**Прикладное программирование**

Практика 3

Объектная модель Машина. Поиск в массиве объектов.

Каждую программу сохранять в отдельном файле для каждого упражнения,

например prog03-1, prog03-2 и т.д., где "03" - номер практического

занятия, 1, 2 и т.д. - номер задания.

1. Примените инкрементный метод конструирования программы для решения следующей

задачи. По примеру 1 из лекции создайте программный модуль, содержащий

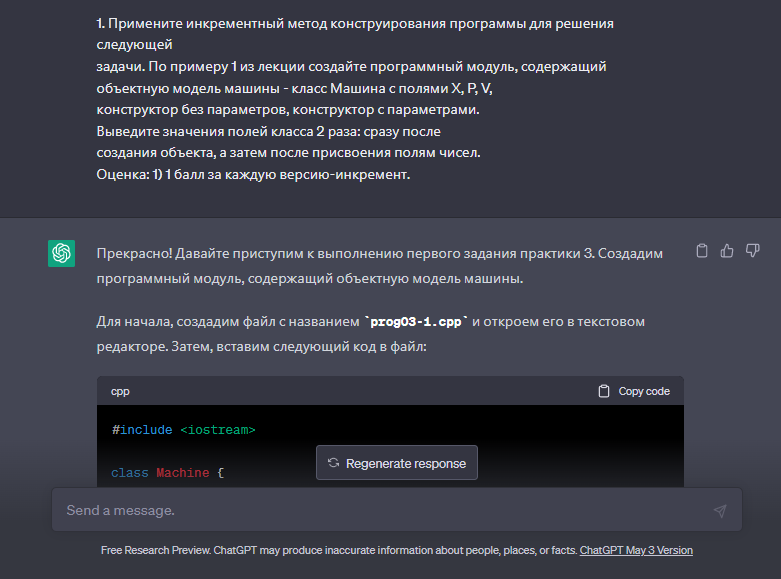
объектную модель машины - класс Машина с полями X, P, V,

конструктор без параметров, конструктор с параметрами.

Выведите значения полей класса 2 раза: сразу после

создания объекта, а затем после присвоения полям чисел.

Оценка: 1) 1 балл за каждую версию-инкремент.


#include <iostream>

class Machine {

public:

double X;

int P;

float V;

Machine() {

X = 0.0;

P = 0;

V = 0.0;

}

Machine(double x, int p, float v) {

X = x;

P = p;

V = v;

}

void printValues() {

std::cout << "X: " << X << ", P: " << P << ", V: " << V << std::endl;

}

};

int main() {

Machine m1; // Создание объекта с помощью конструктора без параметров

m1.printValues();

Machine m2(10.5, 3, 5.7); // Создание объекта с помощью конструктора с параметрами

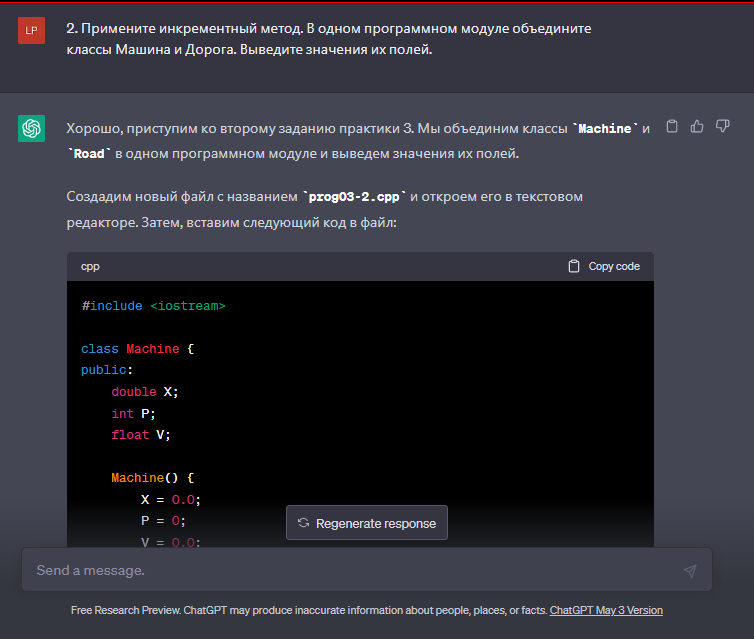
m2.printValues();

