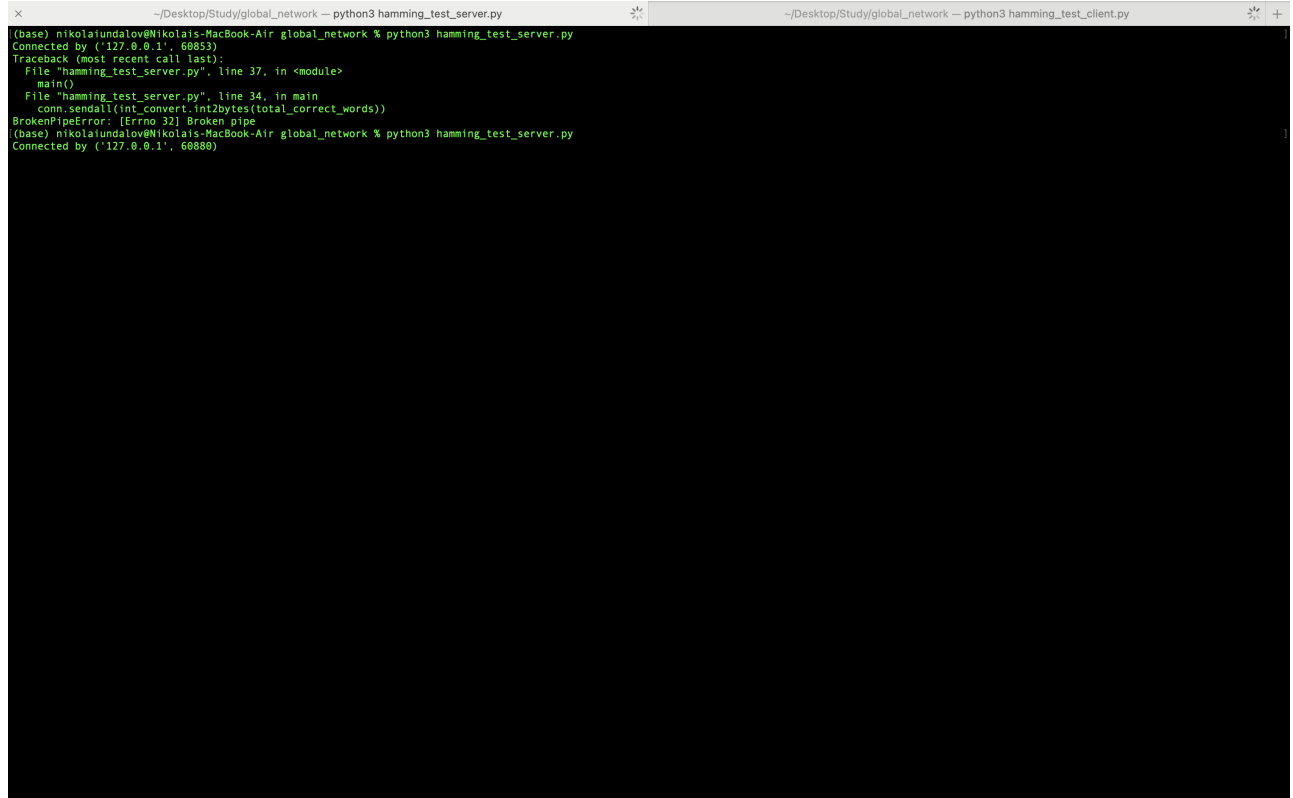


# Сокеты и код Хемминга

github url : [https://github.com/koly6868/misis\\_2021\\_global\\_networks](https://github.com/koly6868/misis_2021_global_networks)

- Запуск сервера.



```
(base) nikolaiundalov@NikolaIs-MacBook-Air: global_network % python3 hamming_test_server.py
Connected by ('127.0.0.1', 60853)
Traceback (most recent call last):
  File "hamming_test_server.py", line 37, in <module>
    main()
  File "hamming_test_server.py", line 34, in main
    conn.sendall(int_convert.int2bytes(total_correct_words))
BrokenPipeError: [Errno 32] Broken pipe
(base) nikolaiundalov@NikolaIs-MacBook-Air: global_network % python3 hamming_test_server.py
Connected by ('127.0.0.1', 60880)
```

