Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе № 18.2**

Дисциплина: «Информатика»

Тема: Объектно-ориентированное программирование. Использование конструкторов

Вариант 11

Выполнил:

Студент группы РИС-20-1б

Ознобихин Елисей Андреевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

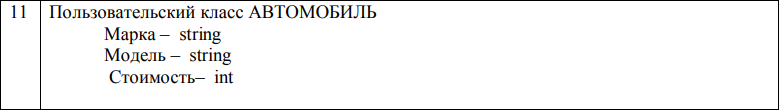
**Пермь, 2021**

**Цель работы**

1. Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.
2. Создание объектов с использованием конструктора.

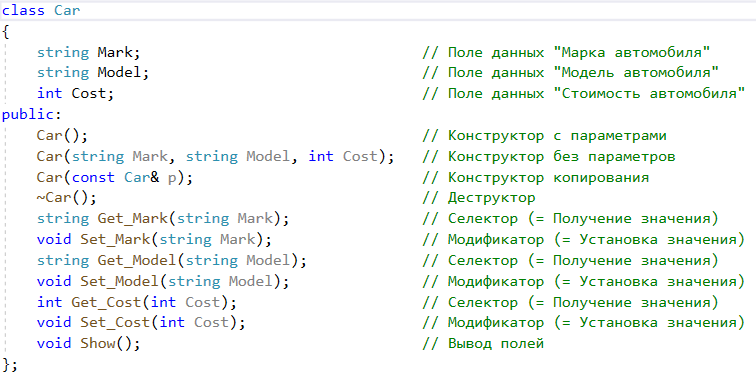
**Постановка задачи**

1. Определить пользовательский класс
2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.
3. Определить в классе деструктор.
4. Определить в классе компоненты-функции для просмотра и установки полей данных (селекторы и модификаторы).
5. Написать демонстрационную программу, в которой продемонстрировать все три случая вызова конструктора-копирования, вызов конструктора с параметрами и конструктора без параметров.