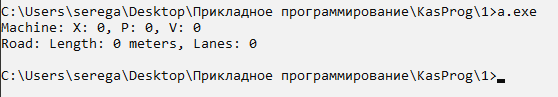
return 0;

}

2. Примените инкрементный метод. В одном программном модуле объедините

классы Машина и Дорога. Выведите значения их полей.





#include <iostream>

class Machine {

public:

double X;

int P;

float V;

Machine() {

X = 0.0;

P = 0;

V = 0.0;

}

Machine(double x, int p, float v) {

X = x;

P = p;

V = v;

}

void printValues() {

std::cout << "Machine: X: " << X << ", P: " << P << ", V: " << V << std::endl;

}

};

class Road {

public:

int length;

int lanes;

Road() {

length = 0;

lanes = 0;

}

Road(int l, int n) {

length = l;

lanes = n;

}

void printValues() {

std::cout << "Road: Length: " << length << " meters, Lanes: " << lanes << std::endl;

}

};

int main() {

Machine m;

Road r;

m.printValues();

r.printValues();

return 0;

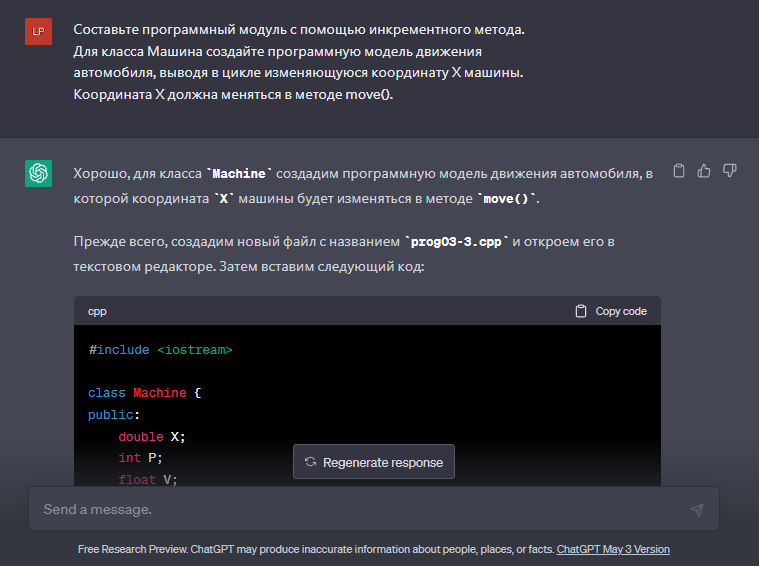
}

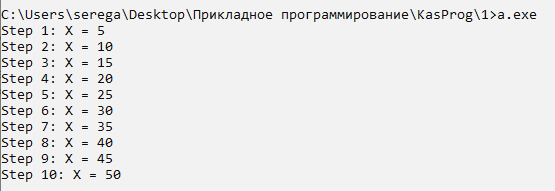
3. Составьте программный модуль с помощью инкрементного метода.

Для класса Машина создайте программную модель движения

автомобиля, выводя в цикле изменяющуюся координату X машины.

Координата X должна меняться в методе move().