- Запуск клиента.



```
(base) nikolaiundalov@NikolaIs-MacBook-Air: global_network % python3 hamming_test_client.py
for exit enter "exit" for start - "start"

```

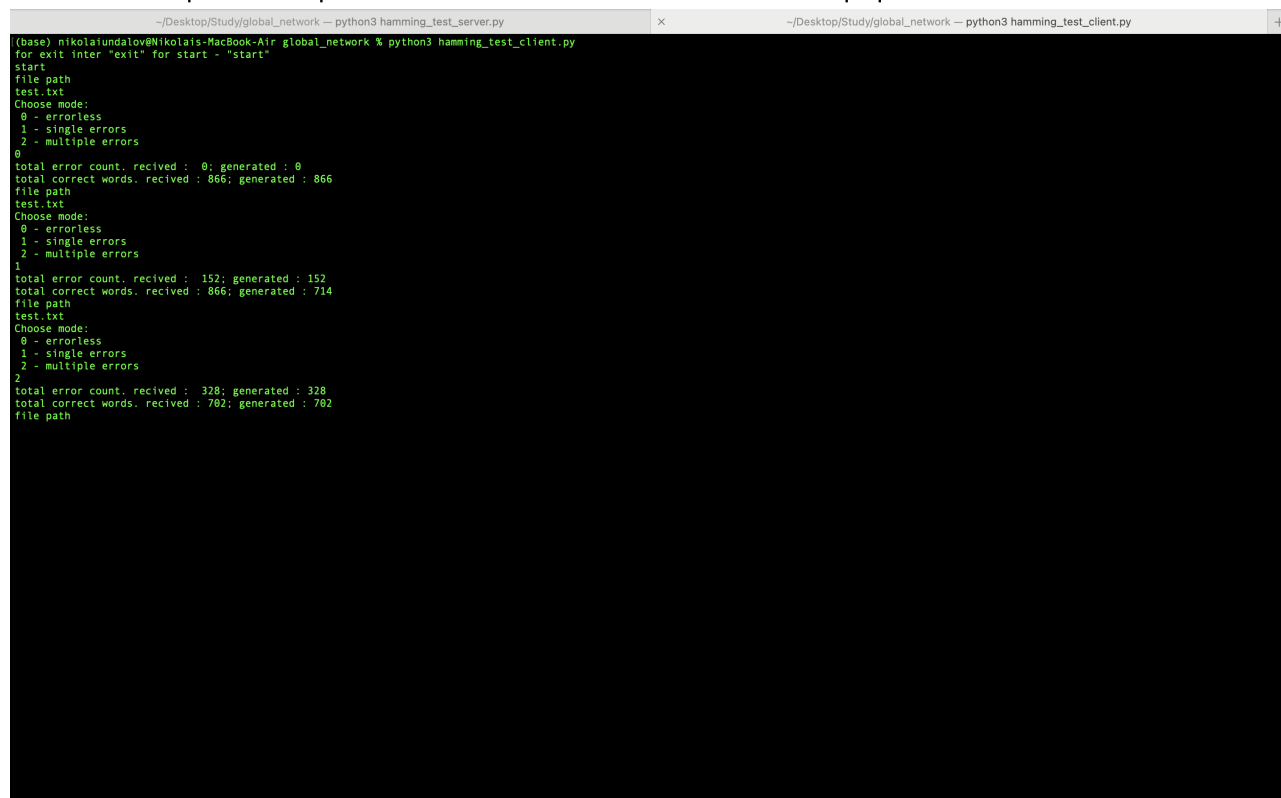
```
(base) nikolaiundalov@NikolaIs-MacBook-Air: global_network % python3 hamming_test_server.py
Connected by ('127.0.0.1', 60880)
```

- Пишем "start" и указываем имя файла для считывания текста.



