МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра информационные технологии и автоматизированные системы

**Дисциплина Информатика**

**Управление событиями**

Выполнил студент ИВТ-22-2б:

Мифтахов Марат Ринатович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил доцент кафедры ИТАС:

Полякова Ольга Андреевна

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Пермь, 2023

**Постановка задачи**

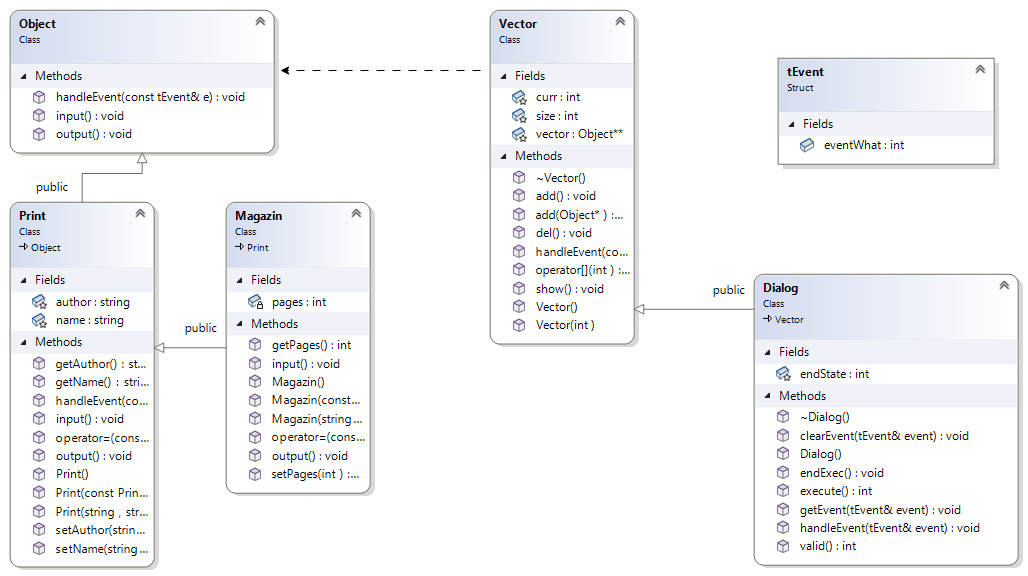
1. Определить иерархию пользовательских классов (см. лабораторную работу №5). Во главе иерархии должен стоять абстрактный класс с чисто виртуальными методами для ввода и вывода информации об атрибутах объектов.
2. Реализовать конструкторы, деструктор, операцию присваивания, селекторы и модификаторы.
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.

**Задание**

1. Базовый класс: ПЕЧАТНОЕ\_ИЗДАНИЕ(PRINT)
2. Поля название– string автор – string
3. Производный класс: ЖУРНАЛ (MAGAZIN)
4. Количество страниц - int
5. Группа – Вектор
6. Команды:
7. Создать группу (формат команды: *m* количество элементов группы).
8. Добавить элемент в группу (формат команды: *+)*
9. Удалить элемент из группы (формат команды *-)*
10. Вывести информацию об элементах группы (формат команды: *s*)
11. Вывести информацию о названиях всех элементов группы (формат команды: *z*)
12. Конец работы (формат команды: *q*)

**Диаграмма классов**

Ниже представлена UML-диаграмма классов.



**Программный код**

Файл Print.h:

#pragma once

#include "Event.h"

#include <string>

#include <iostream>

using namespace std;

class Object {

public:

virtual void input() = 0;

virtual void output() = 0;

virtual void handleEvent(const tEvent& e) = 0;

};

class Print : public Object

{

protected:

string name;

string author;

public:

Print();

Print(string, string);

Print(const Print&);

string getName();

void setName(string);

string getAuthor();

void setAuthor(string);

void handleEvent(const tEvent& e) {

if (e.eventWhat == eventMessage) {

switch (e.command) {

case c\_getAttr:

output();

}

}

}

void input() override {

cin >> name;

cin >> author;

