Проект этап 1. Предварительная обработка текстового корпуса

В **обработке естественного языка (NLP)** большая часть работы начинается с получения и предварительной обработки корпуса текстов. Слово **«корпус»** поначалу может показаться пугающим, но на самом деле оно просто относится к --набору текстовых данных--. Его содержание может быть связано тематически или основываться на определенных языковых явлениях. Корпус обычно имеет какую-то аннотацию, содержащую дополнительную информацию о тексте.

Для большинства лингвистических задач корпус должен быть обработан, прежде чем мы сможем получить доступ ко всей важной информации. Одной из самых основных операций является **токенизация**.

В этом проекте корпус хранится в виде файла `.txt` с кодировкой `UTF-8`. При чтении текстового файла в Python вы можете указать кодировку документа следующим образом:

```
f = open("corpus.txt", "r", encoding="utf-8")
```

Описание

В этом проекте мы будем использовать корпус, содержащий весь сценарий *«Игры престолов»*. Поскольку корпус будет использоваться для «обучения» вероятностной модели, которая будет предсказывать следующее слово в цепочке слов, сгенерированный текст будет напоминать стиль и словарный запас исходного материала. Естественность сгенерированного текста зависит от данных. Чем больше корпус, тем лучше результаты. Корпус, который мы будем использовать в этом проекте, состоит примерно из **300 000 токенов**. Это не идеально, но достаточно, чтобы получить интересные результаты.

После того, как вы закончите этот проект, вы можете использовать любой корпус, с которым хотите поэкспериментировать, с разными стилями и длинами — вы даже можете скомпилировать свой собственный корпус и использовать его. Но пока просто придерживайтес корпуса, предоставленного для этого проекта.

Задачи

Чтобы подготовить корпус для использования в этом проекте, нам необходимо сделать следующие важные шаги:

1. Откройте и прочитайте корпус из предоставленного файла `corpus.txt`. Имя файла должно быть указано пользователем. Обратите внимание, что файл написан в кодировке `UTF-8` и должен находиться в той же папке, что и ваш скрипт Python.

2. **Разбейте корпус на отдельные слова**. Чтобы создать марковскую модель, мы используем простейшую форму токенизации: токены разделяются пробелами, такими как пробелы, символы табуляции и символы новой строки. Знаки препинания следует оставить нетронутыми, так как позже они будут играть важную роль в определении того, где должно заканчиваться предложение.
3. **Получите и распечатайте следующую информацию о корпусе** в разделе выходных данных под названием **Статистика корпуса**:

— количество всех токенов; — количество всех уникальных токенов, то есть количество токенов без повторения. Каждое из вышеперечисленных должно быть в новой строке. 4. Возьмите целое число в качестве пользовательского ввода и напечатайте токен с соответствующим индексом. Повторяйте этот процесс до тех пор, пока не будет введена строка выхода **exit**. Кроме того, убедитесь, что входной индекс на самом деле **является целым числом**, попадающим в диапазон корпуса. Если это не так, **распечатайте сообщение об ошибке и запросите новый ввод**. Сообщения об ошибках должны содержать типы ошибок (Type Error, Index Error, Value Error и т.д.).

Каждый токен должен выводиться с новой строки.

Пример

Символ «больше чем», за которым следует пробел **(>)**, представляет **ввод пользователя**. Вывод программы должен выглядеть так. Обратите внимание, что это всего лишь пример: вы можете получить совершенно другие результаты.

```
> corpus.txt
Статистика корпуса текста
Всего токенов: 32434234
Уникальные токены: 433242

> 0
What
> 4
They're
> 5
savages.
> 32
like
> 42
ever
> 65
dead
> 256
the
> 532
are
> 756
king,
> 943287563823572346
Index Error. Пожалуйста, введите целое число, которое находится в
диапазоне корпуса.
> six
```

```
Type Error. Пожалуйста введите число.
> -1
North!
> exit
```