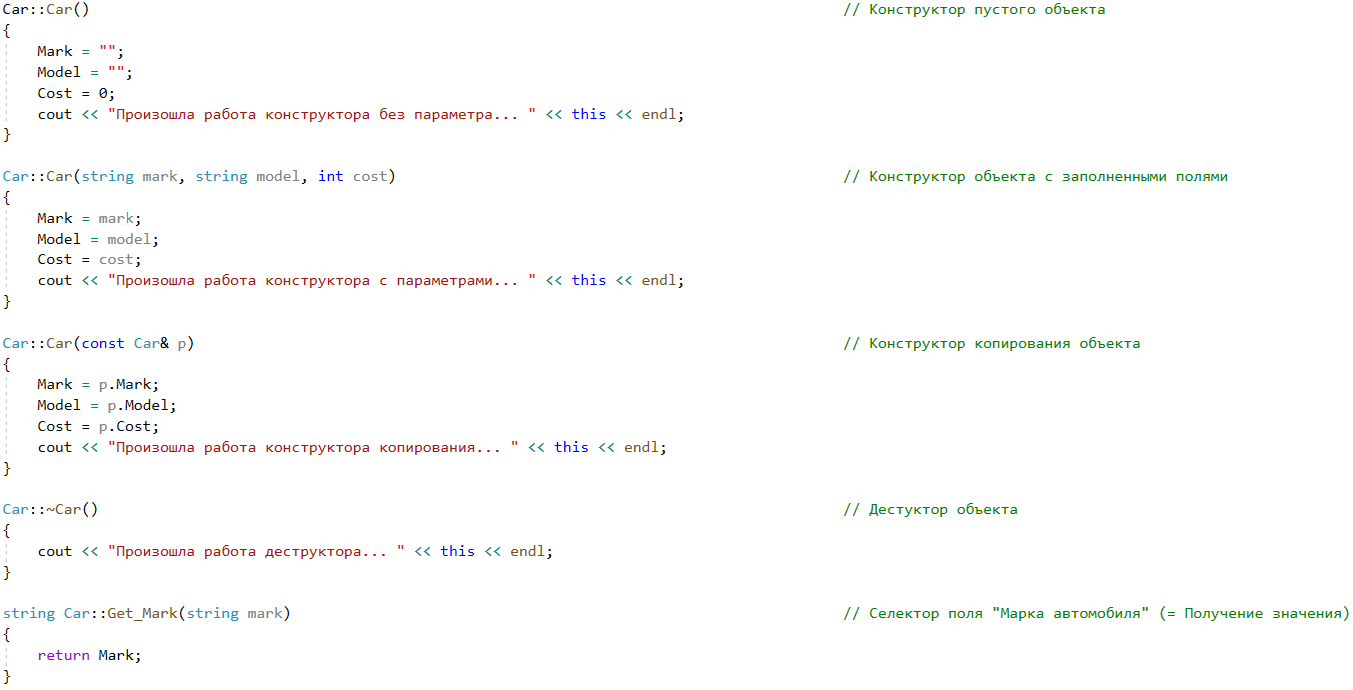


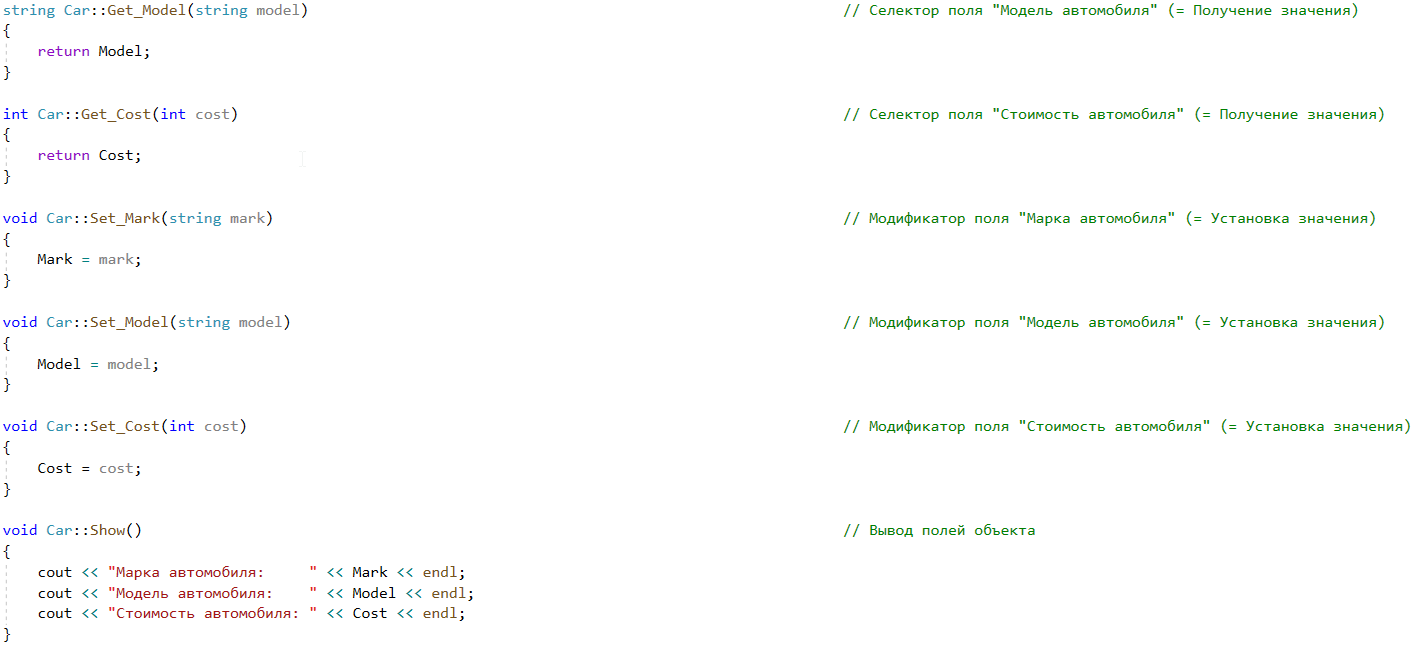
**Анализ задачи**

1. Для решения задачи необходимо:
   1. Реализовать определение класса Car.

