Что сегодня сделаем

Библиотеку для получения курсов валют в 2 строки

In [ ]:

```
from libs.exchange import Rate
```

In [ ]:

```
Rate().usd()
```

In [ ]:

```
Rate('full').AZN()
```

Задача

Дана строка со значениями, которые разделены запятыми:

In [1]:

```
line = '2019-07-01,organic,4293'
```

Напишите функцию column_count, которая возвращает число столбцов в такой строке

In [3]:

```
len('2019-07-01,organic,4293'.split(','))
```

Out[3]:

```
3
```

In [9]:

```
def line_count(line, count_empty_lines=True, sep=','):
    if count_empty_lines:
        return len(line.split(sep))
    else:
        if line:
            return len(line.split(sep))
        else:
            return 0
```

In [10]:

```
line_count(line)
```

Out[10]:

```
3
```

In [11]:

```
line_count('2019-07-01')
```

Out[11]:

```
1
```

In [13]:

```
line_count('', count_empty_lines=False)
```

Out[13]:

```
0
```

Классы

Пример использования переменной в разных методах

In [14]:

```
class AnyName:
    def method_1(self):
        self.currency = 'usd'

    def method_2(self):
        print(self.currency)
```

In [15]:

```
a = AnyName()
```

In [19]:

```
from datetime import datetime
```

In [ ]:

```
datetime.strptime
```

In [16]:

```
a.method_1()
```

In [17]:

```
a.currency
```

Out[17]:

```
'usd'
```

In [ ]:

```
b.
```

In [22]:

```
b = AnyName()
```

In [23]:

```
b.method_2()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-23-59f1720a6a7b> in <module>
----> 1 b.method_2()

<ipython-input-14-9c2240d258f9> in method_2(self)
      4
      5     def method_2(self):
----> 6         print(self.currency)

AttributeError: 'AnyName' object has no attribute 'currency'
```

In [20]:

```
a.currency
```

Out[20]:

```
'usd'
```

In [21]:

```
a.method_2()
```

```
usd
```

Метод init

In [24]:

```
# метод __init__ выполняется при вызове класса

class Rate:
    def __init__(self):
        self.format = 'value'
        print(self.format)
```

In [25]:

```
r = Rate()
```

```
value
```

In [26]:

```
r.format
```

Out[26]:

```
'value'
```

Класс для курсов валют

In [27]:

```
class Rate:
    def __init__(self):
        self.format = 'value'

    def show_current_format(self):
        return self.format
```

In [28]:

```
r = Rate()
```

In [29]:

```
r.show_current_format()
```

Out[29]:

```
'value'
```

Пример инициализации со значением переменной

In [30]:

```
class Rate:
    def __init__(self, format_):
        self.format = format_

    def show_current_format(self):
        return self.format
```

In [ ]:

```
r = Rate(format_='value')
```

In [ ]:

```
r.show_current_format()
```

Или сразу со значением по умолчанию

In [31]:

```
class Rate:
    def __init__(self, format_='value'):
        self.format = format_

    def show_current_format(self):
        return self.format
```

In [32]:

```
r = Rate()
r.show_current_format()
```

Out[32]:

```
'value'
```

In [33]:

```
r_full = Rate(format_='full')
r_full.show_current_format()
```

Out[33]:

```
'full'
```

In [34]:

```
r.format
```

Out[34]:

```
'value'
```

In [35]:

```
# значения переменных класса можно менять

r.format = 'full_123'
r.show_current_format()
```

Out[35]:

```
'full_123'
```

Задание

Создайте класс сотрудника Employee. При инициализации класса задается имя сотрудника name и его текущая зарплата salary. Напишите следующие методы:

1. Метод up, который увеличивает зарплату сотрудника на 100
2. Метод print, который выводит на экран текущую зарплату сотрудника в формате "Сотрудник Иван, зарплата 100"

In [39]:

```
class Employee:
    def __init__(self, name, salary):
        self.name = name
        self.salary = salary

    def up(self):
        self.salary += 100

    def print_(self):
        print('Сотрудник {}, зарплата {}'.format(self.name, self.salary))
```