}

void output() override {

cout << "name: " << name << endl;

cout << "author: " << author << endl;

}

Print& operator=(const Print&);

friend ostream& operator<<(ostream&, const Print&);

friend istream& operator>>(istream&, Print&);

};

Файл Magazin.h:

#pragma once

#include "Print.h"

class Magazin :

public Print

{

int pages;

public:

Magazin();

Magazin(string, string, int);

Magazin(const Magazin&);

void setPages(int);

int getPages();

void output() override {

Print::output();

cout << "pages: " << pages << endl;

}

void input() override {

Print::input();

cin >> pages;

}

Magazin& operator=(const Magazin&);

friend ostream& operator<<(ostream&, const Magazin&);

friend istream& operator>>(istream&, Magazin&);

};

Файл Dialog.h:

#pragma once

#include "Vector.h"

#include "Event.h"

class Dialog : public Vector {

protected:

int endState;

public:

Dialog();

~Dialog() {};

void getEvent(tEvent& event);

int execute();

void handleEvent(tEvent& event);

void clearEvent(tEvent& event);

int valid();

void endExec();

};

Dialog::Dialog() : Vector() {

endState = 0;

}

void Dialog::getEvent(tEvent& event) {

string opInt = "+-szqm";

string s;

string param;

char code;

cout << '>';

cin >> s;

code = s[0];

if (opInt.find(code) >= 0) {

event.eventWhat = eventMessage;

switch (code) {

case 'm':

event.command = c\_makeGroup;

break;

case '+':

event.command = c\_addObj;

break;

case '-':

event.command = c\_delObj;

break;

case 's':

event.command = c\_showGroup;

break;

case 'z':

event.command = c\_getAttr;

break;

case 'q':

event.command = c\_quit;

break;

}

if (s.length() > 1) {

param = s.substr(1, s.length() - 1);

int A = atoi(param.c\_str());

event.param = A;

}

}

else event.eventWhat = eventNothing;

}

int Dialog::execute() {

tEvent event;

do {

endState = 0;

getEvent(event);

handleEvent(event);

} while (!valid());

return endState;

}

int Dialog::valid() {

if (endState == 0) return 0;

else return 1;

}

void Dialog::clearEvent(tEvent& event) {

event.eventWhat = eventNothing;

}

void Dialog::endExec() {

endState = 0;

}

void Dialog::handleEvent(tEvent& event) {

if (event.eventWhat == eventMessage) {

switch (event.command) {

case c\_makeGroup:

size = event.param;

vector = new Object \* [size];

curr = 0;

clearEvent(event);

break;

case c\_addObj:

add();

clearEvent(event);

break;

case c\_delObj:

del();

break;

case c\_showGroup:

show();

clearEvent(event);

break;

case c\_quit:

endExec();

clearEvent(event);

break;

default:

Vector::handleEvent(event);

}

}

}

Файл Vector.h:

#include "Magazin.h"

#include "Print.h"

#pragma once

class Vector {

protected:

Object\*\* vector;

int curr;

int size;

public:

Vector();

Vector(int);

~Vector();

void handleEvent(const tEvent& e) {

if (e.eventWhat == eventMessage) {

Object\*\* p = vector;

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

(\*p)->handleEvent(e);

p++;

}

}

}

void show();

void add();

void del();

void add(Object\*);

Object& operator[](int);

friend ostream& operator<<(ostream&, const Vector&);

friend istream& operator>>(istream&, Vector&);

};

Vector::Vector() {

vector = 0;

curr = 0;

size = 0;

}

Vector::Vector(int size) {

this->size = size;

curr = 0;

vector = new Object \* [size];

}

Vector::~Vector() {

if (vector != 0) {

delete vector;

}

}

void Vector::show() {

cout << \*this;

}

void Vector::add() {

if (curr < size) {

cout << "1.Print, 2.Magazine" << endl;

int m;

cin >> m;

if (m == 1) {

Print\* p = new Print;

cout << "Enter name and author" << endl;

p->input();

add(p);

