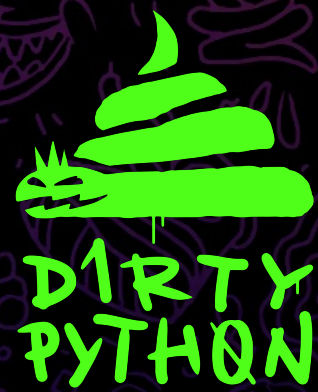
DIRTY PYTHON 1.0

STONE 17TH PRESENTS

ЛЕКЦИЯ 8



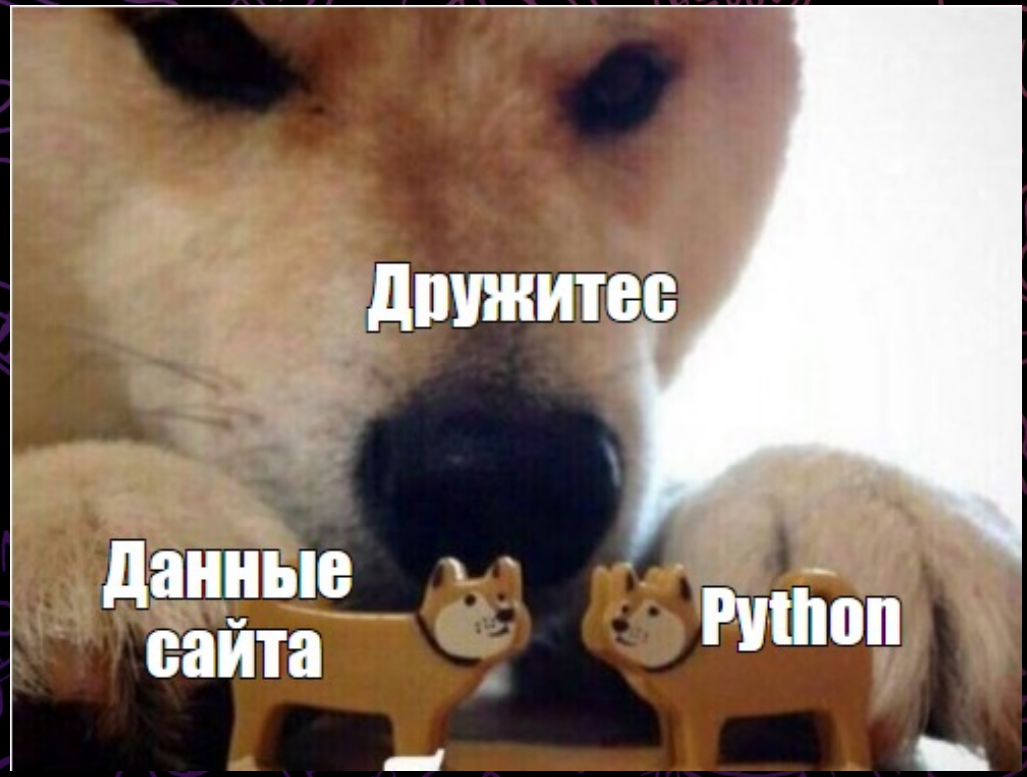
ЧТО У НАС СЕГОДНЯ?

Парсинг

Что за зверь?

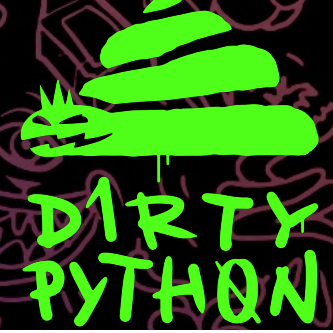
Как его ловить?