In [40]:

```
ivan = Employee(name='Иван', salary=50)
```

In [41]:

```
ivan.salary
```

Out[41]:

```
50
```

In [54]:

```
ivan.up()
ivan.print_()
```

```
Сотрудник Иван, зарплата 1350
```

In [55]:

```
elena = Employee(name='Елена', salary=300)
```

In [62]:

```
elena.up()
elena.print_()
```

```
Сотрудник Елена, зарплата 1000
```

Полная версия класса

In [75]:

```
import requests
```

In [76]:

```
class Rate:
    def __init__(self, format_='value'):
        self.format = format_

    def exchange_rates(self):
        """
        Возвращает ответ сервиса с информацией о валютах в виде:

        {
            'AMD': {
                'CharCode': 'AMD',
                'ID': 'R01060',
                'Name': 'Армянских драмов',
                'Nominal': 100,
                'NumCode': '051',
                'Previous': 14.103,
                'Value': 14.0879
                },
            ...
        }
        """
        self.r = requests.get('https://www.cbr-xml-daily.ru/daily_json.js')
        return self.r.json()['Valute']

    def make_format(self, currency):
        """
        Возвращает информацию о валюте currency в двух вариантах:
        - полная информация о валюте при self.format = 'full':
        Rate('full').make_format('EUR')
        {
            'CharCode': 'EUR',
            'ID': 'R01239',
            'Name': 'Евро',
            'Nominal': 1,
            'NumCode': '978',
            'Previous': 79.6765,
            'Value': 79.4966
        }

        Rate('value').make_format('EUR')
        79.4966
        """
        response = self.exchange_rates()

        if currency in response:
            if self.format == 'full':
                return response[currency]

            if self.format == 'value':
                return response[currency]['Value']

        return 'Error'

    def eur(self):
        """Возвращает курс евро на сегодня в формате self.format"""
        return self.make_format('EUR')

    def usd(self):
        """Возвращает курс доллара на сегодня в формате self.format"""
```

```
        return self.make_format('USD')

    def brl(self):
        """Возвращает курс бразильского реала на сегодня в формате self.format"""
        return self.make_format('BRL')
```

In [77]:

```
r = Rate(format_='full')
```

In [78]:

```
r.exchange_rates()
```

Out[78]:

```
{'AUD': {'ID': 'R01010',
  'NumCode': '036',
  'CharCode': 'AUD',
  'Nominal': 1,
  'Name': 'Австралийский доллар',
  'Value': 47.7337,
  'Previous': 47.8415},
 'AZN': {'ID': 'R01020A',
  'NumCode': '944',
  'CharCode': 'AZN',
  'Nominal': 1,
  'Name': 'Азербайджанский манат',
  'Value': 40.9491,
  'Previous': 40.9759},
 'GBP': {'ID': 'R01035',
  'NumCode': '826',
  'CharCode': 'GBP',
  'Nominal': 1,
```

In [79]:

```
r.usd()
```

Out[79]:

```
{'ID': 'R01235',
 'NumCode': '840',
 'CharCode': 'USD',
 'Nominal': 1,
 'Name': 'Доллар США',
 'Value': 69.5725,
 'Previous': 69.618}
```

In [80]:

```
r = Rate()
```

In [81]:

```
r.usd()
```

Out[81]:

```
69.5725
```

In [ ]:

```
r.exchange_rates()
```

Документация необходима почти всем методам

In [82]:

```
?r.exchange_rates
```

Наследование

Разработчикам финансового департамента помимо курса надо работать с кодами валют. Как сохранить сохранить разработку класса Rate у нас, а полезные функции передать финансистам?

In [83]:

```
class CurrencyCodes(Rate):
    def __init__(self):
        super().__init__(format_='full')
```

Теперь классу CurrencyCodes доступны все методы класса Rate. Можем продолжать разработку в нужном направлении.