}

else if (m == 2) {

Magazin\* p = new Magazin;

cout << "Enter name, author and pages quantity" << endl;

p->input();

add(p);

}

}

else {

cout << "add(): size overflow" << endl;

}

}

void Vector::add(Object\* o) {

if (curr < size) {

vector[curr] = o;

curr++;

}

else throw "awfaw";

}

void Vector::del() {

if (curr = 0) return;

delete vector[curr];

curr--;

}

Object& Vector::operator[](int pos) {

return \*vector[pos];

}

ostream& operator<<(ostream& out, const Vector& v) {

for (int i = 0; i < v.curr; i++) {

out << "-----" << i + 1 << "-----" << endl;

v.vector[i]->output();

}

return out;

}

istream& operator>>(istream& in, Vector& v) {

for (int i = 0; i < v.curr; i++) {

v.vector[i]->input();

}

return in;

}

Файл Print.cpp:

#include "Print.h"

using namespace std;

Print::Print() {

name = "";

author = "";

}

Print::Print(string n, string a) : name(n), author(a) {}

Print::Print(const Print& t) {

name = t.name;

author = t.author;

}

string Print::getName() {

return name;

}

string Print::getAuthor() {

return author;

}

void Print::setName(string n) {

name = n;

}

void Print::setAuthor(string a) {

author = a;

}

Print& Print::operator=(const Print& t) {

name = t.name;

author = t.author;

return \*this;

}

ostream& operator<<(ostream& out, const Print& p) {

out << p.name << ": " << p.author << ": ";

return out;

}

istream& operator>>(istream& in, Print& p) {

in >> p.name >> p.author;

return in;

}

Файл Magazin.cpp:

#include "Magazin.h"

#include <iostream>

using namespace std;

Magazin::Magazin(): Print::Print() {

pages = 0;

}

Magazin::Magazin(string a, string b, int p): Print::Print(a, b) {

pages = p;

}

Magazin::Magazin(const Magazin& m): Print::Print(m) {

pages = m.pages;

}

void Magazin::setPages(int p) {

pages = p;

}

int Magazin::getPages() {

return pages;

}

Magazin& Magazin::operator=(const Magazin& m) {

Print::operator=(m);

pages = m.pages;

return \*this;

}

ostream& operator<<(ostream& out, const Magazin& m) {

out << static\_cast<Print>(m) << m.pages;

return out;

}

istream& operator>>(istream& in, Magazin& m) {

in >> static\_cast<Print&>(m) >> m.pages;

return in;

}

Файл Event.h:

#pragma once

const int eventNothing = 0;

const int eventMessage = 100;

const int c\_addObj = 1;

const int c\_delObj = 2;

const int c\_getAttr = 3;

const int c\_showGroup = 4;

const int c\_makeGroup = 6;

const int c\_quit = 101;

struct tEvent {

int eventWhat;

union {

int command;

struct {

int message;

int param;

};

};

};

Файл main.cpp:

#include <iostream>

#include "Magazin.h"

#include "Print.h"

#include "Dialog.h"

#include <Windows.h>

using namespace std;

int main() {

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

cout << "m + число - создать группу" << endl;

cout << "+ добавить элемент в группу" << endl;

cout << "- удалить элемент из группы" << endl;

cout << "s вывести информацию об элементах группы" << endl;

cout << "z вывести информацию об названиях группы" << endl;

cout << "q конец" << endl;

Dialog D;

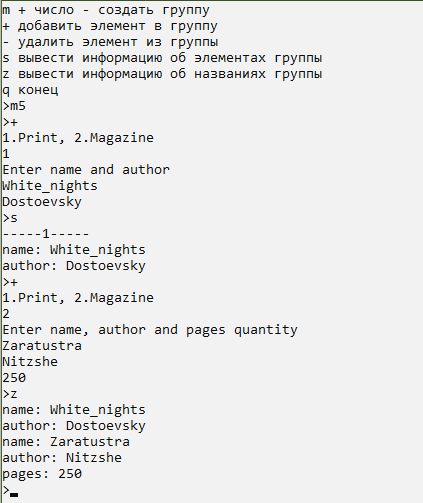
D.execute();

return 0;

}

**Вывод программы**

Ниже представлен вывод программы на консоль.