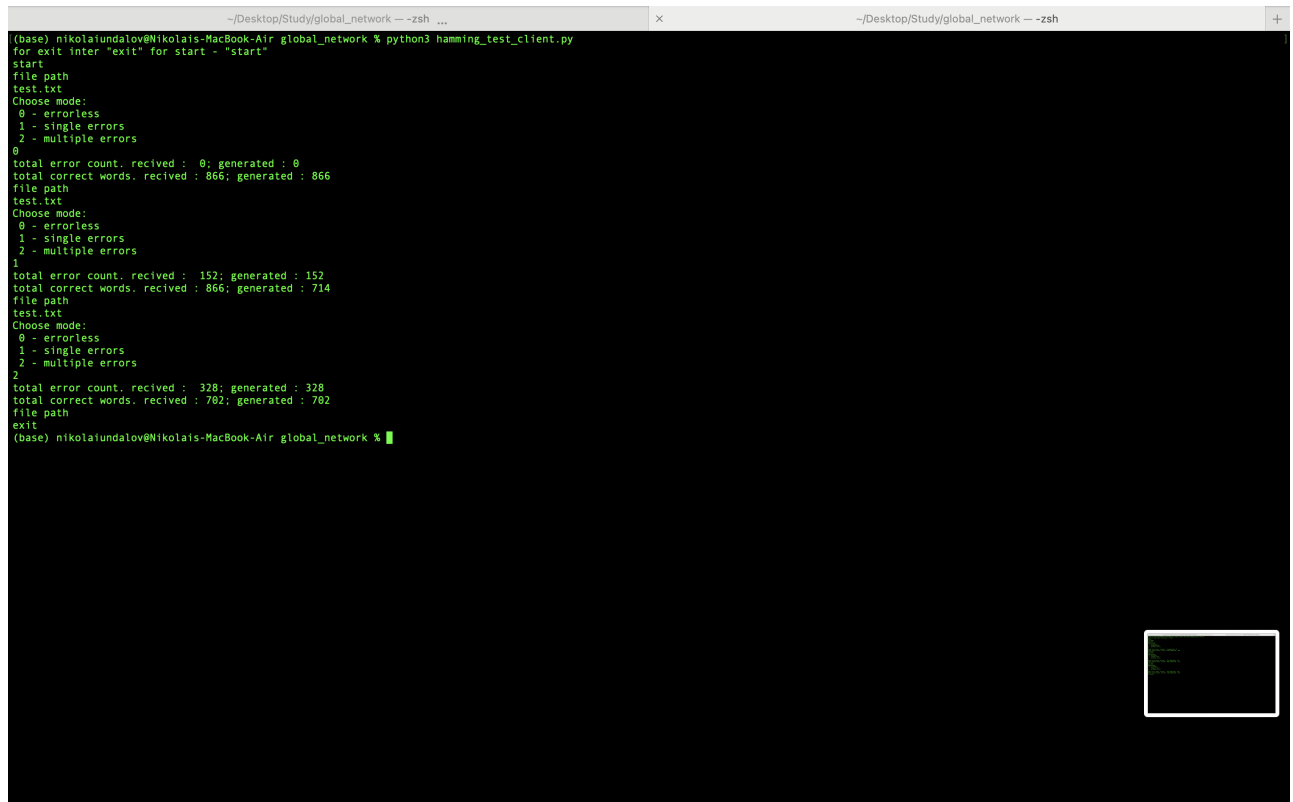
```
(base) nikolaiundalov@Nikolais-MacBook-Air: global_network % python3 hamming_test_client.py
for exit inter "exit" for start - "start"
start
file path
```

- Указываем режим отправки (режим внесения ошибок) и ждем ответа. На экран выводится сообщение о количестве ошибок сгенерированных и распознанных, а также количество правильно распознанных слов и количество сгенерированных без ошибок слов.



```
(base) nikolaiundalov@Nikolais-MacBook-Air: global_network % python3 hamming_test_client.py
for exit inter "exit" for start - "start"
start
file path
test.txt
Choose mode:
0 - errorless
1 - single errors
2 - multiple errors
0
total error count. received : 0; generated : 0
total correct words. received : 866; generated : 866
file path
test.txt
Choose mode:
0 - errorless
1 - single errors
2 - multiple errors
1
total error count. received : 152; generated : 152
total correct words. received : 866; generated : 714
file path
test.txt
Choose mode:
0 - errorless
1 - single errors
2 - multiple errors
2
total error count. received : 328; generated : 328
total correct words. received : 702; generated : 702
file path
```

- Используем разные режимы генерации ошибок.

