Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8**

Дисциплина: «Основы алгоритмизации и программирования»

Тема : "Программа, управляемая событиями"

Выполнил работу:

Студент группы РИС-22-1Б

Батин В. В.

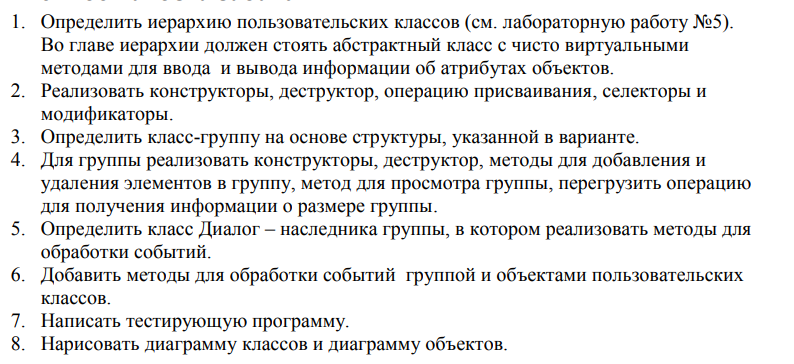
Проверила:

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

**Постановка задачи**



**Вариант 5**

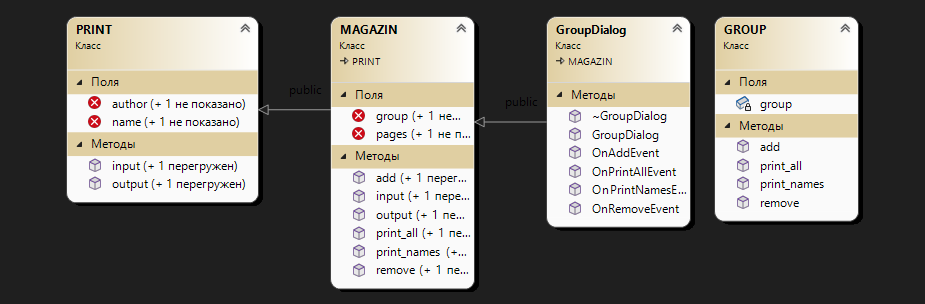
Базовый класс Print

Производный класс Magazine

Класс группа Group

Команды(+,-,s,q,m): добавить/удалить элемент, вывод на экран, выход, создать группу)

**Диаграмма классов**

****

**Код**

#include <iostream>

#include <string>

#include <vector>

class PRINT {

public:

std::string name;

std::string author;

public:

virtual void input() = 0;

virtual void output() = 0;

};

class GROUP {

private:

std::vector<PRINT\*> group;

public:

void add(PRINT\* p) {

group.push\_back(p);

}

void remove(int index) {

group.erase(group.begin() + index);

}

void print\_all() {

for (int i = 0; i < group.size(); i++) {

std::cout << "Element #" << i + 1 << std::endl;

group[i]->output();

std::cout << std::endl;

}

}

void print\_names() {

for (int i = 0; i < group.size(); i++) {

std::cout << group[i]->name << std::endl;

}

}

};

class MAGAZIN : public PRINT {

private:

int pages;

GROUP group;

public:

void input() override {

std::cout << "Enter name: ";

std::cin >> name;

std::cout << "Enter author: ";

std::cin >> author;

std::cout << "Enter number of pages: ";

std::cin >> pages;

}

void output() override {

std::cout << "Name: " << name << std::endl;

std::cout << "Author: " << author << std::endl;

std::cout << "Number of pages: " << pages << std::endl;

}

void add(PRINT\* p) {

group.add(p);

}

void remove(int index) {

group.remove(index);

}

void print\_all() {

group.print\_all();

}

void print\_names() {

group.print\_names();

}

};

class GroupDialog : public MAGAZIN {

public:

GroupDialog() {}

~GroupDialog() {}

void OnAddEvent() {

PRINT\* p = new MAGAZIN();

p->input();

add(p);

}

void OnRemoveEvent(int index) {

remove(index);

}

void OnPrintAllEvent() {

print\_all();

}

void OnPrintNamesEvent() {

print\_names();

}

};

int main() {

GroupDialog\* group\_dialog = new GroupDialog();

char command = ' ';

int n = 0;

while (command != 'q') {

std::cout << "Enter command: ";

std::cin >> command;

switch (command) {

case 'm':

std::cout << "Enter group size: ";

std::cin >> n;

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

PRINT\* p = new MAGAZIN();

p->input();

group\_dialog->add(p);

}

break;

case '+':

group\_dialog->OnAddEvent();

break;

case '-':

int index;

std::cout << "Enter index: ";

std::cin >> index;

group\_dialog->OnRemoveEvent(index - 1);

break;

case 's':

group\_dialog->OnPrintAllEvent();

break;

case 'z':

group\_dialog->OnPrintNamesEvent();

break;

case 'q':

std::cout << "Goodbye!" << std::endl;

break;

default:

std::cout << "Invalid command!" << std::endl;

break;