#include <iostream>

class Machine {

public:

double X;

int P;

float V;

Machine() {

X = 0.0;

P = 0;

V = 0.0;

}

Machine(double x, int p, float v) {

X = x;

P = p;

V = v;

}

void move() {

for (int i = 0; i < 10; i++) {

X += V; // Изменяем координату X машины с помощью скорости V

std::cout << "Step " << i + 1 << ": X = " << X << std::endl;

}

}

};

int main() {

Machine m(0.0, 0, 5.0); // Создаем объект машины с начальными значениями

m.move(); // Вызываем метод движения машины

return 0;

}

4. Класс Книга TBook содержит поля Название, Автор, Количество

экземпляров (Name, Author, Count). Напишите программу,

в которой создается массив объектов-книг, выполняется поиск

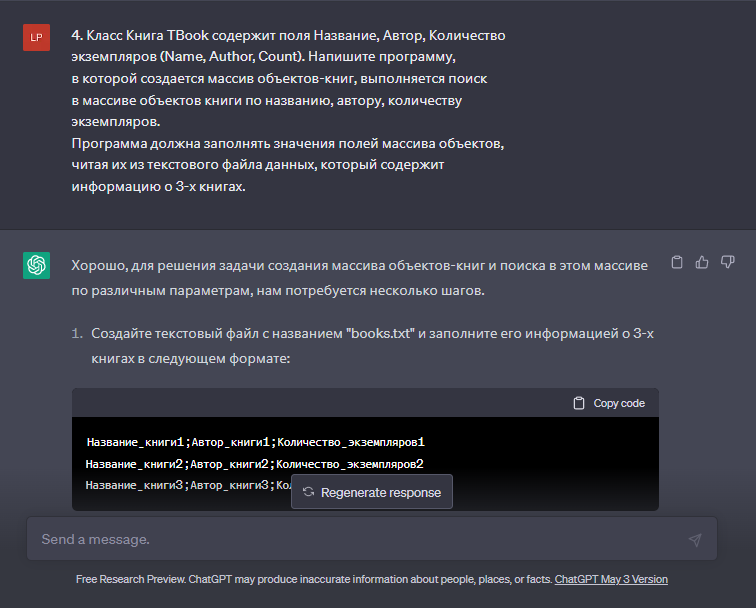
в массиве объектов книги по названию, автору, количеству

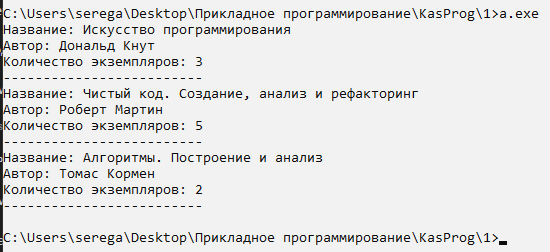
экземпляров.

Программа должна заполнять значения полей массива объектов,

читая их из текстового файла данных, который содержит

информацию о 3-х книгах.





#include <iostream>

#include <fstream>

#include <string>

class TBook {

public:

std::string Name;

std::string Author;

int Count;

TBook() {

Name = "";

Author = "";

Count = 0;

}

TBook(const std::string& name, const std::string& author, int count) {

Name = name;

Author = author;

Count = count;

}

void Print() {

std::cout << "Название: " << Name << std::endl;

std::cout << "Автор: " << Author << std::endl;

std::cout << "Количество экземпляров: " << Count << std::endl;

std::cout << "-------------------------" << std::endl;

}

};

int main() {

const int NUM\_BOOKS = 3;

TBook books[NUM\_BOOKS];

std::ifstream inputFile("books.txt");

if (inputFile.is\_open()) {

for (int i = 0; i < NUM\_BOOKS; i++) {

std::string line;

if (std::getline(inputFile, line)) {

std::string name, author;

int count;

size\_t pos1 = line.find(';');

size\_t pos2 = line.find(';', pos1 + 1);

name = line.substr(0, pos1);

author = line.substr(pos1 + 1, pos2 - pos1 - 1);

count = std::stoi(line.substr(pos2 + 1));

books[i] = TBook(name, author, count);

}

}

inputFile.close();

}

else {

std::cout << "Ошибка открытия файла!" << std::endl;

return 1;

}

// Вывод информации о книгах

for (int i = 0; i < NUM\_BOOKS; i++) {

books[i].Print();

}

return 0;

}