Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.8**

Дисциплина: «информатика»

Тема: простое наследование

Вариант 5

Выполнил работу

студент группы РИС-20-1б

Зверев А.Д.

Проверила

Доцент кафедры ИТАС

Полякова О.А.

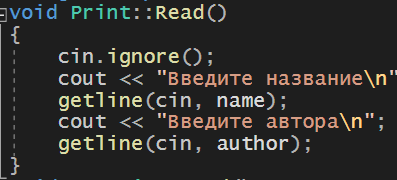
Пермь, 2021

**Постановка задачи**

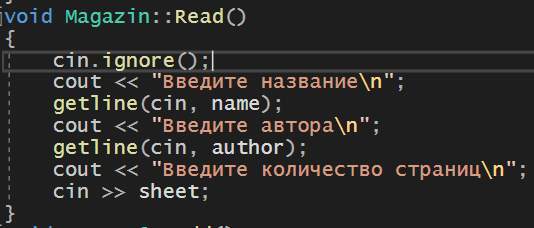
1. Определить иерархию пользовательских классов (см. лабораторную работу №5). Во главе иерархии должен стоять абстрактный класс с чисто виртуальными методами для ввода и вывода информации об атрибутах объектов.
2. Реализовать конструкторы, деструктор, операцию присваивания, селекторы и модификаторы.
3. Определить класс—группу на основе структуры, указанной в варианте.
4. Для группы реализовать конструкторы, деструктор, методы для добавления и удаления элементов в группу, метод для просмотра группы, перегрузить операцию для получения информации о размере группы.
5. Определить класс Диалог—наследника группы, в котором реализовать методы для обработки событий.
6. Добавить методы для обработки событий группой и объектами пользовательских классов.
7. Написать тестирующую программу.
8. Нарисовать диаграмму классов и диаграмму объектов.
9. Создать базовый класс ПЕЧАТНОЕ\_ИЗДАНИЕ(PRINT), название и автор типа string.
10. Создать производный класс ЖУРНАЛ (MAGAZIN). Количество страниц-int.
11. Создать группу Вектор (Vector).
12. В программе должны присутствовать команды:
13. Создать группу (формат команды: m количество элементов группы).
14. Добавить элемент в группу (формат команды +).
15. Удалить элемент из группы (формат команды —).
16. Вывести информацию об элементах группы (формат команды: s).
17. Вывести информацию об названиях всех элементов группы (формат команды: z).
18. Конец работы (формат команды: q).

**Анализ задачи**

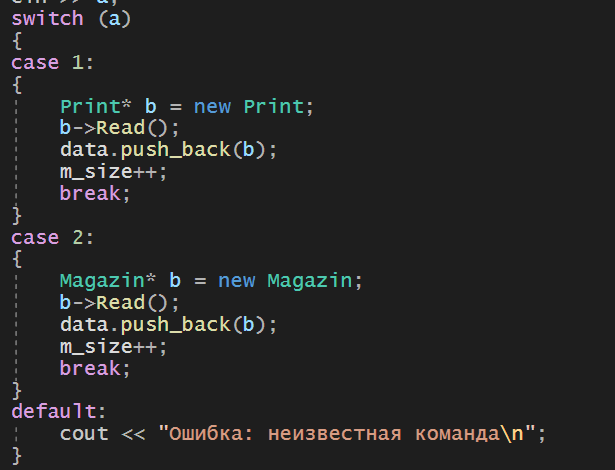
1. Для решения задачи необходимо…
   1. Написать функцию Print::Read типа void, в которой в программу заносится названия печатных изданий с клавиатуры;



* 1. Написать функцию Magazin::Read типа void, в которой в программу вводятся все данные об журналах;



* 1. Написать switch, который будет отвечать за корректность введенных данных с клавиатуры и вызывать различные функции;



1. В программе были использованы следующие типы данных:
   1. Тип данных int для хранения введенного выбора;



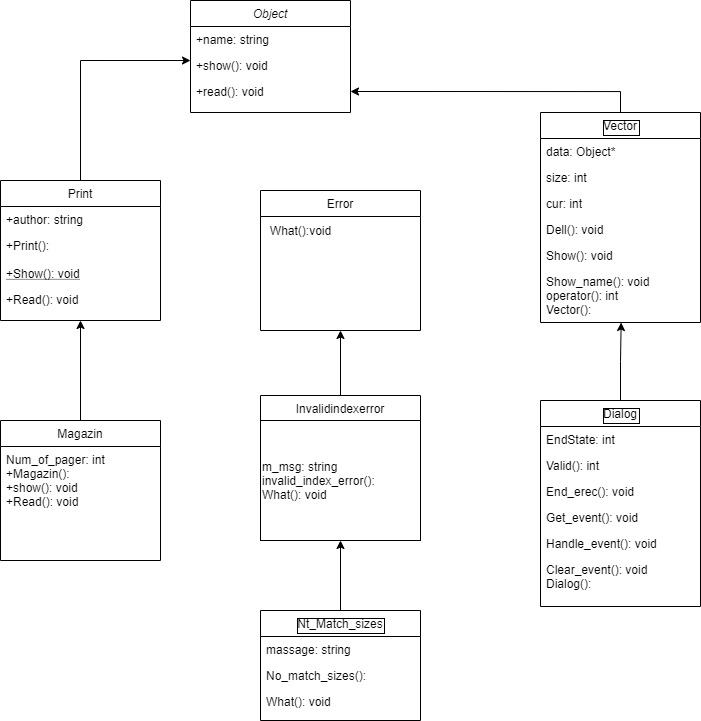
* 1. Тип данных string для хранения данных о введенных произведениях;