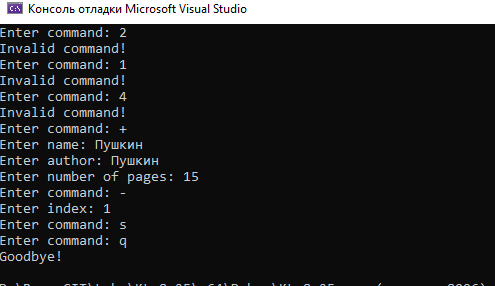
}

}

return 0;

}

**Результат работы проргаммы**

****

**Ответы на контрольные вопросы**

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

Класс-группа - это класс, который содержит в себе коллекцию объектов других классов и предоставляет методы для работы с этой коллекцией.

Например, можно создать класс-группу "Список студентов", который будет содержать объекты класса "Студент". Класс-группа будет предоставлять методы для добавления/удаления студентов, получения списка студентов и т.д.

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

template <typename T>

class List {

public:

// Конструкторы и деструктор

List();

List(const List<T>& other);

~List();

// Операторы

List<T>& operator=(const List<T>& other);

T& operator[](int index);

const T& operator[](int index) const;

// Методы

void insert(int index, const T& value);

void remove(int index);

int size() const;

bool isEmpty() const;

private:

// Внутренний класс узла списка

struct Node {

T data;

Node\* next;

Node(const T& d) : data(d), next(nullptr) {}

};

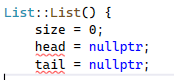
Node\* head;

int length;

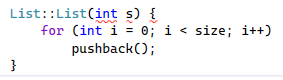
};

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

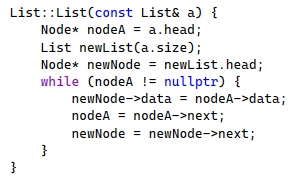
без параметров:



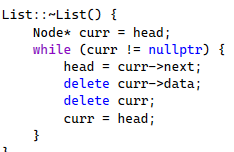
с параметром:



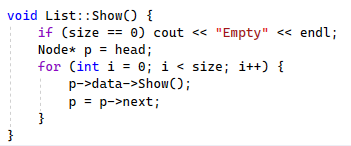
копирования:



4. Привести пример деструктора для класса-группы Список.



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



6. Какой вид иерархии дает группа?

Группа дает второй вид иерархии - иерархию объектов(иерархию типа целое/часть), построенную на основе агрегации, первый вид иерархия классов, построенная на основе наследования.

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

В иерархии классов, содержащихся в группе объектов, на вершине должен находиться абстрактный класс, так как он определяет общий интерфейс, который должен быть реализован всеми классами-наследниками. Это позволяет упростить дальнейшую работу с объектами классов-наследников и обеспечить единообразие в их использовании. Кроме того, такая организация позволяет обеспечить полиморфизм и использовать указатели на базовый абстрактный класс для работы с объектами различных классов-наследников через общий интерфейс.

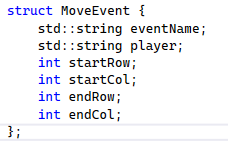
8. Что такое событие? Для чего используются события?

событие (event) представляет собой сигнал, сообщающий программе о каком-то действии, произошедшем в системе или приложении. События используются для организации обработки пользовательских действий, взаимодействия компонентов приложения, реализации асинхронной обработки и многих других задач.

9. Какие характеристики должно иметь событие-сообщение?

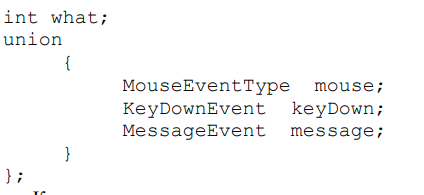
Объект TEvent состоит из двух частей. Первая (what) задает тип события, определяющий источник данного события. Вторая задает информацию, передаваемую с событием. Для разных типов событий содержание информации различно. Поле what может принимать следующие значения:  
• evNothing это пустое событие, которое означает, что ничего делать не надо. Полю what присваивается значение evNothing, когда событие обработано каким-либо объектом.  
• evMessage - событие-сообщение от объекта.

10. Привести пример структуры, описывающей событие.



11. Задана структура события



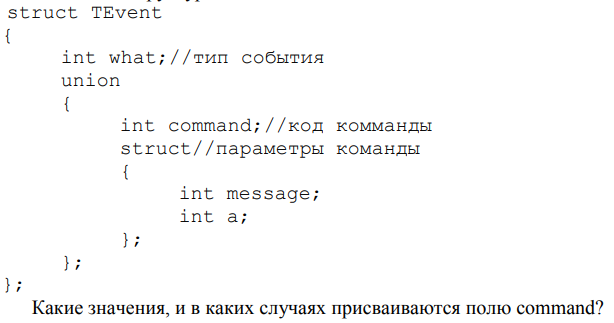


Какие значения, и в каких случаях присваиваются полю what?

