1) Создайте интерактивный калькулятор.

Пользовательский ввод представляет собой формулу, состоящую из числа, оператора (+,-,\*,/,\*\*) и другого числа, разделенных пробелом (например, 1 + 1). Проверьте input по следующим правилам:

1. Если входные данные не состоят из 3 действительных элементов, вызовите InputFormulaError, которая является пользовательским исключением.

2. Если первый и третий элемент не являются действительными числами — выводится ошибка InputNumberError.

3. Если второй элемент не соответствует поддерживаемым операторам — выводится ошибка InputOperatorError.

Если ввод верен, выполните вычисление и выведите результат (в случае возникновения ошибки при вычислении — также выводим ее). Затем пользователю предлагается ввести новый ввод и так далее, пока пользователь не введет quit.

Взаимодействие может выглядеть так:

>>> 1 + 1

2.0

>>> 3,2 - 1,5

1.70

>>> quit

2) Создайте класс Validator который позволяет проводить проверку данных пользователя при регистрации передаваемых в виде кортежа (login, password, email)

В данном классе реализовать метод validate(user\_data), который позволяет проверить передаваемый кортеж по правилам:

login — не менее 6 символов

password — не менее 8 символов, буквы в верхнем и нижнем регистре, не менее одного специального символа (+-/\*! и т.д)

email — присутствует символ @, оканчивается . и 2 символами (.by)

Валидация каждого элемента в кортеже производится отдельным методом для каждого элемента (validate\_email, validate\_login, validate\_password) в которых в случае непрохождения валидации вызывается ошибка (InvalidPassword, InvalidLogin, InvalidEmail), при соответствии — возвращается значение True. В методе validate необходимо предусмотреть обработку этих ошибок и в случае их наличия — вызвать ошибку ValidationError.

Ошибки создать самостоятельно

например

validator = Validator()

validator.validate(user\_login, Some!Password, [mail@mail.com](mailto:mail@mail.com))  
# True

validator.validate(user, Some!Password, [mail@mail.com](mailto:mail@mail.com))

# ValidationError

3\*) Создайте класс, который имеет атрибут queue (допускается использование списка) который имеет метод add позволяющий добавлять в queue следующие объекты — целые числа, числа с плавающей запятой, строки. При этом в момент добавления происходит валидация элементов по следующим правилам:

1. Целые числа — должны делится на 8, состоять из не более чем 4 символов

2. Числа с запятой — не более 3 символов после запятой

3. Строки — длина не более 4 символов без дублирования символов

В результате работы метода add элементы прошедшие валидацию добавляются в queuе, элементы не прошедшие валидацию выводятся пользователю с сообщением о причине недобавления, например

q=Queue()

q.add(1, 16, 280, 80000, 2.51, 1.875, text, data, world)

InvalidIntDivision → 1 # не делится на 8

InvalidIntNumberCount → 80000 # больше 4 символов

InvalidFloat → 1.875 # больше 2 символов после запятой

InvalidTextLength → world # больше 4 символов

DuplicatesInText → data # повторяющиеся символы

q.queue

# [16,280,text]

!!! название ошибок можно выбрать самостоятельно !!!

Полезные ссылки

try/except: [https://realpython.com/python-with-statement/#the-try-finally-approach](https://realpython.com/python-with-statement/" \l "the-try-finally-approach)

Errors and exceptions: <https://docs.python.org/3/tutorial/errors.html>

Exceptions Hierarchy: [https://docs.python.org/3/library/exceptions.html#exception-hierarchy](https://docs.python.org/3/library/exceptions.html" \l "exception-hierarchy)

User-Defined exceptions: <https://www.programiz.com/python-programming/user-defined-exception>