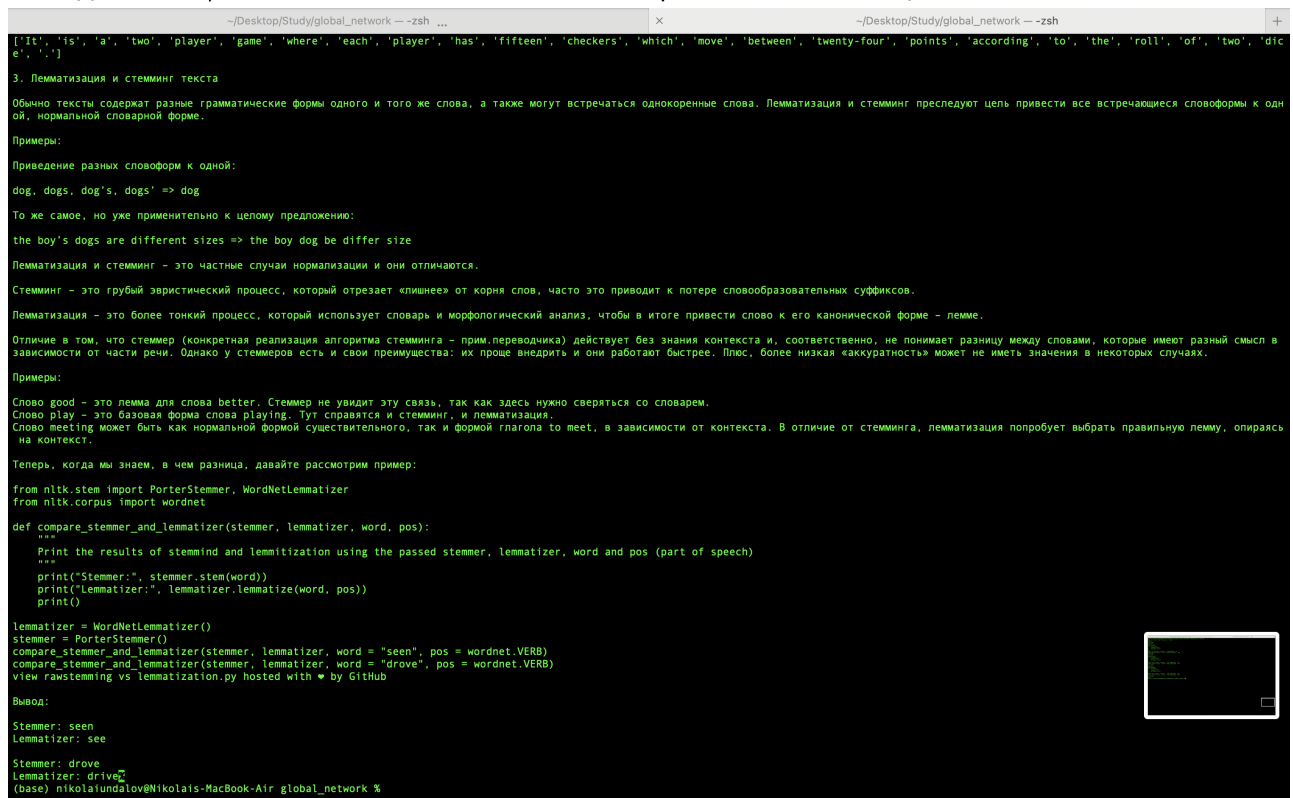


```

~/Desktop/Study/global_network -- -zsh ... x ~/Desktop/Study/global_network -- -zsh +
(base) nikolaiundalov@Nikola's-MacBook-Air: global_network % python3 hamming_test_client.py
for exit inter "exit" for start - "start"
start
file path
test.txt
Choose mode:
0 - errorless
1 - single errors
2 - multiple errors
0
total error count. received : 0; generated : 0
total correct words. received : 866; generated : 866
file path
test.txt
Choose mode:
0 - errorless
1 - single errors
2 - multiple errors
1
total error count. received : 152; generated : 152
total correct words. received : 866; generated : 714
file path
test.txt
Choose mode:
0 - errorless
1 - single errors
2 - multiple errors
2
total error count. received : 328; generated : 328
total correct words. received : 702; generated : 702
file path
test.txt
exit
(base) nikolaiundalov@Nikola's-MacBook-Air: global_network %

