# Направление «Python-разработчик», компания ЕВРАЗ

Добро пожаловать на виртуальную стажировку компании ЕВРАЗ!

Предлагаем тебе погрузиться в будни Python-разработчика в компании ЕВРАЗ — одном из крупнейших игроков в секторе Metals & Mining.

В рамках проекта перед тобой стоит задача по рефакторингу кода, то есть процесса переработки и улучшения, чтобы он стал более простым и понятным.

Будь готов, что на выполнение задания потребуется 1–2 часа твоего времени. Однако все зависит от уровня владения Python, а также наличия необходимого программного обеспечения (ПО) на компьютере.

Во время работы над проектом ты научишься:

1. Лучше разбираться в коде, а не просто писать что-то работающее.
2. Проектировать необходимые изменения, что является важной частью процесса разработки.

**Рекомендуемый тайминг:** 60–90 минут.

Желаем удачи!

# Задание. Разделяй и властвуй!

Коллектив IT-департамента холдинга ЕВРАЗ ждет твоего вовлечения в процессы Python-разработки. И в качестве первого задания айтишники ЕВРАЗа предлагают тебе усовершенствовать ранее написанный, но, увы, неидеальный код.

Утром ты обнаружил электронное письмо с постановкой задания от руководителя с пометкой «СРОЧНО/ВАЖНО».

Привет!

Сейчас наш департамент борется с громоздкими программами, написанными на Python. Надеемся, что ты поможешь нам с одним таким «динозавром» — кодом, который предсказывает расход известняка при приготовлении шихты[1] ☺

Стоит отметить, что рефакторинг[2] является важным навыком для любого разработчика. Есть несколько инструментов для анализа качества Python-кода. Например, pycodestyle — простая консольная утилита[3], которая проверяет то, насколько код удовлетворяет требованиям PEP8[4].

Как видно из таблицы 1 во вложении, предлагаемый код имеет серьезный список несоответствий правилам, описанным в PEP8. Поэтому твой старший коллега Алексей предлагает провести улучшение кода путем создания модульной структуры, когда несколько файлов связаны друг с другом с помощью импорта.

Теперь вместо одного кода у тебя должно быть несколько отдельных файлов:

- Во-первых, файл _main_.py — файл верхнего уровня, который запускается для работы программы.
- Затем несколько пользовательских модулей, а именно:
  - ✔ Файл data_examples.py, содержащий все примеры данных, включая **CHARGE_CHEMISTRY**, **LIMESTONE_CONSUMPTIONS**, **CHARGE_CONSUMPTIONS**, **COKE_CONSUMPTIONS**, **COKE_SIEVING**, **LIMESTONE_CAO**.
  - ✔ Файл pipeline.py, содержащий функции **predict_cao** и класс **MeanDummyRegressor** из исходного кода, но теперь в виде класса, потому что в файле верхнего уровня будут использованы инструкции from и import для подключения модулей.
  - ✔ Наконец, Алексей рекомендует выделить обработку данных в файл feature_engineering.py, где должна находится функция по поиску всех NaN-значений с помощью метода fillna, а также функция создания фичей (features в исходном файле).

Ждем обновленную программу.

Спасибо!

[1] Шихта — смесь различных компонентов, которая необходима для выплавки сплава. Загружается внутрь печного оборудования и в составе может содержать как чистые металлы, так и отходы производства.
[2] Рефакторинг (refactoring), или перепроектирование кода — процесс изменения внутренней структуры программы, не затрагивающий ее внешнего поведения и имеющий целью облегчить понимание ее работы.
[3] Утилита — вспомогательная компьютерная программа в составе общего программного обеспечения для выполнения специализированных типовых задач, связанных с работой оборудования и ОС.
[4] PEP8 (Python Enhancement Proposal) — документ, описывающий стандарты, которых следует придерживаться при написании кода на Python.

## Вложения

Таблица 1. Вывод замечаний при запуске pycodestyle

```
C:\User\Desktop\code_for_refactoring.py:5:1: E302 expected 2 blank lines, found 1
C:\User\Desktop\code_for_refactoring.py:11:15: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:12:15: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:13:15: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:14:14: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:28:16: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:29:14: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:77:15: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:78:14: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:126:15: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:127:14: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:174:80: E501 line too long (92 > 79 characters)
C:\User\Desktop\code_for_refactoring.py:175:6: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:175:80: E501 line too long (82 > 79 characters)
C:\User\Desktop\code_for_refactoring.py:177:80: E501 line too long (91 > 79 characters)
C:\User\Desktop\code_for_refactoring.py:178:5: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:182:9: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:190:39: E127 continuation line over-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:197:80: E501 line too long (85 > 79 characters)
C:\User\Desktop\code_for_refactoring.py:223:80: E501 line too long (80 > 79 characters)
C:\User\Desktop\code_for_refactoring.py:296:9: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:314:13: E131 continuation line unaligned for
hanging indent
C:\User\Desktop\code_for_refactoring.py:318:13: E131 continuation line unaligned for
hanging indent
C:\User\Desktop\code_for_refactoring.py:322:13: E131 continuation line unaligned for
hanging indent
C:\User\Desktop\code_for_refactoring.py:326:13: E131 continuation line unaligned for
hanging indent
C:\User\Desktop\code_for_refactoring.py:331:13: E131 continuation line unaligned for
hanging indent
C:\User\Desktop\code_for_refactoring.py:337:1: E305 expected 2 blank lines after class
or function definition, found 1
C:\User\Desktop\code_for_refactoring.py:338:80: E501 line too long (84 > 79 characters)
C:\User\Desktop\code_for_refactoring.py:339:17: E128 continuation line under-indented
for visual indent
C:\User\Desktop\code_for_refactoring.py:339:65: W292 no newline at end of file
```

## Полезные материалы

- Файл с кодом для рефакторинга.
- Статья о том, как установить Python: Скачать Python 3 - Установка Python 3 на Windows, Ubuntu, macOS (python-scripts.com).
- Статья о модулях в Python: Модули Python: Создание, импорт и совместное использование - pythobyte.com.
- Статьи о том, как проводить рефакторинг кода: 10 предпочтительных методов рефакторинга кода на Python / Хабр (habr.com), Упрощение кода приложений Python с помощью рефакторинга. Часть 2 - Еще один блог веб-разработчика (webdevblog.ru).

## Формат конечного результата

Папка с файлами: _main_.py, data_examples.py, pipeline.py, feature_engineering.py.

## Форма загрузки результата

Пожалуйста, загрузи решение в формате zip-архива, содержащего все необходимые файлы.

## Пример решения

У тебя будет возможность ознакомиться с примером решения задания после отправки своей версии.