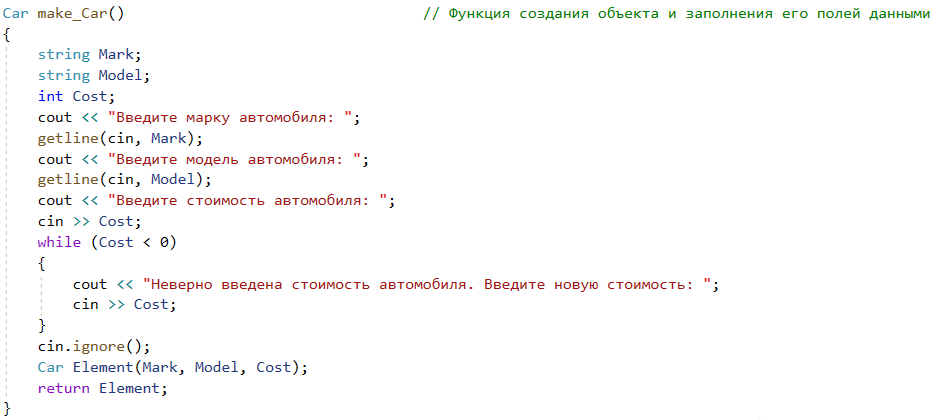


* 1. Реализовать определение конструкторов без параметра, с параметрами и копирования, деструктора, селекторов и модификаторов класса.





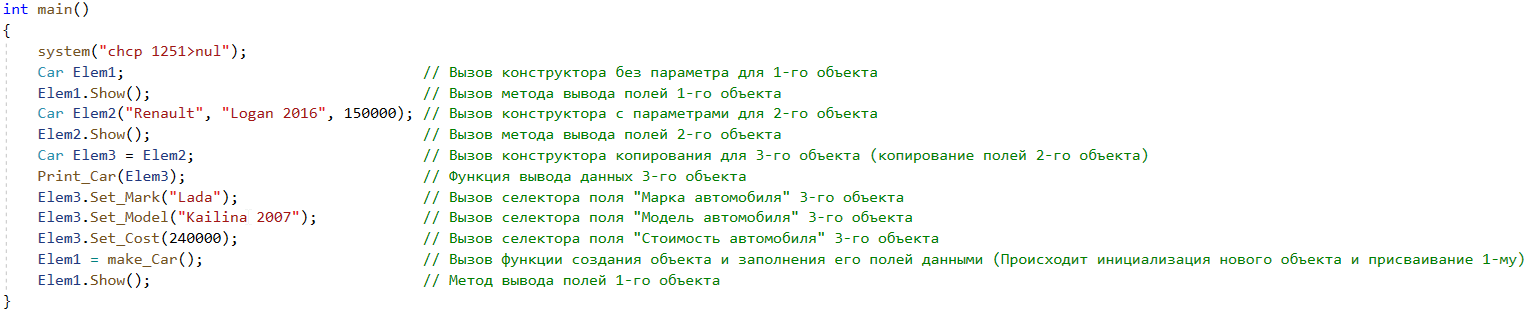
* 1. Разработать функцию make\_Car для выделения памяти под новые данные в атрибутах объекта и обращения к конструктору с параметрами.



* 1. Разработать функцию Print\_Car для обращения к методу вывода полей.



* 1. Реализовать применение этих функций в главной функции.



