Привет! Проверь как на аппер джуниор. Спасибо!

'''2. Выполнить функцию, которая принимает несколько параметров, описывающих данные

пользователя: имя, фамилия, год рождения, город проживания, email, телефон. Функция

должна принимать параметры как именованные аргументы. Осуществить вывод данных о

пользователе одной строкой.

'''

from typing import Optional

def print\_identity(surname: Optional[str] =None, name: Optional[str] =None, birth\_date: Optional[str] =None, place\_of\_living: Optional[str] =None, phone\_number: Optional[str] =None, email: Optional[str] =None)-> None:

print(

f'\nThe person name is {name} \nand his surname is {surname}, \nhe/she was born in {birth\_date}, \nfor now hi/she is living in {place\_of\_living}, \nhere is his/her contact information, \nphone number is {phone\_number} \nand email is {email}')

pass

print\_identity(name='Vova', surname='Petrov', birth\_date='01.10.1987', phone\_number='+4423459854',

place\_of\_living='London', email='tut@gmail.com')

def check\_for\_letters(input\_question: Optional[str] = None, repeat\_question: Optional[str] = None)-> str:

while True:

user\_input = str(input(f'\n{input\_question}'))

if user\_input.isalpha():

return user\_input

else:

print(f'\n{repeat\_question}')

continue

enter\_name = check\_for\_letters(input\_question='Enter the first name: ',

repeat\_question='You are entered not letters, try to enter first name again')

enter\_surname = check\_for\_letters(input\_question='Please enter customer surname: ',

repeat\_question='You are entered not letters, try to enter surname again')

enter\_place\_of\_living = check\_for\_letters(input\_question='Please enter customer place of living: ',

repeat\_question='You are entered not letters, try to enter place of living again')

from datetime import datetime

def check\_date(input\_question: Optional[str] =None, repeat\_question: Optional[str] =None)-> str:

while True:

birth\_date = str(input(f'\n{input\_question}'))

try:

datetime.strptime(birth\_date, '%d.%m.%Y')

return birth\_date

except (ValueError, TypeError):

print(f'\n{repeat\_question}')

continue

birth\_date = check\_date(input\_question='Please enter customer birth\_date in following format - dd.mm.yyyy: ',

repeat\_question='You are entered not date, please enter date in following format - dd.mm.yyyy: ')

from email\_validator import validate\_email, EmailNotValidError

def check\_email(input\_question: Optional[str] =None, repeat\_question: Optional[str] =None)-> str:

while True:

customer\_email = input(f'\n{input\_question}')

try:

email\_check = validate\_email(customer\_email, check\_deliverability=False)

return customer\_email

except EmailNotValidError:

print(f'\n{repeat\_question}')

customer\_email = check\_email(input\_question='Please enter customer email: ',

repeat\_question='you are entered not email, please enter email: ')

import re

def check\_phone\_number(input\_question: Optional[str] =None, repeat\_question: Optional[str] =None)-> str:

while True:

enter\_phone\_number = input(f'\n{input\_question}')

if re.fullmatch(r'\+?\d{7,15}', enter\_phone\_number) != None:

return enter\_phone\_number

else:

print(f'\n{repeat\_question}')

continue

pass

customer\_phone\_number = check\_phone\_number(input\_question='Please enter phone number in following format - +xxxxxxxxxx\n(+, country code, phone number with no dash, dots, spaces, slash): ', repeat\_question='You are entered not phone number, please enter phone number in following format - +xxxxxxxxxx: ')

print\_identity(surname=enter\_surname, name=enter\_name, birth\_date=birth\_date, place\_of\_living=enter\_place\_of\_living, phone\_number=customer\_phone\_number, email=customer\_email)