Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)

**Лабораторная работа № 1  
Классы C++**

Выполнил студент группы ИУ8-24

Конов Михаил

**ЦЕЛЬ**

Цель работы состоит в изучении основных понятий объектно- ориентированного программирования языка Си++ – классов и объектов, и овладении навыками разработки программ на языке Си++ с использованием объектно- ориентированных средств.

**УСЛОВИЕ ЗАДАЧИ**  
Вариант 10

Класс - книга. Параметры (поля): автор, название, количество страниц. Статус доступа всех полей private. Класс включает: конструктор, функцию печать параметров, при необходимости функции доступа к полям. Распечатать параметры книги с максимальным количеством страниц.

**ТЕКСТ ПРОГРАММЫ**

#include <iostream>

#include <fstream>

#include <vector>

#include <string>

using namespace std;

class Book{

private:

string author;

string title;

int pages;

public:

//пустой конструктор

Book(){}

//конструктор для инициализации полей

Book(string a, string t, int p){

author = a;

title = t;

pages = p;

}

//функция для инициализации полей

void set(string a, string t, int p){

author = a;

title = t;

pages = p;

}

//функция для печати полей

print(){

cout << "Автор: " << author << endl;

cout << "Название книги: " << title << endl;

cout << "Количество страниц: " << pages << endl;

}

//get- функции

int get\_pages(){return pages;}

string get\_author(){return author;}

string get\_title(){return title;}

//set- функции

void set\_title(string t){title = t;}

void set\_pages(int p){pages = p;}

void set\_author(string a){author = a;}

};

void reading\_check(ifstream& s){

cout << "Считано байт: " << s.tellg() << endl;

}

void output\_function(Book& B1, ofstream& f){

B1.print();

f << B1.get\_author() << endl << B1.get\_title() << endl << B1.get\_pages();

}

int main()

{

setlocale(LC\_ALL, "RUSSIAN");

ifstream book\_list("input.txt");

//считываем количество книг (оптимизировать

int n;

book\_list >> n;

book\_list.ignore();

//создаем динамический массив

Book \*pBook;

pBook = new Book[n];

//создаем вектор

vector<Book> VBook(n);

reading\_check(book\_list);

//вспомогательные переменные для записи в массив

string atr = "";

string ttl = "";

int pgs = 0;

reading\_check(book\_list);

//считываем в массив и в вектор

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

getline(book\_list, atr);

getline(book\_list, ttl);

book\_list >> pgs;

book\_list.ignore();

pBook[i].set(atr, ttl, pgs);

VBook[i].set(atr, ttl, pgs);

//вспомогательные функции для проверки записи

pBook[i].print();

VBook[i].print();

}

reading\_check(book\_list);

cout << "Книги успешно считаны из файла!"<< endl;

//открываем вывод

ofstream ot("output.txt");

//ищем самую длинную книгу в массиве и записываем в пустую

Book B0("", "", 0);

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

if (pBook[i].get\_pages() > B0.get\_pages()){

B0.set\_pages(pBook[i].get\_pages());

B0.set\_author(pBook[i].get\_author());

B0.set\_title(pBook[i].get\_title());

}

}

//выводим

cout << "\nКнига с наибольшим количеством страниц (в массиве): ";

ot << "\nКнига с наибольшим количеством страниц (в массиве): ";

output\_function(B0, ot);

//ищем самую длинную книгу в векторе и записываем в пустую

Book B1("","", 0);

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

Book& tmp = VBook[i];

if(tmp.get\_pages() > B1.get\_pages()){

B1.set\_pages(tmp.get\_pages());

B1.set\_author(tmp.get\_author());

B1.set\_title(tmp.get\_title());

}

}

//выводим

cout << "\nКнига с наибольшим количеством страниц (в векторе): ";

ot << "\nКнига с наибольшим количеством страниц (в векторе): ";

output\_function(B1, ot);

//закрываем файл и сворачиваемся

book\_list.close();

return 0;

}

**ВЫВОД**

В ходе данной лабораторной работы мы создали класс Book с полями author, title, pages, которые служат для хранения данных об объекте класса: автора, названия и страниц соответственно. Заданные конструкторы и методы позволяют как создавать объекты класса из готовых значений, так и присваивать значения изначально пустым объектам. Методы get- и set- класса Book позволяют получать доступ к полям класса со статусом доступа private, изменяя или получая их значения. Для хранения множества книг использованы динамический массив и контейнер библиотеки STL vector.

Для демонстрации класса приведена программа, в которой из файла input.txt считываются несколько книг в динамический массив и в вектор, затем среди них находится книга с наибольшим количеством страниц и записывается в файл output.txt.

Этот класс может быть полезен при работе с строковыми и числовыми данными в программировании на C++.