Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования  
«Пермский национальный исследовательский политехнический университет»

(ПНИПУ)

**Факультет:** электротехнический

**Направление:** Промышленная мехатроника и робототехника

ОТЧЁТ

По лабораторной работе №8

«Классы»

Семестр 1

Выполнил:

студент группы ПРТ-21-1б

Поздняков Е.С.

Пермь 2022

**Задание №1**

**Постановка задачи**

Базовый класс:

ПЕЧАТНОЕ ИЗДАНИЕ(PRINT)

Название – string

Автор – string

Производный класс

ЖУРНАЛ (MAGAZINE)

Количество страниц – int

Группа – Список (List)

Команды:

- создать группу (формат команды m: количество элементов группы)

- добавить элемент в группу (формат команды +)

- удалить элемент из группы (формат команды -)

- Вывести информацию об элементах группы (формат команды s)

- вывести информацию о названиях всех элементов группы (формат команды z)

- конец работы (формат команды q)

**Анализ**

1. Объявляем класс а с параметром размера множества. Пользователь вводит значения элементов множества.
2. Объявляем класс a1 из 5 элементов равных 7. Объявляем класс c и инициализируем его значением разности двух классов.
3. При помощи класса Iterator осуществляется вывод значений итератора i. При помощи перегруженных операторов инкремента и декремента осуществляется изменение значения i.

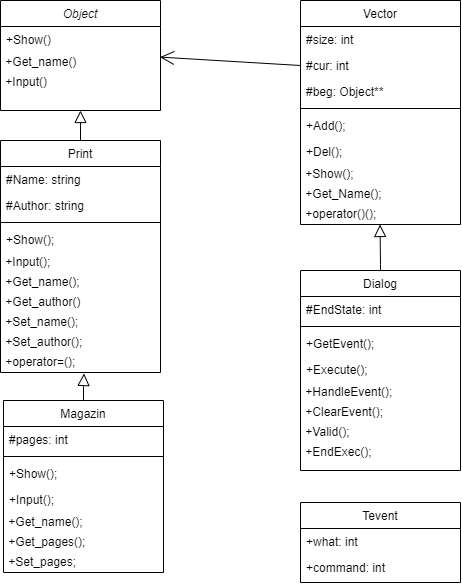
**Используемые переменные**

Plenty a – первое множество;

Plenty a1 – второе множество;

int с – значение элемента множества;

**UML-диаграмма**



**Код C++**

Основной файл:

#include <iostream>

#include "Event.h"

using namespace std;

class Object

{

public:

Object() {};

virtual void Show() = 0;

virtual void Input() = 0;

~Object() {};

virtual string GetName() = 0;

};

class Print : public Object

{

protected:

string name;

string author;

public:

Print() : name("-"), author("-") {};

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

Print (const Print& p)

{

name = p.name;

author = p.author;

}

void SetName(string n)

{

name = n;

}

void SetAuthor(string a)

{

author = a;

}

string GetName(){return name;}

string GetAuthor(){return author;}

Print& operator=(const Print&p)

{

name = p.name;

author = p.author;

return \*this;

}

void Show()

{

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

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

}

void Input()

{

cout << "Name = "; cin >> name;

cout << "Author = "; cin >> author;

}

};

class Magazine : public Print

{

protected:

int pages;

public:

Magazine() : Print()

{

pages = 0;

}

Magazine(string n, string a, int p) : Print(n, a)

{

pages = p;

}

Magazine(const Magazine& m)

{

name = m.name;

author = m.author;

pages = m.pages;

}

void SetPages(int p)

{

pages = p;

}

string GetName(){return name;}

string GetAuthor(){return author;}

Magazine& operator=(const Magazine& m)

{

name = m.name;

author = m.author;

pages = m.pages;

return \*this;

}

void Show()

{

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

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

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

}

void Input()

{

cout << "Name = "; cin >> name;

cout << "Author = "; cin >> author;

cout << "Pages = "; cin >> pages;

}

~Magazine(){};

};

class List

{

protected:

Object\*\* beg;

int size;

int cur;

public:

List()

{

beg = 0;

size = 0;

cur = 0;

}

List(int n)

{

beg = new Object\*[n];

cur = 0;

size = n;

}

void GetName()

{

Object\*\* p = beg;

for(int i = 0; i < cur; i++)

{

(\*p)->GetName();

p++;

}

}

void Add()

{

Object\* p;

cout << "1. Print\n2. Magazine" << endl;

int y; cin >> y;

if (y == 1)

{

Print\* a = new Print;

a->Input();

p = a;

if (cur < size)

{

beg[cur] = p;

cur++;

}

}

else

if (y == 2)

{

Magazine\* b = new Magazine;

b->Input();

p = b;

if (cur < size)

{

beg[cur] = p;

cur++;

}

}

}

void Show()

{

if (cur == 0) cout << "Empty" << endl;

Object\*\*p = beg;

for (int i = 0; i < cur; i++)

{

(\*p)->Show();

p++;

}

cout << endl;

}

int operator() ()

{

return cur;

}

void Del()

{

if (cur == 0) return;

cur--;

}

};

class Dialog : public List

{

protected:

int EndState;

public:

Dialog(void);

