#Задание 1

data = [

[13, 25, 23, 34],

[45, 32, 44, 47],

[12, 33, 23, 95],

[13, 53, 34, 35],

]

def diag(data):

diag\_val=0

for i in range(len(data)):

diag\_val+=data[i][i]

return diag\_val

diag(data)

#Задание 2

data = [1, '5', 'abc', 20, '2']

def square(data):

summ=0

for i in range(len(data)):

if str(data[i]).isdigit():

summ+=int(data[i])\*\*2

return summ

square(data)

#Задание 3

import requests

r = requests.get('https://www.cbr-xml-daily.ru/daily\_json.js')

valuta=r.json()['Valute']

spisokValut=[]

#Вычисление максимального курса

for key in valuta.keys():

spisokValut.append(valuta[key]['Value'])

max\_val=max(spisokValut)

#Вывод названия валюты, у которой максимальный курс

for key in valuta.keys():

if valuta[key]['Value']==max\_val:

print (valuta[key]['Name'])

#Задание 4

import requests

class Rate:

def \_\_init\_\_(self, format='full', diff='True'): #добавлен новый параметр diff

self.format = format

self.diff=diff

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

},

...

}

"""

r = requests.get('https://www.cbr-xml-daily.ru/daily\_json.js')

return 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'

#Добавлен новый метод

def make\_format\_name(self, currency):

"""" Возвращает название валюты"""

response = self.exchange\_rates()

if currency in response:

return response[currency]['Name']

return 'Error'

def eur(self):

"""Возвращает курс евро на сегодня в формате self.format"""

"""Если параметр self.diff='True', то передается разница курсов"""

if self.diff == 'True':

return self.make\_format('EUR')['Value']-self.make\_format('EUR')['Previous']

if self.diff == 'False':

return self.make\_format('EUR')['Value']

def usd(self):

"""Возвращает курс доллара на сегодня в формате self.format"""

"""Если параметр self.diff='True', то передается разница курсов"""

if self.diff == 'True':

return self.make\_format('USD')['Value']-self.make\_format('USD')['Previous']

if self.diff == 'False':

return self.make\_format('USD')['Value']

#Задание 5

def Fib(n):

summ=[1,2]

if n==1:return 1

else:

for i in range(n-2):

summ.append(summ[i]+summ[i+1])

return sum(summ)

print('Введите количество чисел')

Fib(int(input()))

#Задание 6

data = ['2018-01-01', 'yandex', 'cpc', 100]

def changeFormat(data):

slovar={}

i=0

if i==len(data)-1: return data[i]

if i<len(data):

slovar[data[i]]=changeFormat(data[i+1:])

return slovar

print(changeFormat(data))