1. В ходе работы были использованы типы данных:
   1. Для конструктора без параметров Car класса Car не используются аргументы:

Сам конструктор заполняет атрибуты объекта класса “пустотами”

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_s5PMbyzEJ7.png

* 1. Для конструктора с параметрами Car класса Car используются следующие аргументы:
  2. Тип string: марка автомобиля.
  3. Тип string: модель автомобиля.
  4. Тип int: стоимость автомобиля.

Сам конструктор заполняет атрибуты объекта класса параметрами.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_jULFbKO2yg.png

* 1. Для конструктора копирования Car класса Car используются следующие аргументы:
  2. Адресация типа const Car: объект копирования.

Сам конструктор заполняет атрибуты объекта класса атрибутами объекта копирования.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_9jWomKcp2b.png

* 1. Для деструктора Car класса Car не используются аргументы:

Сам деструктор удаляет атрибуты объекта класса.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_3BjKJOrkSE.png

* 1. Для селектора Get\_Mark используются следующие аргументы:
  2. Тип string: марка автомобиля.

Сам селектор имеет тип string и возвращает значение поля “марка автомобиля”.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_GfSKNsYClt.png

* 1. Для селектора Get\_Model используются следующие аргументы:
  2. Тип string: модель автомобиля.

Сам селектор имеет тип string и возвращает значение поля “модель автомобиля”.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_iZqFz1Vi9K.png

* 1. Для селектора Get\_Cost используются следующие аргументы:

1. Тип int: стоимость автомобиля.

Сам селектор имеет тип int и возвращает значение поля “стоимость автомобиля”.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_1BFEwURhg5.png

* 1. Для модификатора Set\_Mark используются следующие аргументы:

1. Тип string: марка автомобиля.

Сам модификатор имеет тип void, поскольку при работе селектора не нужно возвращать значение.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_J1sVTngJ7u.png

* 1. Для модификатора Set\_Model используются следующие аргументы:

1. Тип string: модель автомобиля.

Сам модификатор имеет тип void, поскольку при работе селектора не нужно возвращать значение.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_mW0TyQDw8D.png

* 1. Для модификатора Set\_Cost используются следующие аргументы:

1. Тип int: стоимость автомобиля.

Сам модификатор имеет тип void, поскольку при работе селектора не нужно возвращать значение.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_rH17G0B1U9.png

* 1. Для метода Show не используются аргументы:

Сам метод имеет тип void, поскольку при работе селектора не нужно возвращать значение.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_ZcF5Hu8AO3.png

* 1. Для функции make\_Car не используются аргументы:

Сама функция имеет тип Car и возвращает объект с заполненными атрибутами.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_DmAyHQxwlb.png

* 1. Для функции Print\_Car используются следующие аргументы:
  2. Тип Car: объект для вывода атрибутов.

Сама функция имеет тип void, поскольку при работе селектора не нужно возвращать значение.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_HgJQwMpND1.png

1. Для решения задачи данные были представлены в следующем виде:
   1. Для работы с данными используются атрибуты класса.
2. Для операций ввода и вывода использовались следующие операторы и функции:
   1. Ввод данных реализован с помощью оператора cin и getline, используемых при реализации функции make\_Car, в главной функции и при реализации конструкторов и деструктора.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_VfRO6XHU7P.png

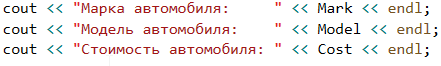
C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_tdlZP5kR83.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_VZb0ZOhzxO.png

* 1. Вывод данных реализован с помощью оператора cout, используемого при реализации метода Show и функций make\_Car и Print\_Car.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_77wMLbQvy8.png

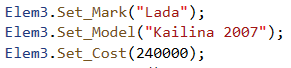
C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_3hvfV7DgOs.png



