# Разведочный анализ данных.

## 1. Постановка задачи

Необходимо:

1. Провести разведочный анализ данных с целью ответа на вопрос: "С каким из поставщиков стали следует заключить договор?";
2. Написать программу, выполняющую разведочный анализ;
3. Оформить результаты и выводы в виде презентации, используя средства LaTex и Beamer.

## 2. Описание задачи

После оглушительного успеха в освобождении Астапора, Миэрина и Юнкая от власти работорговцев Дейенерис Бурерожденная открыла себе доступ к Летнему морю, а следовательно – путь в Вестерос.

Для ведения войны с Семью Королевствами нужно оружие, а для оружия нужна сталь. Нет никаких сомнений в кузнечном искусстве Безупречных, однако поставщики стали не столь надежны.

Два основных поставщика стали - это Westeros Inc. и Harpy Co. На протяжении нескольких месяцев мы закупаем сталь у обеих компаний, и каждая из них предлагает ощутимую скидку при заключении эксклюзивного договора на поставку.

Советник королевы Тирион Ланнистер знает о твоем умении принимать взвешенные рациональные решения и просит помощи в объективном решении вопроса о том, с какой из компаний следует заключить эксклюзивный договор на поставку стали.

У Тириона есть записи о производстве мечей каждым из кузнецов-безупречных, а также данные о количестве сломанных мечей в каждый из месяцев ведения боевых действий.

# 3. Исходные данные

Дан CSV-файл с данными о производстве оружия и количестве единиц сломанного оружия за каждый месяц каждым из кузнецов. Есть два основных поставщика стали - Westeros Inc. и Harpy  Co. Все кузнецы обладают высоким мастерством и производят оружие одинаково, поэтому качество их работы зависит исключительно от материала.
Исходный файл содержит в себе:

- “unsullen.id” – номер кузнеца, проводившего работу;
- “production.date” – месяц производства;
- “report.date” – месяц отчета;
- “produced” – количество мечей, произведенных за соответствующий месяц;
- “defects” – количество сломанных мечей;
- “supplier” – соответсвующий поставщик стали.

# 4. Решение

Разведочный анализ данных – это анализ основных свойств данных, нахождение в них общих закономерностей, распределений и аномалий, построение начальных моделей, зачастую с использованием визуализации.
Для поставленной задачи мы построим графики, отражающие:

1. Общее количество произведенного и сломавшегося оружия каждой из компании за 6 месяцев сотрудничества, относительную частоту появления дефекта (доля сломанных мечей);
2. Доли сломанных мечей **в** i-ый месяц эксплуатации;
3. Доли сломанных мечей **на** i-ый месяц эксплуатации;
4. Доли сломанных мечей на следующий месяц после производства;

Стоит заметить, что во всех пунктах, кроме первого, мы работает с долями, а не с фактическими значениям, так как в рамках задачи сравнения они будут показательнее из-за разного объема поставок.

# 5. Описание программы

Сгруппировав исходные данные по полю 'supplier' и просуммировав с помощью метода sum() по 'produced' и 'defect', нашли общее количество произведенного и сломавшегося оружия. Соответственно, отношение второго значения к первому будет давать относительную частоту появления дефекта.

Чтобы найти долю сломанных мечей в i-й месяц эксплуатации, опишем вспомогательную функцию `get_part_data(data)`, которая будет возвращать следующие массивы: `defects_month` – число сломанных мячей после (i + 1)-го месяца эксплуатации, `prod_month` – число произведенных мечей в (i+1) месяце, `part_broke` – доля сломанных мечей в i-й месяц. Первый два массива вычисляются очевидно. Подробнее опишем последний.

Пусть у нас уже есть массивы: `defects_month` = [d1, .., d6] и `prod_month` = [p1, .., p6]. Чтобы найти долю сломанных мечей за 1 месяц нужно поделить d1 на сумму элементов `prod_month`, то есть на общее количество произведенного оружия. Чтобы найти долю сломанных мечей за 2 месяц, мы d2 должны поделить на сумму элементов `prod_month` без учета последнего значения, так как у нас нет данных о количестве дефектов в 8 месяце, аналогично вычисляются оставшиеся элементы: `part_broke[i]` = `defects_month[i] / prod_month[:6 - i].sum()`. Таким образом, мы построили статистику поломки продукции после каждого месяца эксплуатации, то есть показали с какой скоростью мечи ломаются.

Затем построим график, отражающий долю сломанных мечей на каждый месяц. Благодаря нему мы определим, сколько мечей осталось пригодных для использования в каждый месяц. Эта информация будет полезна заказчику, так как для него важно, чтобы во время военных действий было как можно больше целого оружия. Для этого используем все ту же функцию `get_part_data(data)`, только после просуммируем массив `part_broke`.

Далее мы решили проверить доли сломанных мечей на следующий месяц после производства для поиска аномалии. Вдруг какой-нибудь из поставщиков сменил источник, из которого получал сталь, что привело к резкому повышению поломок и ухудшению качества его продукции. Или есть какая-то другая причина, которая так могла повлиять на ситуацию. Для этого описали вспомогательную функцию `get_anomaly(data)`, которая сгруппировала по месяцам данные о поломках и производстве и нашла отношение.

По результатам каждого пункта на экран выводим гистограммы или графики для наглядной визуализации.

# 6. Используемые библиотеки и основные функции

В данном задании мы использовали:

- модуль **pandas** – библиотека для обработки и анализа данных; `read_csv`- считывание данных из CSV файла; метод loc-используется для доступа по строковой метке; `reset_index`- переиндексирование; drop - удаление идексов; groupby - группировка по одному или нескольким столбцам.

- модуль **numpy**- библиотека, добавляющая поддержку больших многомерных массивов и матриц, вместе с большой библиотекой высокоуровневых (и очень быстрых) математических функций для операций с этими массивами, например, zeros - создает массив из нулей; shape — кортеж натуральных чисел, показывающий длину массива по каждой оси; arange - создание массива, основанного на числовых диапазонах.

- модуль **matplotlib**-библиотека двумерной графики с помощью которой можно создавать высококачественные рисунки различных форматов; метод matplotlib.ticker позволяет изменять положение меток, которые задают шаг сетки; matplotlib inline указывает, что график необходимо построить все в той же оболочке Jupyter, но теперь он выводится как обычная картинка; matplotlib.pyplot-предназначен для построения интерактивных графиков и про-

стых случаев программной генерации графиков; plot.bar- позволяет отрисовывать гистограммы; subplot- отображение несколько независимых графиков в одном окне; метод `set_facecolor` - изменяет цвет фигуры; grid- отображает в виде таблицы гистограмму; legend- окно, позволяющая определить что соответствует определенному цвету линии.

**Работу выполнили Корнюхина Софья, Попова Диана, Ситникова Екатерина, 311 группа; совместная работа над кодом и README.**