Министерство высшего образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

Лабораторная работа №8

Программа, управляемая событиями

Выполнил

Студент группы РИС-22-2б

Вековшинин Д. А.

Проверила

Доцент кафедры ИТАС

Полякова О. А.

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.

7. Написать тестирующую программу.

8. Нарисовать диаграмму классов и диаграмму объектов.

Вариант 10: Базовый класс:

ПЕЧАТНОЕ\_ИЗДАНИЕ(PRINT)

Название– string

Автор – string

Производный класс

ЖУРНАЛ (MAGAZIN)

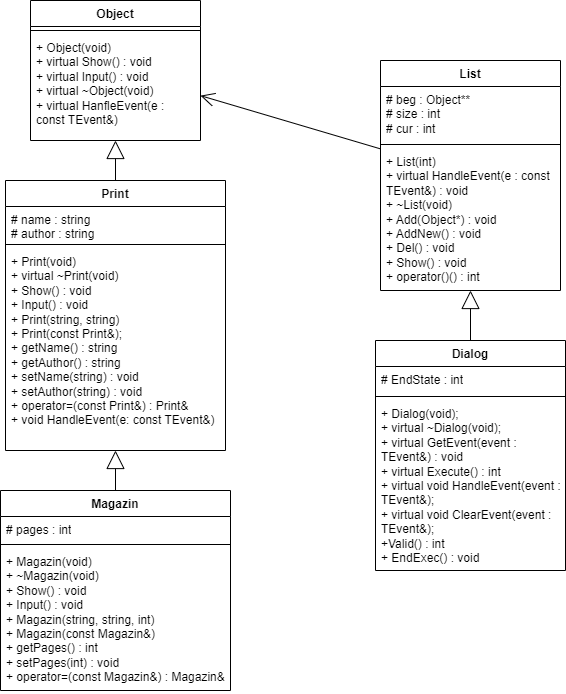
Количество страниц - int

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)

UML-диаграмма



Определение компонентных функций класса Print

//конструктор без параметров

Print::Print(void)

{

name = "";

author = "";

}

//деструктор

Print::~Print(void)

{}

//констрктор с параметрами

Print::Print(string N, string A)

{

name = N;

author = A;

}

//конструктор копирования

Print::Print(const Print& print)

{

name = print.name;

author = print.author;

}

//селекторы

void Print::setAuthor(string A)

{

author = A;

}

void Print::setName(string M)

{

name = M;

}

//оператор присваивания

Print& Print::operator=(const Print& p)

{

if (&p == this)return \*this;

name = p.name;

author = p.author;

return \*this;

}

//метод для просмотра атрибутов

void Print::Show()

{

cout << "\nNAME : " << name;

cout << "\nAUTHOR : " << author;

cout << "\n";

}

//метод для ввода значений атрибутов

void Print::Input()

{

cout << "\nName: "; cin >> name;

cout << "\nAuthor: "; cin >> author;

}

void Print::HandleEvent(const TEvent& e)

{

if (e.what == evMessage)//событие-сообщение

{

switch (e.command)

{

case cmGet:cout << "Name= " << getName() << endl;

break;

}

}

}

Определение компонентных функций класса Magazin

Magazin::Magazin(void) :Print()

{

name = "";

author = "";

pages = 0;

}

Magazin::~Magazin(void)

{

}

Magazin::Magazin(string N, string A, int P) : Print(N, A)

{

name = N;

author = A;

pages = P;

}

Magazin::Magazin(const Magazin& M)

{

name = M.name;

author = M.author;

pages = M.pages;

}

void Magazin::setPages(int P)

{

pages = P;

}

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

{

if (&m == this)return \*this;

name = m.name;

author = m.author;

pages = m.pages;

return \*this;

}

void Magazin::Show()

{

cout << "\nNAME : " << name;

cout << "\nAUTHOR : " << author;

cout << "\nPAGES : " << pages;

cout << "\n";

}

void Magazin::Input()

{

cout << "\nName: "; cin >> name;

cout << "\nAuthor: "; cin >> author;

cout << "\nPages: "; cin >> pages;

}

Функция main()

void main()

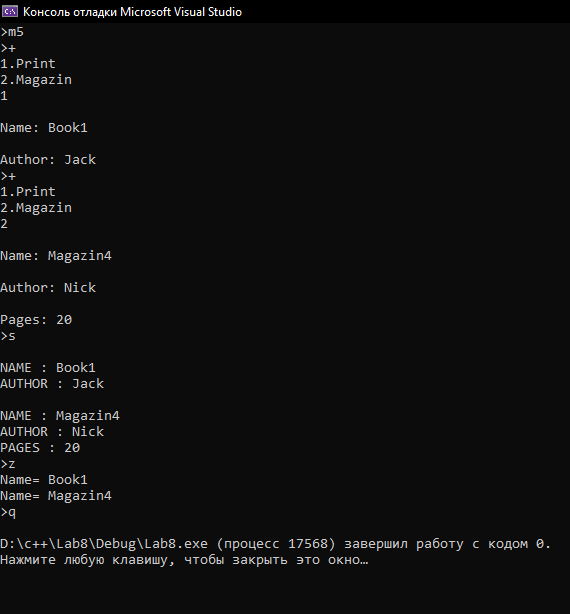
{

Dialog D;

D.Execute();

}

Результаты работы программы



Ответы на контрольные вопросы

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

Группа - это объект, в который включены другие объекты. Объекты, входящие в группу, называются элементами группы. Элементы группы, в свою очередь, могут быть группой.

Примеры групп:

1. Окно в интерактивной программе, которое владеет такими элементами, как поля ввода и редактирования данных, кнопки, списки выбора, диалоговые окна и т.д.

Примерами таких окон являются объекты классов, порожденных от абстрактного класса TGroup( TDeskTop, TWindow, TDialog) в иерархии классов библиотеки Turbo Vision, и объекты классов, порожденных от TWindowObject в иерархии классов библиотеки OWL.

2. Агрегат, состоящий из более мелких узлов.

3. Огород, состоящий из растений, системы полива и плана выращивания.

4. Некая организационная структура (например, ФАКУЛЬТЕТ, КАФЕДРА,

СТУДЕНЧЕСКАЯ ГРУППА).

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

template<typename T>

class List

{

public:

List();

~List();

List(const List& other);

void pop\_front();

void push\_back(T data);

void clear();

int GetSize() { return Size; }

T& operator[](const int index);

void push\_front(T data);

void insert(T data, int index);

void removeAt(int index);

void pop\_back();

private:

template<typename T>

class Node

{

public:

Node\* pNext;

T data;

Node(T data = T(), Node\* pNext = nullptr)

{

this->data = data;

this->pNext = pNext;

}

};

int Size;

Node<T>\* head;

};

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

List::List()

{

size = 0;

head = nullptr;

}

List::List(Node\* h, int n)

{

head = h;

size = n;

}

List::List(const List& l)

{

if (!other.head)

{

size = 0;

head = nullptr;

return;

}

head = new Node(\*other.head);

Node\* cur = head;

Size = 1;

for (Node\* t = other.head->pNext; t != nullptr; t = t->pNext) {

cur->pNext = new Node(\*t);

cur = cur->pNext;

++Size;

}

cur->pNext = nullptr;

}

1. Привести пример деструктора для класса-группы Список.

List::~List()

{

while (Size)

{

Node\* temp = head;s

head = head->pNext;

delete temp;

Size--;

}

}