Министерство науки и высшего образования РФ

 Пензенский государственный университет Кафедра "Вычислительная техника"

Отчет по лабораторной работе №1  
  «Основные приемы форматирования. Настройка стилей. Создание таблиц и диаграмм. Разделы, колонки, списки. Составление оглавления»

Работу выполнили студенты группы 23ВВВ3 бригады №9:

Полиневский Вадим

Проверили:

Кандидат т.н. Гудков А.А

Кандидат т.н. Евсеева И. Е

Пенза 2025

**Цель работы**: Изучить основы работы с классами и объектами на языке си++.

**Задаие :**Разработать классы для описанных ниже объектов. Включить в класс конструктор по умолчанию, конструктор с параметрами, конструктор копирования и деструктор. Поля класса определить с модификатором доступа private, для доступа к ним реализовать методы Set- и Get- с модификатором доступа public.

**Описание практической части**:

Для выполнения программы было использовано два модуля: **lab\_1.cpp** с функцией main и файл заголовка **header.h.**

Листинг:

**lab\_1.cpp:**

#include <iostream>

#include <vector>

#include "header.h"

using namespace std;

int main (){

unsigned int count\_objects {3};

vector <Bus> objects;

cout << "lab\_1" << endl;

for (int i = 0; i< count\_objects; i++) {

if (i == 0) Set\_Values\_For\_ClassObject(1, objects, i);

if (i == 1) Set\_Values\_For\_ClassObject(2, objects, i);

if (i == 2) Set\_Values\_For\_ClassObject(3, objects, i);

}

Find\_in\_vector(objects);

}

**header.h:**

#pragma once

#include <iostream>

#include <string>

using namespace std;

class Bus {

private:

string Name\_Driver;

unsigned int Number\_Path;

string Brand\_Bus;

unsigned int Year\_Beginning\_Start;

unsigned int Mileage;

public:

Bus (){

Name\_Driver = "No name";

Brand\_Bus = "No Brand";

Year\_Beginning\_Start = 2000;

}

Bus (string \_Name\_Driver, unsigned int \_Number\_Path, string \_Brand\_Bus, unsigned int \_Year\_Start, unsigned int \_Mileage) {

Name\_Driver = \_Name\_Driver;

Number\_Path = \_Number\_Path;

Brand\_Bus = \_Brand\_Bus;

Year\_Beginning\_Start = \_Year\_Start;

Mileage = \_Mileage;

}

Bus (const Bus& \_object\_BUS) {

Name\_Driver = \_object\_BUS.Name\_Driver;

Number\_Path = \_object\_BUS.Number\_Path;

Brand\_Bus = \_object\_BUS.Brand\_Bus;

Year\_Beginning\_Start = \_object\_BUS.Year\_Beginning\_Start;

Mileage = \_object\_BUS.Mileage;

}

~Bus(){

}

void set\_Name\_Driver(string \_Name) {Name\_Driver = \_Name;}

void set\_Number\_Path(unsigned int \_Number\_Path){Number\_Path = \_Number\_Path;}

void set\_Brand\_Bus(string \_Brand\_Bus){Brand\_Bus = \_Brand\_Bus;}

void set\_Year\_Begginning\_Start(unsigned int \_Year\_Beginning\_Start){Year\_Beginning\_Start = \_Year\_Beginning\_Start;}

void set\_Mileage(unsigned int \_Mileage){Mileage = \_Mileage;}

string get\_Name\_Driver() {return Name\_Driver;}

unsigned int get\_Number\_Path(){return Number\_Path;}

string get\_Brand\_Bus(){return Brand\_Bus;}

unsigned int get\_Year\_Begginning\_Start(){return Year\_Beginning\_Start;}

unsigned int get\_Mileage(){return Mileage;}

};

void Print\_Vulue(Bus temp\_object){

cout << temp\_object.get\_Name\_Driver() << ", "<< temp\_object.get\_Number\_Path()<< ", "<< temp\_object.get\_Brand\_Bus()<< ", "<< temp\_object.get\_Year\_Begginning\_Start() <<", "<< temp\_object.get\_Mileage() << endl;