```

- Выводим текст, использованный в качестве отправляемого сообщения.



```

~/Desktop/Study/global_network -- -zsh ... x ~/Desktop/Study/global_network -- -zsh +
["It", "is", "a", "two", "player", "game", "where", "each", "player", "has", "fifteen", "checkers", "which", "move", "between", "twenty-four", "points", "according", "to", "the", "roll", "of", "two", "dice", "."]

3. Лемматизация и стемминг текста

Обычно тексты содержат разные грамматические формы одного и того же слова, а также могут встречаться однокоренные слова. Лемматизация и стемминг преследуют цель привести все встречающиеся словоформы к одной, нормальной словарной форме.

Примеры:

Приведение разных словоформ к одной:
dog, dogs, dog's, dogs' => dog

То же самое, но уже применительно к целому предложению:
the boy's dogs are different sizes => the boy dog be differ size

Лемматизация и стемминг – это частные случаи нормализации и они отличаются.

Стемминг – это грубый эвристический процесс, который отрезает «лишнее» от корня слов, часто это приводит к потере словообразовательных суффиксов.

Лемматизация – это более тонкий процесс, который использует словарь и морфологический анализ, чтобы в итоге привести слово к его канонической форме – лемме.

Отличие в том, что стеммер (конкретная реализация алгоритма стемминга – прим.переводчика) действует без знания контекста и, соответственно, не понимает разницу между словами, которые имеют разный смысл в зависимости от части речи. Однако у стеммеров есть и свои преимущества: их проще внедрить и они работают быстрее. Плюс, более низкая «какуратность» может не иметь значения в некоторых случаях.

Примеры:

Слово good – это лемма для слова better. Стеммер не увидит эту связь, так как здесь нужно сверяться со словарем.
Слово play – это базовая форма слова playing. Тут справятся и стемминг, и лемматизация.
Слово meeting может быть как нормальной формой существительного, так и формой глагола to meet, в зависимости от контекста. В отличие от стемминга, лемматизация попытается выбрать правильную лемму, опираясь на контекст.

Теперь, когда мы знаем, в чем разница, давайте рассмотрим пример:

from nltk.stem import PorterStemmer, WordNetLemmatizer
from nltk.corpus import wordnet

def compare_stemmer_and_lemmatizer(stemmer, lemmatizer, word, pos):
    """
    Print the results of stemming and lemmatization using the passed stemmer, lemmatizer, word and pos (part of speech)
    """
    print("Stemmer:", stemmer.stem(word))
    print("Lemmatizer:", lemmatizer.lemmatize(word, pos))
    print()

lemmatizer = WordNetLemmatizer()
stemmer = PorterStemmer()
compare_stemmer_and_lemmatizer(stemmer, lemmatizer, word = "seen", pos = wordnet.VERB)
compare_stemmer_and_lemmatizer(stemmer, lemmatizer, word = "drove", pos = wordnet.VERB)
view rawstemming vs lemmatization.py hosted with ❤ by GitHub

Вывод:
Stemmer: seen
Lemmatizer: see

Stemmer: drove
Lemmatizer: drive
(base) nikolaiundalov@Nikola's-MacBook-Air: global_network %

```