In [84]:

```
cc = CurrencyCodes()
```

In [85]:

```
CurrencyCodes().usd()
```

Out[85]:

```
{'ID': 'R01235',
 'NumCode': '840',
 'CharCode': 'USD',
 'Nominal': 1,
 'Name': 'Доллар США',
 'Value': 69.5725,
 'Previous': 69.618}
```

Допишем в класс что-нибудь свое новенькое

In [86]:

```
class CurrencyCodes(Rate):
    def __init__(self):
        super().__init__(format_='full')

    def currency_id(self, currency):
        """Получение идентификатора валюты"""
        return self.make_format(currency)['ID']
```

In [87]:

```
currency = CurrencyCodes()
```

In [88]:

```
currency.currency_id('USD')
```

Out[88]:

```
'R01235'
```

Система повышения сотрудников

In [89]:

```
class Employee:
    def __init__(self, name, seniority):
        self.name = name
        self.seniority = seniority

        self.grade = 1

    def grade_up(self):
        """Повышает уровень сотрудника"""
        self.grade += 1

    def publish_grade(self):
        """Публикация результатов аккредитации сотрудников"""
        print(self.name, self.grade)

    def check_if_it_is_time_for_upgrade(self):
        pass
```

In [90]:

```
class Developer(Employee):
    def __init__(self, name, seniority):
        super().__init__(name, seniority)

    def check_if_it_is_time_for_upgrade(self):
        # для каждой аккредитации увеличиваем счетчик на 1
        # пока считаем, что все разработчики проходят аккредитацию
        self.seniority += 1

        # условие повышения сотрудника из презентации
        if self.seniority % 5 == 0:
            self.grade_up()

        # публикация результатов
        return self.publish_grade()
```

In [91]:

```
# проверяем как работает система повышения сотрудников на примере отдела разработки

# разработчик Александр только что пришел в компанию
alex = Developer('Александр', 0)
```

In [92]:

```
for i in range(20):
    alex.check_if_it_is_time_for_upgrade()
```

```
Александр 1
Александр 1
Александр 1
Александр 1
Александр 2
Александр 2
Александр 2
Александр 2
Александр 2
Александр 3
Александр 3
Александр 3
Александр 3
Александр 3
Александр 4
Александр 4
Александр 4
Александр 4
Александр 4
Александр 5
```

Импорт классов и функций

In [93]:

```
from libs.exchange import my_sum
```

In [94]:

```
my_sum(1, 2)
```

Out[94]:

```
3
```

In [95]:

```
from libs.exchange import Rate
```

In [96]:

```
Rate().AZN()
```

Out[96]:

```
40.9491
```

In [ ]:

```
# такой способ импорта крайне не рекомендуется
from libs.exchange import *
```

Если библиотека лежит в произвольной папке

In [97]:

```
import sys
sys.path
```

Out[97]:

```
['/Users/kbashevoy/Desktop/Нетология/Занятия/Записи/06. Классы в python',
 '/Users/kbashevoy/anaconda3/lib/python37.zip',
 '/Users/kbashevoy/anaconda3/lib/python3.7',
 '/Users/kbashevoy/anaconda3/lib/python3.7/lib-dynload',
 '',
 '/Users/kbashevoy/.local/lib/python3.7/site-packages',
 '/Users/kbashevoy/anaconda3/lib/python3.7/site-packages',
 '/Users/kbashevoy/anaconda3/lib/python3.7/site-packages/aeosa',
 '/Users/kbashevoy/anaconda3/lib/python3.7/site-packages/IPython/extension
s',
 '/Users/kbashevoy/.ipython',
 '/Users/kbashevoy/Desktop/Нетология/Занятия/Записи/06. Классы в python/lib
s',
 'адрес_папки_с_файлами',
 '/Users/kbashevoy/Desktop/Нетология/Занятия/Записи/06. Классы в python/lib
s']
```

In [98]:

```
# пример

import sys
sys.path.append('/Users/kbashevoy/Desktop/Нетология/Занятия/Записи/06. Классы в python/libs
```

In [99]:

```
from exchange import my_sum
```

In [100]:

```
my_sum(3, 3)
```

Out[100]:

```
6
```