}

void Enter\_Value(string \*Name, string \*Brand, unsigned int\* Year, unsigned int \*Number, unsigned int \*Milage) {

cout << "ENTER VALUES: " ;

cout << "enter Name\_Driver: ";

cin >> \*Name;

cout << "enter Number\_Path: ";

cin >> \*Number;

cout << "enter Brand\_Bus: ";

cin >> \*Brand;

cout << "enter Year\_Beginning\_Start: ";

cin >> \*Year;

cout << "enter Mileage: ";

cin >> \*Milage;

}

void Set\_Values\_For\_ClassObject(int value, vector<Bus>& Vector, int ptr\_Vector) {

string Name, Brand;

unsigned int Number, Milage , Year;

Enter\_Value(&Name, &Brand, &Year, &Number, &Milage);

switch (value)

{

case 1:

{

Bus object;

object.set\_Name\_Driver(Name);

object.set\_Number\_Path(Number);

object.set\_Brand\_Bus(Brand);

object.set\_Year\_Begginning\_Start(Year);

object.set\_Mileage(Milage);

Vector.push\_back(object);

break;

}

case 2:

{

Bus object(Name, Number, Brand, Year, Milage);

Vector.push\_back(object);

break;

}

case 3:

{

Bus object = Vector[ptr\_Vector - 1];

Vector.push\_back(object);

break;

}

default:

break;

}

}

void Sorting\_by\_Number\_Path(unsigned int number\_path, Bus temp\_object){

cout << "Sortig by number path: " << number\_path << endl;

if (number\_path == temp\_object.get\_Number\_Path()) Print\_Vulue(temp\_object);

cout << "--------" << endl;

}

void Sorting\_by\_Year(Bus temp\_object){

unsigned int year\_now = 2025;

cout << "Sortig by bus using more 10 year:" << endl;

if ((year\_now - temp\_object.get\_Year\_Begginning\_Start()) >= 10) Print\_Vulue(temp\_object);

cout << "--------"<< endl;

}

void Sorting\_by\_Mileage(Bus temp\_object){

cout << "Sorting by mileage" << endl;

if (temp\_object.get\_Mileage() >= 10000) Print\_Vulue(temp\_object);

cout << "--------" << endl;

}

void Find\_in\_vector(vector <Bus> temp\_object){

unsigned int temp\_number\_path;

cout << "Enter number path for find: ";

cin >> temp\_number\_path;

for (int i = 0; i< temp\_object.size(); i++) {

Sorting\_by\_Number\_Path(temp\_number\_path ,temp\_object[i]);

}

for (int i = 0; i < temp\_object.size(); i++) {

Sorting\_by\_Year(temp\_object[i]);

}

for (int i =0; i < temp\_object.size(); i++) {

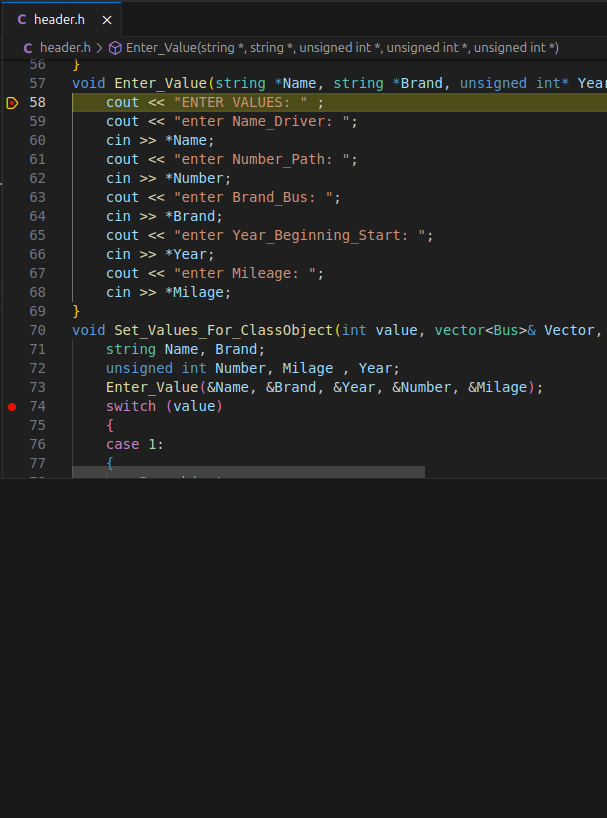
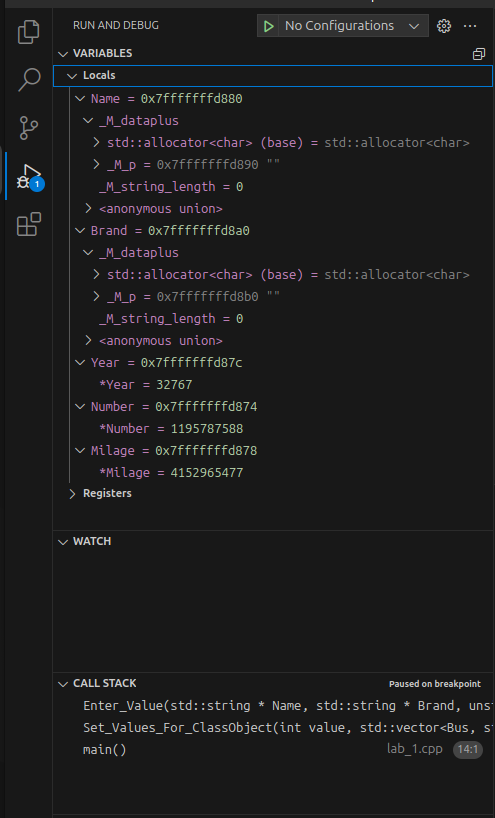
Sorting\_by\_Mileage(temp\_object[i]);