****

**Контрольные вопросы**

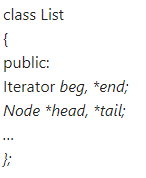
1. Что такое класс-группа? Привести примеры таких классов.

Ответ: класс-группа — объект, в который включены другие объекты.

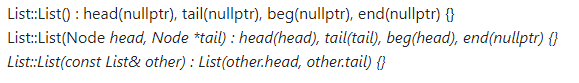
Пример: двумерный массив (вектор векторов)

1. Привести пример описания класса-группы Список (List).

Ответ:

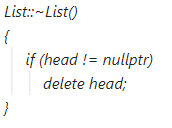
1. Привести пример конструктора (с параметром, без параметров, копирования) для класса-группы Список.

Ответ:



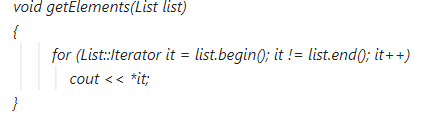
1. Привести пример деструктора для класса-группы Список.

Ответ:



1. Привести пример метода для просмотра элементов для класса-группы Список.

Ответ:



1. Какой вид иерархии дает группа?

Ответ: Иерархию объектов

1. Почему во главе иерархии классов, содержащихся в группе объектов должен находится абстрактный класс?

Ответ: Во главе иерархии классов должен находится абстрактный класс для удобного расширения кода

1. Что такое событие? Для чего используются события?

Ответ: Событие — информация, полученная в ходе действий пользователя.

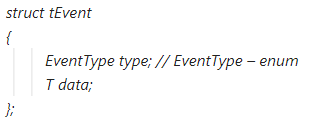
События используют для взаимодействия пользователя с интерфейсом графического приложения.

1. Какие характеристики должно иметь событие-сообщение?

Ответ: Событие-сообщение должно иметь код сообщения.

1. Привести пример структуры, описывающей событие.

Ответ:



1. Задана структура события:

*struct TEvent*

*{*

*int what;*

*union*

*{*

*MouseEventType mouse;*

*KeyDownEvent keyDown;*

*MessageEvent message;*

*}*

*};*

Какие значения, и в каких случаях присваиваются полю *what*?

Ответ: Полю what присваиваются значения пришло ли событие.

1. В: Задана структура события:

*struct TEvent*

*{*

*int what;*

*union*

*{*

*int command;*

*struct*

*{*

*int message;*

*int a;*

*};*

*}*

*};*

Какие значения, и в каких случаях присваиваются полю *command*?

Ответ: Полю command присваиваются значения типа события в случае требуеомого what

1. Задана структура события:

*struct TEvent*

*{*

*int what;*

*union*

*{*

*int command;*

*struct*

*{*

*int message;*

*int a;*

*};*

*}*

*};*

Для чего используются поля a и message?

Ответ: message – данные

1. Какие методы необходимы для организации обработки сообщений?

Ответ: метод получения события и метод обработчик сообщений

1. Какой вид имеет главный цикл обработки событий-сообщений?

Ответ: Цикл с флажком

1. Какую функцию выполняет метод ClearEvent()? Каким образом?

Ответ: устанавливает значения по умолчанию для данного экземпляра

1. Какую функцию выполняет метод HandleEvent ()?Каким образом?

Ответ: Обрабатывает событие

1. Какую функцию выполняет метод GetEvent ()?

Ответ: Получает новое событие

1. Для чего используется поле End State? Какой класс (объект) содержит это поле?

Ответ: endState содержит класс Dialog для отображения завершения обработки

1. Для чего используется функция Valid()?

Ответ: Проверка событие на соответствие обрабатываемым событиям