* 1. Тип данных void для различных функций;



**Диаграмма классов**



**Решение**

#include "lab18.h"

#include <string>

#include <iostream>

#include <fstream>

#include <math.h>

using namespace std;

void Print::Read()

{

cin.ignore();

cout << "Введите название\n";

getline(cin, name);

cout << "Введите автора\n";

getline(cin, author);

}

void Magazin::Read()

{

cin.ignore();

cout << "Введите название\n";

getline(cin, name);

cout << "Введите автора\n";

getline(cin, author);

cout << "Введите количество страниц\n";

cin >> sheet;

}

void Vector8::Add()

{

int a;

cout << "Выберите тип добавляемого объекта\n";

cout << "1. печатное издание\n";

cout << "2. журнал\n";

cin >> a;

switch (a)

{

case 1:

{

Print\* b = new Print;

b->Read();

data.push\_back(b);

m\_size++;

break;

}

case 2:

{

Magazin\* b = new Magazin;

b->Read();

data.push\_back(b);

m\_size++;

break;

}

default:

cout << "Ошибка: неизвестная команда\n";

}

}

void Vector8::Dell()

{

data.pop\_back();

m\_size--;

}

void Dialog::GetEvent(TEvent& event)

{

string commands = "+-szq";

char s;

cout << "Введите операцию\n+ добавить элемент\n- удалить элемент\ns вывод\nz вывод названий\nq конец\n";

cin >> s;

if (commands.find(s) >= 0)

{

event.what = 100;

event.command = commands.find(s) + 1;

}

else event.what = 0;

}

int Dialog::Execute()

{

TEvent event;

do

{

EndState = 0;

GetEvent(event);

HandleEvent(event);

} while (!EndState);

return EndState;

}

void Dialog::HandleEvent(TEvent& event)

{

if (event.what == 100)

{

switch (event.command)

{

case 1:

Add();

ClearEvent(event);

break;

case 2:

Dell();

ClearEvent(event);

break;

case 3:

Show();

ClearEvent(event);

break;

case 4:

ShowName();

ClearEvent(event);

break;

case 5:

EndState = 1;

ClearEvent(event);

break;

}

}

}

void Dialog::ClearEvent(TEvent& event) { event.what = 0; }

int Vector8::operator()() { return this->m\_size; }

void Vector8::Show() { for (int i = 0; i < m\_size; i++) data[i]->Show(); }

void Vector8::ShowName() { for (int i = 0; i < m\_size; i++) cout << data[i]->name << endl; }

void Print::Show() { cout << this->name << ":" << this->author << endl; }

void Magazin::Show() { cout << this->name << ":" << this->author << ":" << this->sheet << endl; }

#include <iostream>

#include <fstream>

#include "lab18.h"

#include <string>

#include <windows.h>

#include <vector>

#include <list>

#include <queue>

#include <stack>

#include <set>

#include <map>

#include <algorithm>

#include <numeric>

using namespace std;

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Vector8 V;

Dialog D;

D.Execute();

}

#pragma once

#include <string>

#include <iostream>

#include <list>

#pragma once

#include <string>

#include <iostream>

#include <list>

#include <vector>

#include <fstream>

using namespace std;

struct MyError

{

virtual void what() = 0;

};

struct InvalidIndexError : public MyError

{

string m\_msg;

InvalidIndexError() { m\_msg = "Exception: invalid list index"; }

void what() { cout << m\_msg << endl; }

};

struct NoMatchSizes : public MyError

{

string m\_msg;

NoMatchSizes() { m\_msg = "Exception: sizes of lists are different"; }

void what() { cout << m\_msg << endl; }

};

class Object8

{

public:

string name;

virtual void Show() = 0;

virtual void Read() = 0;

};

class Print : public Object8

{

public:

string author;

Print() {}

Print(string Name, string Author)

{

name = Name;

author = Author;

}

void Show();

void Read();

};

class Magazin : public Print

{

public:

int sheet;

Magazin() {}

Magazin(string Name, string Author, int Sheet) :Print(Name, Author)

{

sheet = Sheet;

}

void Show();

void Read();

};

struct Vector8

{

vector <Object8\*> data;

int m\_size;

int cur;

void Add();

void Dell();

void Show();

void ShowName();

int operator()();

Vector8() { m\_size = 0; }

};

struct TEvent

{

int what;

union

{

int command;

struct

{

int message;

int a;

};

};

};

struct Dialog : Vector8

{

int EndState;

int Valid();

void EndExec();

int Execute();

void GetEvent(TEvent& event);

void HandleEvent(TEvent& event);

void ClearEvent(TEvent& event);

Dialog() { EndState = 0; }

};

**Скриншоты**

