Музафаров К.Р. ИДБ-21-06
Лабараторная работа 2
Часть 1

```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
using namespace std;

class Car
{
public:
    Car() {}
    virtual ~Car() {}
};

class Porsche: public Car
{
    string model;
public:
    Porsche(string car) { model = car; }
    ~Porsche() { cout << "Porsche " << model << endl; }
};
class Tesla: public Car
{
    string model;
public:
    Tesla(string car) { model = car; }
    ~Tesla() { cout << "Tesla " << model << endl; }
};
class Lada: public Car
{
    string model;
public:
    Lada(string car){ model = car; }
    ~Lada() { cout << "Lada " << model << endl;}
};

int main()
{
    setlocale(LC_ALL, "Russian");
    vector <Car*> v;
    ifstream File("/Users/karimmuzafarov/Desktop/
myfile.txt");
    string iter;
    cout << "Список автомобилей: \n";
    while (!File.eof())
```

```
    {
        File >> iter;
        if (iter == "Porsche") {
            File >> iter;
            v.push_back(new Porsche(iter));
        }

        else if (iter == "Tesla")
        {
            File >> iter;
            v.push_back(new Tesla(iter));
        }
        else if (iter == "Lada")
        {
            File >> iter;
            v.push_back(new Lada(iter));
        }
    }

    for (int i = 0; i < v.size(); i++)
        delete v[i];

return 0;
}
```

Текст файла txt:	**Вывод программы:**
Porsche 911	**Porsche 911**
Tesla modelS	**Tesla modelS**
Lada 12	**Lada 12**
Porsche 711	**Porsche 711**
Tesla modelX	**Tesla modelX**
Lada 14	**Lada 14**
Porsche 651	**Porsche 651**
Tesla modelY	**Tesla modelY**
Lada 7	**Lada 7**
Porsche 911	**Porsche 911**
Tesla modelZ	**Tesla modelZ**
Lada 14	**Lada 14**

Часть 2

```
#include <iostream>
#include <vector>
#include <typeinfo>
using namespace std;
struct Leaks
{
	~Leaks() { _CrtDumpMemoryLeaks(); }
};
Leaks _l;
class Class1 {
	int a;
public:
	Class1() { a = 0; }
	Class1(int a1) {
		a = a1;
	}
	Class1(const Class1& obj) {
	    a = obj.a;
	}
	virtual Class1* my_copy() {
		Class1* new_obj = new Class1(*this);
		return new_obj;
	}
	virtual ~Class1() = default;
};
class Class2 : public Class1
{
    double b;
public:
    Class2()
    {
        b = 0.0;
    }
	Class2(double b1) {
	    b = b1;
	}
	Class2(const Class2& obj) : Class1(obj) {
		b = obj.b;
	}
	Class2* my_copy() {
		Class2* new_obj = new Class2(*this);
		return new_obj;
	}
	~Class2() = default;
};
int main() {
```

```
      vector <Class1*> DB;
      auto a = 5; auto b = 7.3;
      Class1* a1 = new Class1(a);
      Class2* b1 = new Class2(b);
      DB.push_back(a1->my_copy());
      DB.push_back(b1->my_copy());
      for (int i = 0; i < DB.size(); i++) {
          cout << typeid(DB[i]).name() << endl;
      }
      for (int i = 0; i < DB.size(); i++) {
            delete DB[i];
}
delete a1;
delete b1;
return 0;
}
```