}

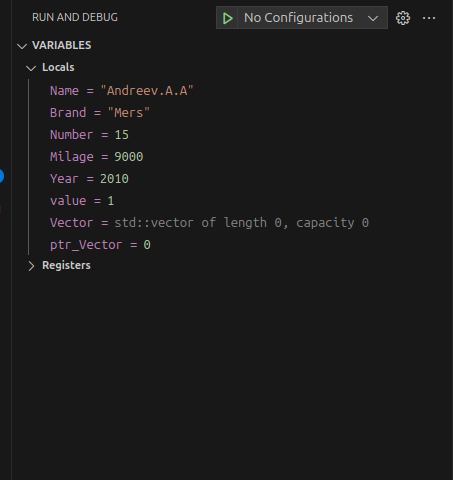
}

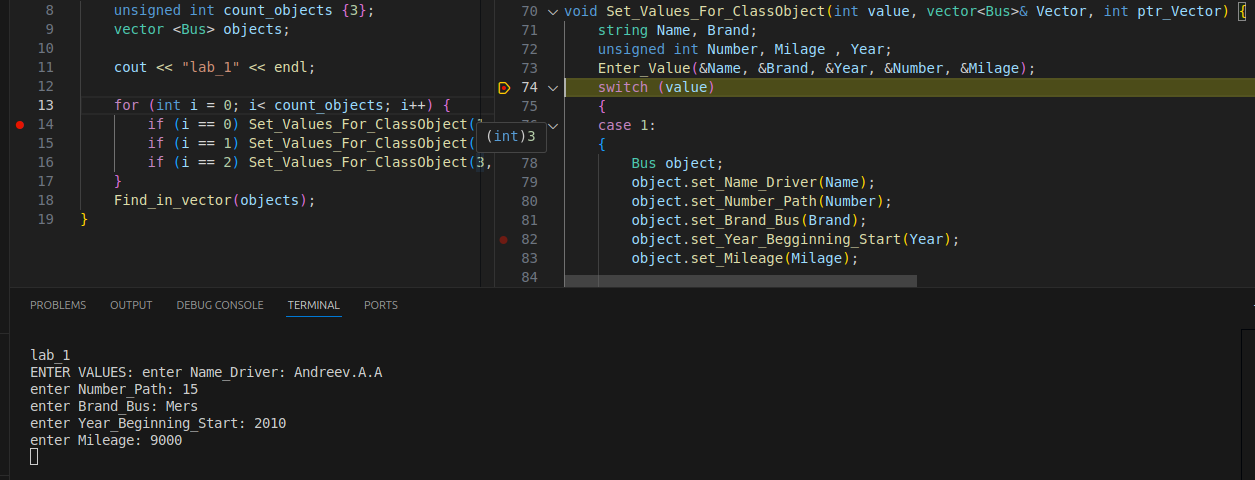
**Отладка:**

Значенение переменных, которые будут использованы для передачи в сеттеры, до пользовательского ввода:

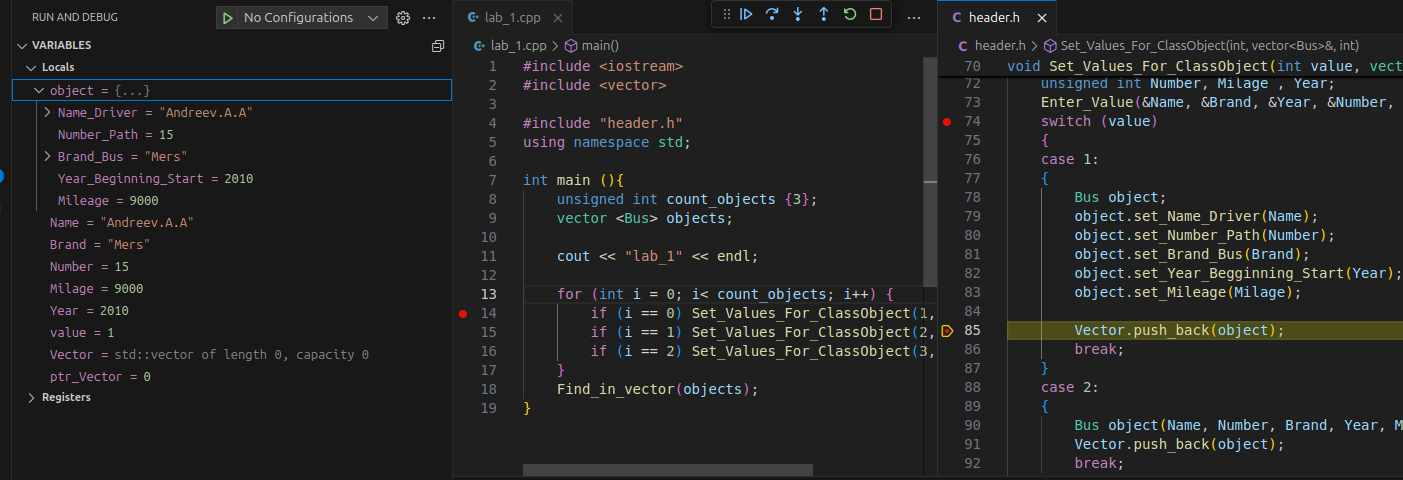


Значение переменных после пользовательского ввода:

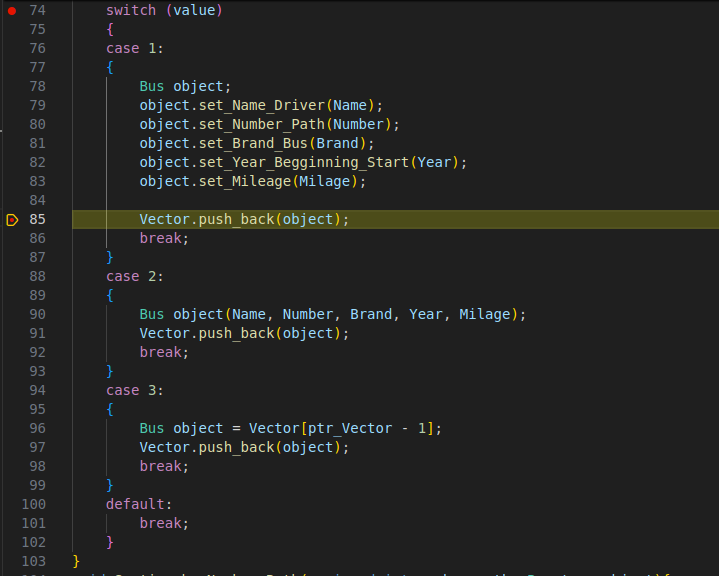




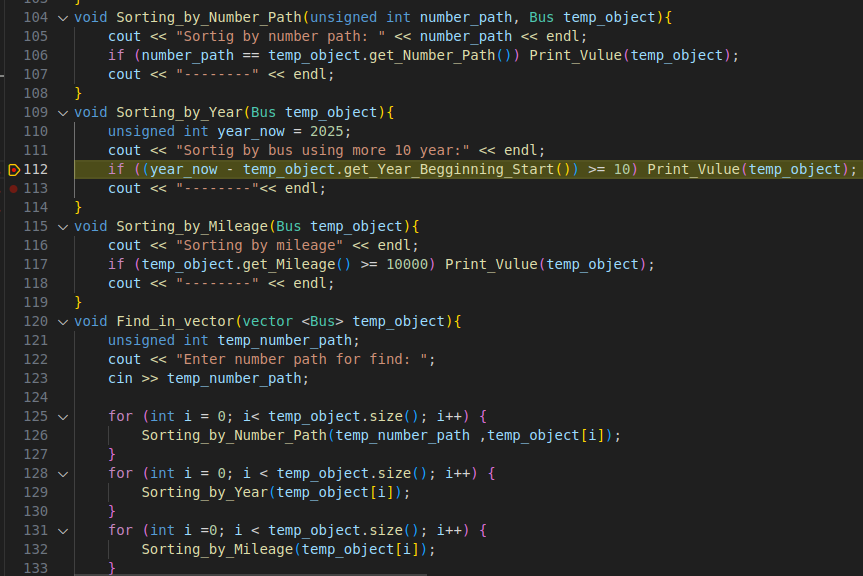
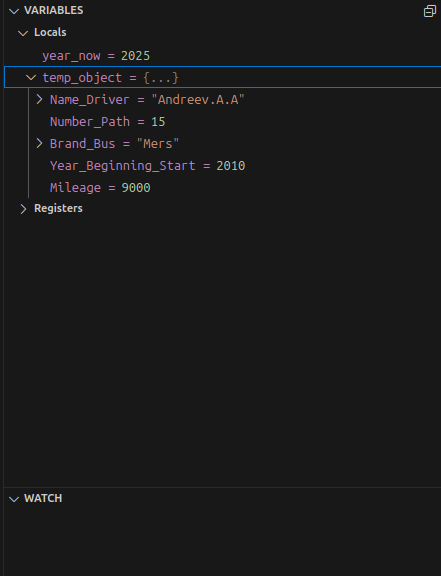
Значение полей класса после пользовательского ввода и перед занесением в вектор(массив):



В данном swith, в каждом case реализованы все виды деструкторов (1 - конструтор по умолчанию с последующей передачей в private поля через сеттеры, 2 - заполнение полей через конструктор с параметрами, 3 - констуктор копирования (передается предыдущий в векторе объект)).



Точка останова после сортировки по номеру маршрута и перед выводом результата сортировки по времени использвания.



**Terminal-Log:**

lab\_1

ENTER VALUES: enter Name\_Driver: Andreev.A.A

enter Number\_Path: 15

enter Brand\_Bus: Mers

enter Year\_Beginning\_Start: 2010

enter Mileage: 9000

ENTER VALUES: enter Name\_Driver: Ivanov.I.I

enter Number\_Path: 30

enter Brand\_Bus: Mers

enter Year\_Beginning\_Start: 2005

enter Mileage: 15000

ENTER VALUES: enter Name\_Driver: Dmitriev.D.D

enter Number\_Path: 45

enter Brand\_Bus: Mers

enter Year\_Beginning\_Start: 2009

enter Mileage: 12000

Enter number path for find: 30

Sortig by number path: 30

--------

Sortig by number path: 30

Ivanov.I.I, 30, Mers, 2005, 15000

--------

Sortig by number path: 30

Ivanov.I.I, 30, Mers, 2005, 15000

--------

Sortig by bus using more 10 year:

Andreev.A.A, 15, Mers, 2010, 9000

--------

Sortig by bus using more 10 year:

Ivanov.I.I, 30, Mers, 2005, 15000

--------

Sortig by bus using more 10 year:

Ivanov.I.I, 30, Mers, 2005, 15000

--------

Sorting by mileage

--------

Sorting by mileage

Ivanov.I.I, 30, Mers, 2005, 15000

--------

Sorting by mileage

Ivanov.I.I, 30, Mers, 2005, 15000

--------

[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-yzwrlnuu.h02" 1>"/tmp/Microsoft-MIEngine-Out-xrtlw3vz.sts"