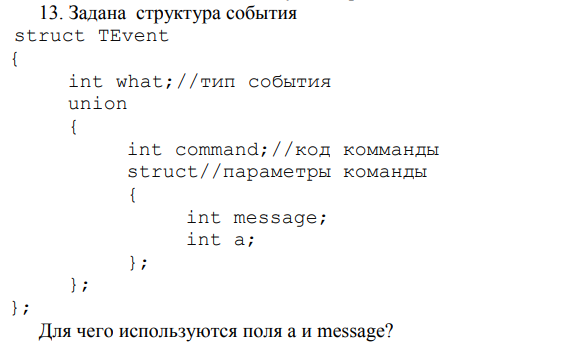
Поле what задает тип события, определяющий источник данного события. Оно может принимать следующие значения:

* evNothing - это пустое событие, которое означает, что ничего делать не надо. Полю what присваивается значение evNothing, когда событие обработано каким-либо объектом.
* evMessage - событие-сообщение от объекта (непустое событие).

12. Задана структура события



Полю command присваиваются коды различных определённых команд. При получении того или иного сообщения, поле command принимает одно из кодов команд .

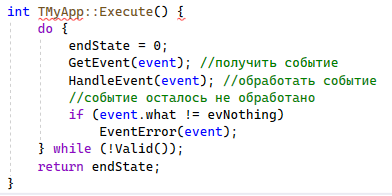


Полю command присваиваются коды различных определённых команд. При получении того или иного сообщения, поле command принимает одно из кодов команд

14. Какие методы необходимы для организации обработки сообщений?

* GetEvent – формирование события;
* Execute реализует главный цикл обработки событий. Он постоянно получает событие путем вызова GetEvent и обрабатывает их с помощью HandleEvent. Этот цикл завершается, когда поступит событие «конец».
* HandleEvent – обработчик событий. Обрабатывает каждое событие нужным для него образом. Если объект должен обрабатывать определенное событие (сообщение), то его метод HandleEvent должен распознавать это событие и реагировать на него должным образом. Событие может распознаваться, например, по коду команды (поле command).
* ClearEvent очищает событие, когда оно обработано, чтобы оно не обрабатывалось далее.
* Valid - проверяет, завершена ли работа.
* EndExec - завершение обработки событий (после вызова этого метода цикл обработки событий заканчивается).

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



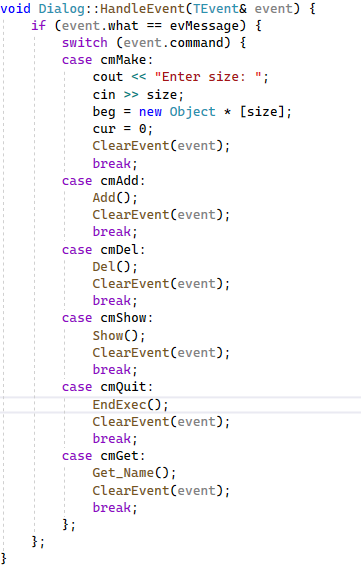
16. Какую функцию выполняет метод ClearEvent()? Каким образом?



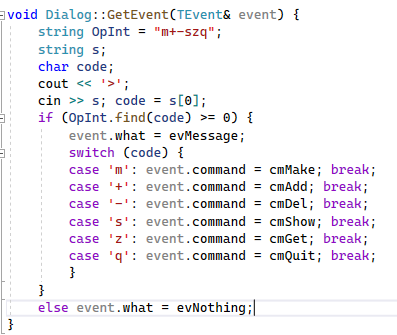
ClearEvent очищает событие, присваивая полю event.what значение evNothing.

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

Метод HandleEvent() выполняет обработку события-сообщения, которое было передано в качестве аргумента метода. Обычно он содержит логику для обработки конкретного типа событий.

  
18. Какую функцию выполняет метод GetEvent ()?

Метод GetEvent() используется для получения информации о следующем событии в очереди событий. Если в очереди нет ни одного события, метод блокируется до появления нового события. Если в очереди есть события, метод возвращает информацию о первом событии в очереди и удаляет его из очереди. Обычно этот метод вызывается в главном цикле обработки событий.



19. Для чего используется поле EndState? Какой класс (объект) содержит это поле?

Поле EndState используется для указания конечного состояния системы после выполнения события. Обычно это поле заполняется в обработчике события, который анализирует информацию, полученную из события, и принимает решение о том, как изменить состояние системы.

Поле EndState используется в классе Dialog и отвечает за состояние работы диалога (идёт/завершён).

20. Для чего используется функция Valid()?

Функция Valid() используется для проверки, является ли текущее состояние объекта допустимым. Она возвращает булево значение true, если текущее состояние объекта допустимо, и false в противном случае.