Парсинг многочлена подручными способами

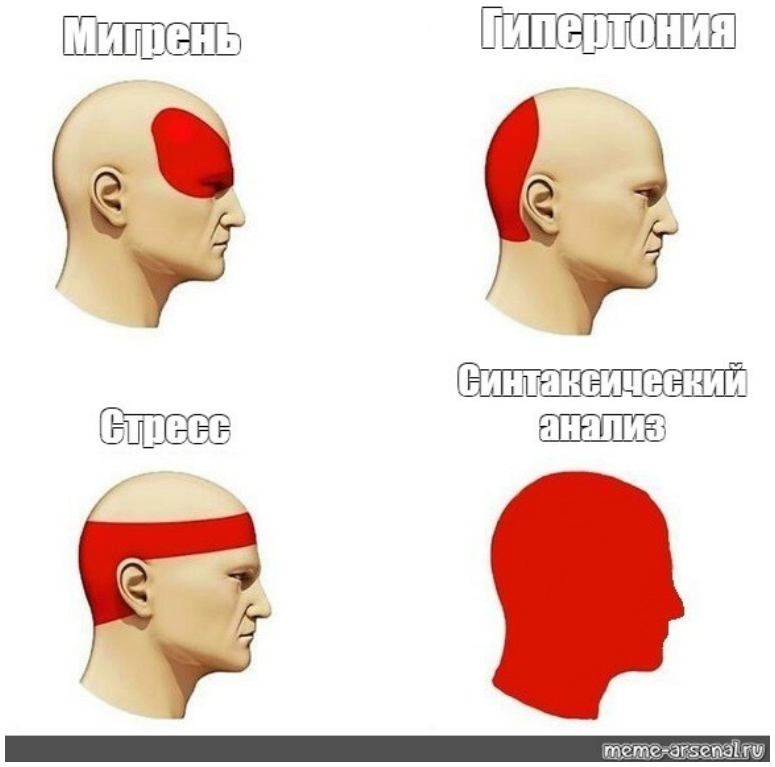


D1RTY PYTHON

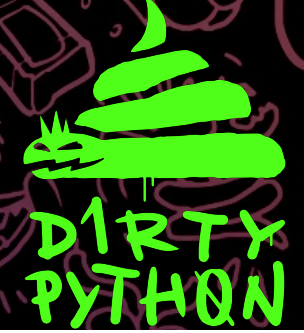
ЧТО ТАКОЕ ПАРСИНГ



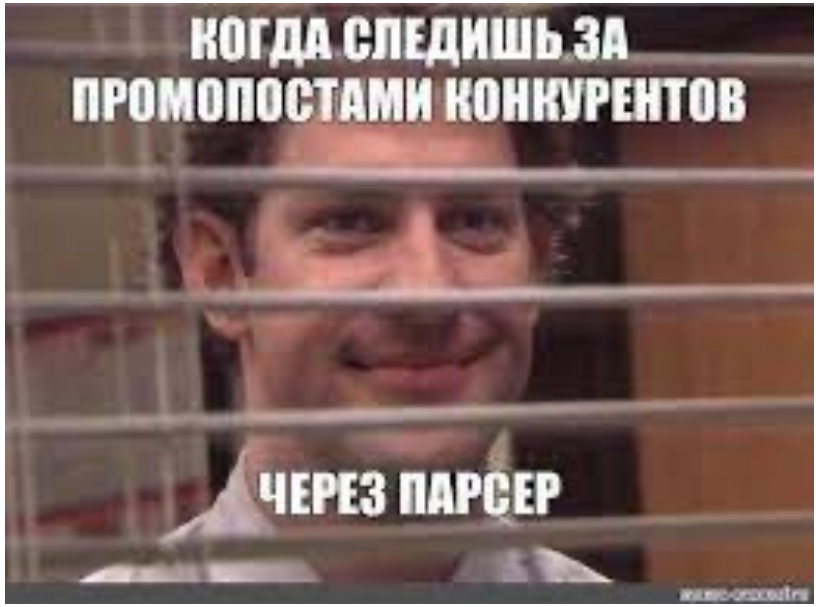
- Это жаргонизм, обозначающий синтаксический анализ: да, вы прямо сейчас занимаетесь парсингом – буквы складываете в слова, обрабатываете нейрончиками и получаете смыслы
- Вот что пишет википедия
- Синтаксический анализ (или разбор, жарг. парсинг ← англ. parsing) в лингвистике и информатике — процесс сопоставления линейной последовательности лексем (слов, токенов) естественного или формального языка с его формальной грамматикой. Результатом обычно является дерево разбора (синтаксическое дерево). Обычно применяется совместно с лексическим анализом.
- Синтаксический анализатор (жарг. парсер ← англ. parser) — это программа или часть программы, выполняющая синтаксический анализ.
- А еще у меня приятель так кота назвал – Парсер)



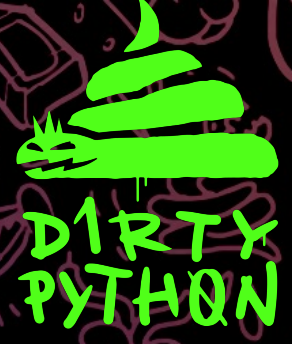
ДЛЯ ЧЕГО ОН НУЖЕН?



- В первую очередь (и большинство под парсингом имеет в виду именно это) - сбор и обработка информации с сайтов: проверка на уникальность, заполнение однотипных каталогов, анализ информации, организация спам-рассылки не обходится без парсинга…
- Но парсить можно не только сайты! Фактически это обработка любой информации, которую можно свести к тому или иному шаблону.
- Мы с вами попробуем попарсить многочлен (простигоспади…)



ЧТО НАМ ПОНАДОБИТЬСЯ?



