Министерство образования Республики Беларусь

УО «Брестский государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №2**

По дисциплине: “Языки программирования”

Тема: “Наследование и виртуальные функции”**Вариант №12**

**Выполнил**: студент 2 курса группы ПО-7 Малинин Егор Севич

**Проверила:** Дряпко А.В.

Брест 2021

**Цель:**

Получить практические навыки создания иерархии классов ииспользованиястатических компонентов класса.

**Постановка задачи:**

Написать программу, в которой создается иерархия классов. Включить полиморфные объекты в связанный список, используя статические компоненты класса. Показать использование виртуальных функций.

Порядок выполнения работы:

1. Определить иерархию классов (в соответствии с вариантом).
2. Определить в классе статическую компоненту - указатель на начало связанного списка объектов и статическую функцию для просмотра списка.
3. Реализовать классы.
4. Написать демонстрационную программу, в которой создаются
5. объекты различных классов и помещаются в список, после чего списокпросматривается.
6. Сделать соответствующие методы не виртуальными и посмотреть, что будет.
7. Реализовать вариант, когда объект добавляется в список при создании, т.е. в конструкторе (смотри пункт 6 следующего раздела).

**Вариант задания:**

Перечень классов:двигатель, двигатель внутреннего сгорания, дизель, турборе-активный двигатель

**Код программы:**

**main.cpp**

#include "engine.h"  
  
list <Engine\*>Engine::objects{};  
  
int main() {  
Combustion\_engine first(1, 2, 3 );  
Diesel second(3, 2, 1, "stop");  
Combustion\_engine third(7, 9, 4 );  
Turbojet\_engine forth(6, 15, "Tesla");  
Engine::print();  
}

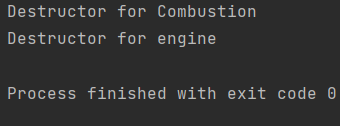
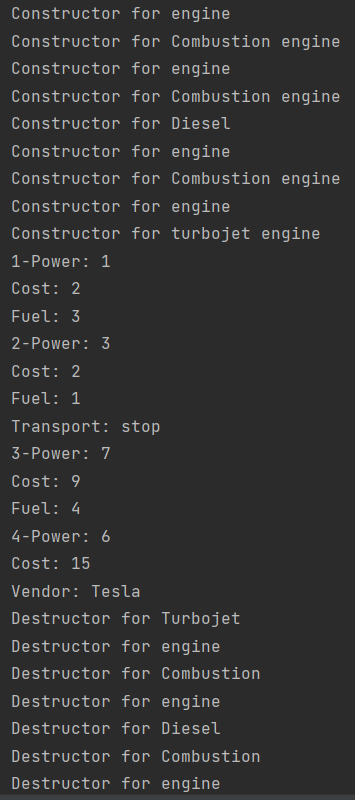
**engine.h**

#ifndef LAB\_2\_ENGINE\_H  
#define LAB\_2\_ENGINE\_H  
#include "iostream"  
#include "string"  
#include "list"  
using namespace std;  
  
  
class Engine{//двигатель  
static list<Engine\*>objects;  
protected:  
int power\_r;//цена  
float cost\_t;//стоимость  
public:  
Engine(int power, float cost);  
 virtual ~Engine();  
 static void print();  
 virtual void show() = 0;  
 void add();  
};  
  
  
class Combustion\_engine : public Engine{//двигательвнутреннегосгорания  
protected:  
int fuel\_l;//расходтоплива  
public:  
Combustion\_engine(int power, float cost, int fuelL);  
 void show() override;  
 virtual ~Combustion\_engine();  
};  
  
  
class Diesel : public Combustion\_engine{//дизель  
protected:  
string transport\_t;//видтранспотра  
public:  
Diesel(int power, float cost, int fuel, string transport\_t);  
 virtual ~Diesel();  
 void show() override;  
};  
  
  
class Turbojet\_engine : public Engine{//турбореактивныйдвигатель  
protected:  
string vendor\_r; //поставщик  
public:  
Turbojet\_engine(int power, float cost, string vendor);  
 virtual ~Turbojet\_engine();  
 void show() override;  
};  
#endif //LAB\_2\_ENGINE\_H

**engine.cpp**

#include "engine.h"  
void Engine::add() {  
objects.push\_back(this);  
}  
  
void Engine ::print() {  
int num =0;  
 for(Engine\* obj : objects) {  
 num++;  
cout <<num <<"-";  
obj->show();  
}  
}  
  
void Combustion\_engine:: show() {  
 cout <<"Power: " <<power\_r <<"\nCost: " <<cost\_t <<"\nFuel: " <<fuel\_l <<endl;  
}  
  
void Diesel:: show() {  
 cout <<"Power: " <<power\_r <<"\nCost: " <<cost\_t <<"\nFuel: " <<fuel\_l <<"\nTransport: " <<transport\_t <<endl;  
}  
  
void Turbojet\_engine:: show() {  
 cout <<"Power: " <<power\_r <<"\nCost: " <<cost\_t <<"\nVendor: " <<vendor\_r <<endl;  
}  
  
  
Engine::Engine(int power, float cost)  
 : power\_r(power), cost\_t(cost) {  
 cout <<"Constructor for engine" <<endl;  
add();  
}  
  
Combustion\_engine::Combustion\_engine(int power, float cost, int fuel)  
 : Engine(power, cost), fuel\_l(fuel){  
 cout <<"Constructor for Combustion engine"<<endl;  
}  
  
Diesel::Diesel(int power, float cost, int fuelL, string transport)  
 : Combustion\_engine(power, cost, fuelL), transport\_t(transport) {  
 cout <<"Constructor for Diesel" <<endl;  
}  
  
Engine::~Engine() {  
 cout <<"Destructor for engine"<<endl;  
}  
  
Combustion\_engine::~Combustion\_engine() {  
 cout <<"Destructor for Combustion "<<endl;  
}  
  
Diesel::~Diesel() {  
 cout <<"Destructor for Diesel"<<endl;  
}  
  
Turbojet\_engine::~Turbojet\_engine() {  
 cout <<"Destructor for Turbojet"<<endl;  
}  
  
Turbojet\_engine::Turbojet\_engine(int power, float cost, string vendor)  
: Engine(power, cost), vendor\_r(vendor) {  
 cout <<"Constructor for turbojet engine"<<endl;  
}

**Результат программы:**



**Вывод:**

Я получил практические навыки создания иерархии классов ииспользованиястатических компонентов класса.