Техническое задание

Есть данные о разных автомобилях и спецтехнике. Данные в представленной таблице с характеристиками. Вся техника разделена на три вида: спецтехника, легковые и грузовые автомобили. Обратите внимание на то, что некоторые характеристики присущи только определенному виду техники.

Например, у легковых автомобилей есть характеристика «кол-во пассажирских мест», а у грузовых автомобилей - габариты кузова: «длина», «ширина» и «высота».

Тип (car\_type) Марка (марка) Кол-во пассажирских мест (Passenger\_seats\_count) Фото (photo\_file\_name) Кузов ДxШxВ, м (body\_whl) Грузоподъемность, Тонн (перевозка) Дополнительно (экстра) автомобиль Nissan xTtrail 4 f1.jpeg 2.5 грузовик Man f2.jpeg 8x3x2.5 20 автомобилей Mazda 6 4 f3.jpeg 2.5 spec\_machine Hitachi f4.jpeg 1.2 Легкая техника для уборки снега Вам необходимо создать свою иерархию классов для данных, которые используются в таблице. Классы должны называться CarBase (базовый класс для всех типов машин), Car (легковые автомобили), Truck (грузовые автомобили) и SpecMachine (спецтехника). Все объекты имеют обязательные атрибуты:

* car\_type, значение типа объекта и может принимать одно из значений: «car», «truck», «spec\_machine».
* photo\_file\_name, имя файла с изображением машины, допустимые названия файлов изображений с расширением из списка: «.jpg», «.jpeg», «.png», «.gif»
* марка, марка производителя машины
* переноска, грузоподъемность

В базовом классе CarBase нужно реализовать метод get\_photo\_file\_ext для расширения изображения файла. Расширение файла можно получить при помощи os.path.splitext.

Для грузового автомобиля необходимо в конструкторе класса определить атрибуты: длина\_тела, ширина\_тела, высота\_тела, соответствующие соответственно за габариты кузова - длина, ширина и высота. Габариты передаются в параметрах body\_whl (строка, в которой разделены латинской буквой «x»). Обратите внимание на то, что характеристики кузова должны быть вещественными числами и характеристики кузова могут быть не валидными (например, пустая строка). В таком случае всем атрибутам, отвечающим за габариты кузова, присваивается равное нулю.

Также для класса грузового автомобиля необходимо реализовать метод get\_body\_volume, возвращающий объем кузова.

В классе Автомобиль должен быть определен атрибут пассажирских мест (количество пассажирских мест), а в классе SpecMachine - экстра (дополнительное описание машины).

Полная информация о атрибутах приведена в таблице ниже, где 1 - означает, что атрибут обязателен для объекта, 0 - атрибут отсутствовать.

Автомобиль Грузовик SpecMachine car\_type 1 1 1 photo\_file\_name 1 1 1 марка 1 1 1 перевозит 1 1 1 пассажира\_мест\_count 1 0 0 body\_width 0 1 0 body\_height 0 1 0 body\_length 0 1 0 extra 0 0 1 Обратите внимание, что у каждого объекта из иерархии должен быть свой набор атрибутов и методов. Например, у класса легковой автомобиль не должно быть методом get\_body\_volume в отличие от класса грузового автомобиля. Имена атрибутов и методы должны совпадать с теми, что выше.

Далее вам необходимо реализовать функцию get\_car\_list. Файл содержит данные, аналогичные строкиам из таблицы. Вам необходимо прочитать этот файл построчно при помощи модуля стандартной библиотеки csv. Затем проанализировать строки на валидность и создать список объектов с автомобилями и специальной техникой. Функция должна возвращать список объектов.

Первая строка в исходном файле - это заголовок csv, который содержит имена колонок. Нужно пропустить первую программу из исходного файла. Обратите внимание на то, что в некоторых строках исходного файла, данные могут быть заполнены некорректно, например, отсутствовать обязательные поля или иметь не валидное значение. В таком случае нужно проигнорировать подобные строки и создать объекты. Строки с пустым или не валидным размером для body\_whl игнорироваться не должны. Вы можете использовать стандартный механизм обработки исключений в процессе чтения, валидации и создания объектов из строк csv-файла. Проверьте работу вашего кода с входным файлом.

Пример кода, демонстрирующего чтение csv файла:

import csv

with open(csv\_filename) as csv\_fd:

reader = csv.reader(csv\_fd, delimiter=';')

next(reader) # пропускаем заголовок

for row in reader:

print(row)

Ниже приведен шаблон кода для выполнения задания:

class CarBase:

def \_\_init\_\_(self, brand, photo\_file\_name, carrying):

pass

class Car(CarBase):

def \_\_init\_\_(self, brand, photo\_file\_name, carrying, passenger\_seats\_count):

pass

class Truck(CarBase):

def \_\_init\_\_(self, brand, photo\_file\_name, carrying, body\_whl):

pass

class SpecMachine(CarBase):

def \_\_init\_\_(self, brand, photo\_file\_name, carrying, extra):

pass

def get\_car\_list(csv\_filename):

car\_list = []

return car\_list

Несколько примеров работы:

>>> from solution import \*

>>> car = Car('Bugatti Veyron', 'bugatti.png', '0.312', '2')

>>> print(car.car\_type, car.brand, car.photo\_file\_name, car.carrying,

... car.passenger\_seats\_count, sep='\n')

car

Bugatti Veyron

bugatti.png

0.312

2

>>> truck = Truck('Nissan', 'nissan.jpeg', '1.5', '3.92x2.09x1.87')

>>> print(truck.car\_type, truck.brand, truck.photo\_file\_name, truck.body\_length,

... truck.body\_width, truck.body\_height, sep='\n')

truck

Nissan

nissan.jpeg

3.92

2.09

1.87

>>> spec\_machine = SpecMachine('Komatsu-D355', 'd355.jpg', '93', 'pipelayer specs')

>>> print(spec\_machine.car\_type, spec\_machine.brand, spec\_machine.carrying,

... spec\_machine.photo\_file\_name, spec\_machine.extra, sep='\n')

spec\_machine

Komatsu-D355

93.0

d355.jpg

pipelayer specs

>>> spec\_machine.get\_photo\_file\_ext()

'.jpg'

>>> cars = get\_car\_list('cars\_week3.csv')

>>> len(cars)

4

>>> for car in cars:

... print(type(car))

...

<class 'solution.Car'>

<class 'solution.Truck'>

<class 'solution.Truck'>

<class 'solution.Car'>

>>> cars[0].passenger\_seats\_count

4

>>> cars[1].get\_body\_volume()

60.0

>>>