- Разумеется, для профессионального парсинга имеются специальные библиотеки - Scrapy, Requests, BeautifulSoup и другие (https://vc.ru/services/249048-luchshie-instrumenty-dlya-samostoyatelnogo-parsinga-veb-saytov-ischerpyvayushchiy-spisok)
- Но мы попробуем обойтись, тем что умеем, а значит нам надо вспомнить строковые методы – поскольку работать мы будем с этим типом данных, и методы словаря – поскольку в словаре удобно хранить то, что мы с вами из строки извлечем

СТРОКОВЫЕ МЕТОДЫ

- Метод replace() работает так, как следует из его названия — изменяя подстроку на нужную: сначала пишем что меняем, потом на что меняем
- Метод split() позволит разделить вам строку – по умолчанию по пробелу, но можно поставить и любой другой разделить. И да, можно через запятую указать на сколько частей вы ходите поделить
- ‘one two three four five’.split(‘ ’, 2) -> [‘one’, ‘two’, ‘three four five’]
- Метод rsplit() сделает это же в обратном направлении

СТРОКОВЫЕ МЕТОДЫ

- Метод str.strip([chars]) вернет копию строки str с удаленными начальными и конечными символами chars. Другими словами, **обрежет строку str с обоих концов**. Аргумент chars - это строка, указывающая набор удаляемых символов. Если аргумент chars не задан или None, то по умолчанию удалит пробелы с обоих концов строки.
- Методы rstrip() и lstrip() - удаляют только с права и слева, соответсвенно
- Могут пригодится методы на проверку содержимого: .isdigit(), .isalpha(), .endswith(), .startswith() и др.

СТРОКОВЫЕ МЕТОДЫ

- Для генерации текста необходимо построить предложения и фразы из словаря слов. Этот процесс обратный разделению строки. Python позволяет нам использовать встроенный строковый метод join() для объединения слов обратно в предложение
- И еще целая куча всяких методов, которые могут пригодится)

МЕТОДЫ СЛОВАРЯ

- **dict.clear**() - очищает словарь.
- **dict.copy**() - возвращает копию словаря.
- classmethod **dict.fromkeys**(seq[, value]) - создает словарь с ключами из seq и значением value (по умолчанию None).
- **dict.get**(key[, default]) - возвращает значение ключа, но если его нет, не бросает исключение, а возвращает default (по умолчанию None).
- **dict.items**() - возвращает пары (ключ, значение).
- **dict.keys**() - возвращает ключи в словаре.
- **dict.pop**(key[, default]) - удаляет ключ и возвращает значение. Если ключа нет, возвращает default (по умолчанию бросает исключение).
- **dict.popitem**() - удаляет и возвращает пару (ключ, значение). Если словарь пуст, бросает исключение KeyError. Помните, что словари неупорядочены.
- **dict.setdefault**(key[, default]) - возвращает значение ключа, но если его нет, не бросает исключение, а создает ключ со значением default (по умолчанию None).
- **dict.update**([other]) - обновляет словарь, добавляя пары (ключ, значение) из other. Существующие ключи перезаписываются. Возвращает None (не новый словарь!).
- **dict.values**() - возвращает значения в словаре

ПАРСИНГ МНОГОЧЛЕНА

- Что такое многочлен? Это выражение типа

a*x^n + b*x^n-1 +… + k*x + j = 0

Нам нужно составить программу, которая будет

1) Составлять многочлен по заданной максимальной степени со случайными коэффициентами и вывести в виде строки
2) Сложить два таких многочлена и получить из него один. Например

3x^2 + 4x – 5 = 0

8x^3 + 13x^2 – x = 0 → 8x^3 + 16x^2 + 3x - 5 = 0

ТОНКОСТИ

- Никто не пишет часть многочлена, если коэффициент равен 0
- Никто не пишет x^1 и x^0
- Помните, что коэффициенты могут быть и отрицательными!

И ЧТО ДАЛЬШЕ?

- ПАРСИМ ДО УПОРА!
- ЕСЛИ С МНОГОЧЛЕНАМИ РАЗОБРАЛИСЬ ПОПРОБУЙТЕ СДЕЛАТЬ 'ЭЛЕКТРОННЫЙ ЖУРНАЛ', КОТОРЫЙ БУДЕТ ВКЛЮЧАТЬ 3-4 ПРЕДМЕТА И ДАННЫЕ ОБ ОЦЕНКАХ 3-4 УЧЕНИКОВ: ПО ФАМИЛИИ ВЫВЕСТИ ВЕДОМОСТЬ СО СРЕДНЕЙ ОЦЕНКОЙ ПО ВСЕМ ПРЕДМЕТАМ ИЛИ ПО ПРЕДМЕТУ ВЫВЕСТИ СРЕДНЮЮ УСПЕВАЕМОСТЬ УЧЕНИКОВ... НА ФАНТАЗИЮ, КОРОЧЕ