~Dialog(){};

void GetEvent (TEvent &event);

void Execute();

void HandleEvent(TEvent& event);

void ClearEvent (TEvent& event);

bool Valid();

void EndExec();

};

Dialog::Dialog(void) : List()

{

EndState = 0;

}

void Dialog::GetEvent (TEvent &event)

{

string OpInt = "m+-szq";

string s;

char code;

cout << '>';

cin >> s; code = s[0];

if (OpInt.find(code) >= 0)

{

event.what = evMessage;

switch (code)

{

case 'm': event.command = cmMake; break;

case '+': event.command = cmAdd; break;

case '-': event.command = cmDel; break;

case 's': event.command = cmShow; break;

case 'z': event.command = cmGet; break;

case 'q': event.command = cmQuit; break;

}

}

else event.what = evNothing;

}

void Dialog::Execute()

{

TEvent event;

do

{

EndState = 0;

GetEvent(event);

HandleEvent(event);

} while (Valid());

}

void Dialog::HandleEvent(TEvent& event)

{

if (event.what == evMessage)

{

switch(event.command)

{

case cmMake:

cout << "Input size: ";

cin >> size;

beg = new Object\*[size];

cur = 0;

ClearEvent(event); break;

case cmAdd: Add();

ClearEvent(event); break;

case cmDel:Del();

ClearEvent(event); break;

case cmShow:Show();

ClearEvent(event); break;

case cmQuit: EndExec();

ClearEvent(event); break;

case cmGet: GetName();

ClearEvent(event); break;

};

};

}

void Dialog::ClearEvent (TEvent& event)

{

event.what = evNothing;

}

bool Dialog::Valid()

{

return EndState == 0;

}

void Dialog::EndExec()

{

EndState = 1;

}

int main()

{

setlocale(0, "");

cout << "m: make group\n+: Add elem\n-: Delete elem\ns: info\nz: names\nq: bb" << endl;

Dialog D;

D.Execute();

return 0;

}

Event.h

#pragma once

const int evNothing = 0; // пустое событие

const int evMessage = 100; // непустое событие

const int cmAdd = 1; // добавить объект в группу

const int cmDel = 2; // удалить объект из группы

const int cmGet = 3; // вывести атрибут всех объектов

const int cmShow = 4; // вывести всю группу

const int cmMake = 6; // создать группу

const int cmQuit = 101; // выход

struct TEvent

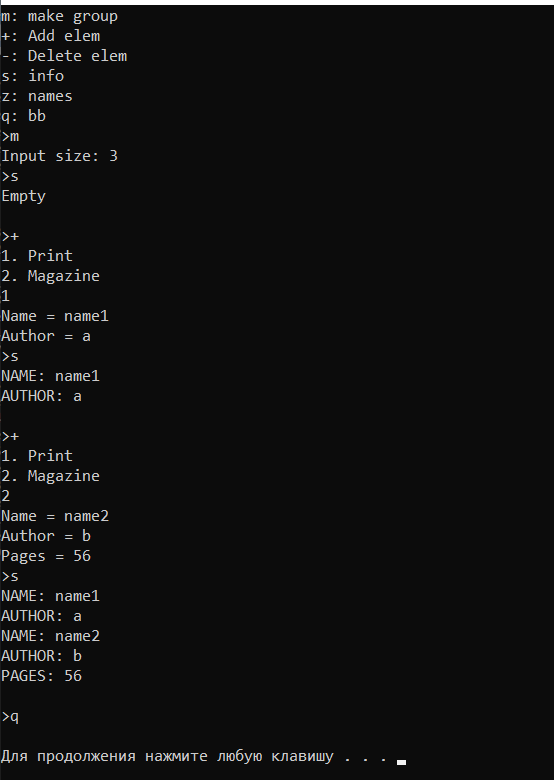
{

int what; // тип события

int command;

};

**Скриншот результата**

****

**Анализ результатов**

Пользователь вводит значения элементов первого множества. Из элементов второго множества вычитаются элементы первого множества и выводятся на экран.

**Контрольные вопросы**

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

Класс-группа – это объект, в который включены другие объекты. Например, иерархическая система университета (факультет, кафедра, группа)

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

class List

{

protected:

Object\*\* beg;

int size;

int cur;};

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

List()

{

beg = 0;

size = 0;

cur = 0;

}

List(int n)

{

beg = new Object\*[n];

cur = 0;

size = n;

}

List (const List& l)

{

cur = 0; size = l.size;

for (int i = 0; i < size; i++)

beg[i] = l.beg[i];

}

4. Привести пример деструктора для класса-группы Список.

~List()

{

delete [] beg;

}

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

void Show()

{

if (cur == 0) cout << "Empty" << endl;

Object\*\*p = beg;

for (int i = 0; i < cur; i++)

{

(\*p)->Show();

p++;

}

cout << endl;

}

6. Какой вид иерархии дает группа?

Иерархия объектов (иерархия типа целое/часть), построенная на основе агрегации или наследования.

7. Почему во главе иерархии классов, содержащихся в группе объектов должен находиться абстрактный класс?

Потому что у классов могут быть разные поля, следовательно, манипулировать этими данными нужно в зависимости от того, с каким классом мы работаем в данный момент.

8. Что такое событие? Для чего используются события?

Событие – это объект, отдельные поля которого характеризуют те или иные свойства передаваемой информации. События управляют действиями.