C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_vR0C5DIg2K.png

1. Поставленные задачи решены следующими действиями:
   1. Определение класса Car было реализовано в заголовочном файле Car.h, определение конструкторов, деструктора, селекторов и модификаторов класса – в файле Car.cpp. Основной блок программы и функция make\_Staff описаны в главном файле Лаба\_18\_2\_main.cpp.
   2. При работе с объектами класса Car обращение к конструктору без параметра аналогичен выделению памяти под переменную. Обращение к конструктору с параметрами аналогичен обращению к методу Init. Обращение к конструктору копирования аналогичен оператору присваивания. Обращение к селекторам и модификаторам аналогичен обращению к методам или к полям структуры.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_XLDMFDQznq.png



C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_Xm00gqhg5Z.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-03\devenv_MP1bovvgDC.png

**Блок-схема программы**

**Реализация задачи на языке С++**

**Заголовочный файл Car.h**

#pragma once

#include <iostream>

using namespace std;

class Car

{

string Mark;

string Model;

int Cost;

public:

Car();

Car(string Mark, string Model, int Cost);

Car(const Car& p);

~Car();

string Get\_Mark(string Mark);

void Set\_Mark(string Mark);

string Get\_Model(string Model);

void Set\_Model(string Model);

int Get\_Cost(int Cost);

void Set\_Cost(int Cost);

void Show();

};

**Файл с описанием методов класса Car.cpp**

#include <iostream>

#include <string>

#include "Car.h"

using namespace std;

Car::Car()

{

Mark = "";

Model = "";

Cost = 0;

cout << "Произошла работа конструктора без параметра... " << this << endl;

}

Car::Car(string mark, string model, int cost)

{

Mark = mark;

Model = model;

Cost = cost;

cout << "Произошла работа конструктора с параметрами... " << this << endl;

}

Car::Car(const Car& p)

{

Mark = p.Mark;

Model = p.Model;

Cost = p.Cost;

cout << "Произошла работа конструктора копирования... " << this << endl;

}

Car::~Car(){

cout << "Произошла работа деструктора... " << this << endl;

}

string Car::Get\_Mark(string mark)

{

return Mark;

}

string Car::Get\_Model(string model)

{

return Model;

}

int Car::Get\_Cost(int cost)

{

return Cost;

}

void Car::Set\_Mark(string mark)

{

Mark = mark;

}

void Car::Set\_Model(string model)

{

Model = model;

}

void Car::Set\_Cost(int cost)

{

Cost = cost;

}

void Car::Show()

{

cout << "Марка автомобиля: " << Mark << endl;

cout << "Модель автомобиля: " << Model << endl;

cout << "Стоимость автомобиля: " << Cost << endl;

}

**Файл с главной программой Лаба\_18\_2\_main.cpp**

#include <iostream>

#include <string>

#include "Car.h"

using namespace std;

Car make\_Car()

{

string Mark;

string Model;

int Cost;

cout << "Введите марку автомобиля: ";

getline(cin, Mark);

cout << "Введите модель автомобиля: ";

getline(cin, Model);

cout << "Введите стоимость автомобиля: ";

cin >> Cost;

while (Cost < 0)

{

cout << "Неверно введена стоимость автомобиля. Введите новую стоимость: ";

cin >> Cost;

}

cin.ignore();

Car Element(Mark, Model, Cost);

return Element;

}

void Print\_Car(Car Element)

{

Element.Show();

}

int main()

{

system("chcp 1251>nul");

Car Elem1;

Elem1.Show();

Car Elem2("Renault", "Logan 2016", 150000);

Elem2.Show();

Car Elem3 = Elem2;

Print\_Car(Elem3);

Elem3.Set\_Mark("Lada");

Elem3.Set\_Model("Kailina 2007");

Elem3.Set\_Cost(240000);

Elem1 = make\_Car();

Elem1.Show();

}

**Скриншоты результатов работы программы**

**Ответы на вопросы**

1. Для чего нужен конструктор?

Конструктор – один из методов инициализации объекта созданного класса. Конструктор имеет одинаковое имя с классом.

1. Сколько типов конструкторов существует в С++?

