**Практическая работа №3 Объекты и классы: абстрактные классы и интерфейсы, пакеты.**

**Задача**

Ваша программа должна быть организована по пакетам:

• Пакет vehicles для классов Vehicle, Car, ElectricCar, и

интерфейса ElectricVehicle.

• Пакет app для тестового класса TestCar.

Создайте абстрактный класс Vehicle, который будет представлять общие характеристики любого транспортного средства. Включите следующие поля:

model, license, color, year, ownerName, insuranceNumber, engineType.

Определите методы: геттеры и сеттеры для каждого поля, а также метод toString(), который возвращает строку с описанием транспортного средства.

Добавьте абстрактный метод vehicleType(), который будет возвращать тип

транспортного средства (например, "Car", "Electric Car"). Класс Car должен наследовать абстрактный класс Vehicle. В конструкторе задавайте тип двигателя как "Combustion". Реализуйте метод vehicleType(), который возвращает "Car".

Определите интерфейс ElectricVehicle, который будет описывать специфические методы для электромобилей. В интерфейсе должны быть следующие методы: getBatteryCapacity() и setBatteryCapacity().

Класс ElectricCar должен наследовать класс Car и реализовывать интерфейс

ElectricVehicle. Реализуйте методы интерфейса и добавьте поле batteryCapacity для хранения информации о емкости батареи. В конструкторе задайте тип двигателя как "Electric".

Требования к тестированию программы:

• Созданы экземпляры классов Car и ElectricCar.

• Изменение года выпуска и имени владельца.

• Изменение страхового номера.

• Получение информации о емкости батареи у электромобиля.

• Вывод информации о транспортных средствах в консоль с помощью

метода toString().

**Задача**

# Проектирование интерфейсов в классах Java #

Импортируйте файл bikeproject.zip в папку на вашем компьютере.

\*\*\*

###### Ссылки ######

[Интерфейсы](https://docs.oracle.com/javase/tutorial/java/concepts/interface.html "Описание из документаии")

[Методы интерфейса](https://docs.oracle.com/javase/tutorial/java/IandI/override.html)

[Использование \*super\*](https://docs.oracle.com/javase/tutorial/java/IandI/super.html)

\*\*\*

## Ответьте на следующие вопросы ##

\* Приведите пример примитивного типа данных, который используется для хранения полей в классе.

\* Приведите пример того, где происходит конкатенация строк.

\* Каковы имена объектов, созданных в этой программе?

\* Сколько конструкторов у каждого класса?

\* Наследование является частью этой программы. Определите супер- и подклассы из этой программы.

\* Горные и шоссейные велосипеды могут быть сконструированы либо с использованием значений по умолчанию (стандартный велосипед), либо настроены в соответствии с потребностями клиента. Используя следующую таблицу, определите примеры значений, назначенных одному из каждого типа стандартного велосипеда

## Модернизация программы ##

1. Создайте интерфейс в проекте велосипеда, который устанавливает название компании-производителя велосипедов как неизменяемое значение. Он также определяет методы, которые должны быть реализованы любым классом, использующим интерфейс.

2. Создайте интерфейс с именем \*\*MountainParts\*\*, который имеет константу с именем \*\*TERRAIN\*\*, которая будет хранить строковое значение «off\_road». Интерфейс определит два метода, которые принимают строковое аргументное имя newValue, и два метода, которые будут возвращать текущее значение поля экземпляра. Методы должны быть названы: \*\*getSuspension\*\*, \*\*setSuspension\*\*, \*\*getType\*\*, \*\*setType\*\*.

3. Создайте интерфейс \*\*RoadParts\*\*, который имеет константу с именем \*\*terrain\*\*, которая будет хранить строковое значение «\*\*track\_racing\*\*». Интерфейс определит два метода, которые принимают строковое аргументное имя newValue, и два метода, которые будут возвращать текущее значение поля экземпляра. Методы должны быть названы: \*\*getTyreWidth\*\*, \*\*setTyreWidth\*\*, \*\*getPostHeight\*\*, \*\*setPostHeight\*\*.

4. Используйте интерфейс \*\*BikeParts\*\* с классом \*\*Bike\*\*, добавляя любые необходимые нереализованные методы. Добавьте требуемый внутренний код для каждого из добавленных методов.

5. Используйте интерфейс \*\*MountainParts\*\* с классом \*\*MountainBike\*\*, добавив все необходимые нереализованные методы. Добавьте требуемый внутренний код для каждого из добавленных методов.

6. Используйте интерфейс \*\*RoadParts\*\* с классом \*\*RoadBike\*\*, добавив все необходимые нереализованные методы. Добавьте требуемый внутренний код для каждого из добавленных методов.

7. Запустите и протестируйте свою программу, она должна работать точно так же, как и раньше.

8. В нижней части класса драйвера обновите высоту столба для bike1 до 20 вместо 22.

9. Выведите значения bike1 на экран, чтобы подтвердить изменение.

10. Запустите и протестируйте свою программу.