Существует три типа конструкторов:

* Конструктор без параметров, для создания “пустого” объекта
* Конструктор с параметрами, для создания объекта с заданными значениями
* Конструктор копирования, для создания объекта с атрибутами ранее созданного.

1. Для чего используется деструктор? В каких случаях деструктор описывается явно?

Деструктор вызывается, когда нужно удалить объект из памяти. Описывать деструктор явно необходимо в случае, когда объект содержит указатели на память, выделяемую автоматически (иначе поля-указатели не буду свободными).

1. Для чего используется конструктор без параметров? Конструктор с параметрами? Конструктор копирования?

Конструктор без параметров используется для создания “пустого” объекта.

Конструктор с параметрами используется для инициализации объекта требуемыми значениями.

Конструктор копирования используется для создания нового объекта путём копирования существующего.

1. В каких случаях вызывается конструктор копирования?

Конструктор копирования вызывается:

* При описании нового объекта с инициализацией другим объектом
* При передаче объекта в функцию по значению
* При возврате объекта из функции

1. Перечислить свойства конструкторов.

* Не возвращает значение, нельзя получить указатель на конструктор
* В классе может быть несколько конструкторов (нужна перегрузка)
* Тип аргументов в конструкторе любой, кроме класса
* Конструктор без параметров – конструктор по умолчанию
* Не указан какой-либо конструктор – компилятор создает конструктор по умолчанию автоматически
* Не наследуются
* Нельзя описывать с модификаторами const, virtual, static

1. Перечислить свойства деструкторов.

* Нет аргументов и не возвращает значение
* Не наследуется
* Нельзя описывать с модификаторами const, static
* Можно описывать с модификатором virtual

1. К каким атрибутам имеют доступ методы класса?

Методы класса имеют доступ ко всем элементам класса, независимо от спецификаторов доступа и порядка объявления.

1. Что представляет собой указатель this?

Указатель this – указатель на объект, используемый в каком-либо из методов класса. Используется в качестве возвращающегося значения.

1. Какая разница между методами, определенными внутри класса и вне класса?

Разница между методами, определенными внутри класса и вне класса состоит лишь в их описании.

1. Какое значение возвращает конструктор?

Любой тип конструктора не возвращает значения.

1. Какие методы создаются по умолчанию?

По умолчанию создаются конструкторы и деструктор.

1. Какое значение возвращает деструктор?

Деструктор не возвращает значение.

1. Дано описание класса

class Student

{

string name;

int group;

public:

Student(string, int);

Student(const Student&);

~Student();

};

Какой метод отсутствует в описании класса?

В данном случае отсутствует конструктор без параметров.

1. Какой метод будет вызван при выполнении следующих операторов:

Student\* s;

s = new Student;

В данном случае будет вызван конструктор без параметров.

1. Какой метод будет вызван при выполнении следующих операторов:

Student s(“Ivanov”, 20);

В данном случае будет вызван конструктор с параметрами.

1. Какие методы будут вызваны при выполнении следующих операторов:

Student s1(“Ivanov”, 20);

Student s2 = s1;

В данном случае будут вызваны конструктор с параметрами и конструктор копирования соответственно.

1. Какие методы будут вызваны при выполнении следующих операторов:

Student s1(“Ivanov”, 20);

Student s2;

s2 = s1;

В данном случае будут вызваны конструктор с параметрами, конструктор без параметров и конструктор копирования соответственно.

1. Какой конструктор будет использоваться при передаче параметра в функцию print():

void print(student a)

{

a.show();

}

В данном случае будет вызван конструктор копирования.

1. Класс описан следующим образом:

class Student

{

string name;

int age;

public:

void set\_name(string);

void set\_age(int);

…….

};

Student p;

Каким образом можно присвоить новое значение атрибуту name объекта p?

Присвоение нового значения атрибуту name объекта p осуществляется путём обращения к модификатору, которое аналогично обращению к полям структуры:

String row = “Ivanov”; // строка, хранит данные для атрибута name

p.set\_name(string row);