МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ПНИПУ)

Факультет: Электротехнический

Кафедра: «Информационные технологии и автоматизированные системы» (ИТАС)

Направление: Разработка информационных систем (РИС)

**О Т Ч Ё Т**   
**о лабораторной работе №8**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: «Программа, управляемая событиями»

Выполнил:

Студент группы РИС-24-3б

Суяргулов А.О.

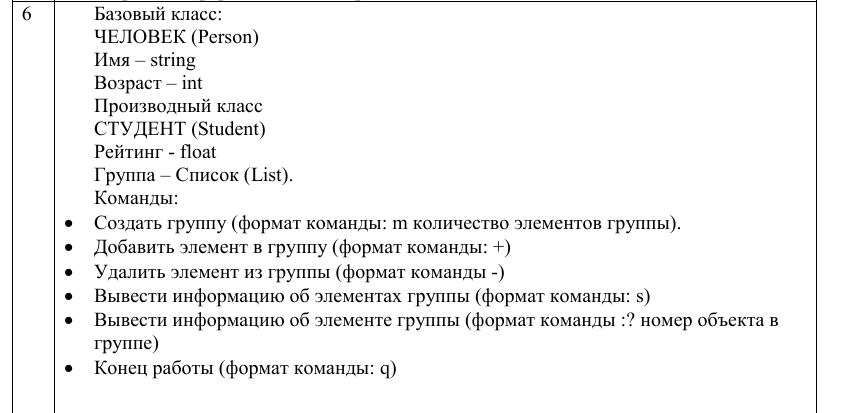
Проверил:

Доцент кафедры ИТАС

Полякова О.А.

Пермь, 2025

ВВЕДЕНИЕ



ОСНОВНАЯ ЧАСТЬ

**Диаграмма классов:**

**Код:**

**Main:**

#include <iostream>

#include <string>

#include <vector>

// Класс для человека

class Person {

public:

// Конструктор - вызывается при создании объекта Person

Person(std::string name, int age) {

this->name = name; // Сохраняем имя

this->age = age; // Сохраняем возраст

}

// Функция для вывода информации о человеке

void printInfo() {

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

std::cout << "Возраст: " << age << std::endl;

}

private:

std::string name; // Имя человека

int age; // Возраст человека

};

// Класс для студента (наследуется от Person)

class Student : public Person {

public:

// Конструктор для студента (вызывает конструктор Person)

Student(std::string name, int age, float rating) : Person(name, age) {

this->rating = rating; // Сохраняем рейтинг

}

// Переопределяем функцию printInfo, чтобы добавить рейтинг

void printInfo() {

Person::printInfo(); // Сначала выводим информацию о человеке

std::cout << "Рейтинг: " << rating << std::endl; // Добавляем рейтинг

}

private:

float rating; // Рейтинг студента

};

// Класс для группы людей (или студентов)

class Group {

public:

// Конструктор

Group() {

// Ничего не делаем при создании

}

// Функция для добавления человека (или студента) в группу

void addPerson(Person\* person) {

people.push\_back(person); // Добавляем указатель на человека в вектор

}

// Функция для вывода информации о всех в группе

void printGroupInfo() {

std::cout << "Информация о группе:" << std::endl;

// Проходим по всем людям в группе

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

std::cout << "Человек " << i + 1 << ":" << std::endl;

people[i]->printInfo(); // Вызываем функцию вывода информации для каждого человека

std::cout << std::endl;

}

}

private:

std::vector<Person\*> people; // Вектор для хранения указателей на людей

};

int main() {

// Создаем группу

setlocale(LC\_ALL, "RU");

Group myGroup;

// Создаем несколько людей и студентов

Person\* person1 = new Person("Иван", 20);

Student\* student1 = new Student("Петр", 22, 4.5);

Person\* person2 = new Person("Мария", 19);

// Добавляем их в группу

myGroup.addPerson(person1);

myGroup.addPerson(student1);

myGroup.addPerson(person2);

// Выводим информацию о группе

myGroup.printGroupInfo();

// Важно! Освобождаем память, чтобы не было утечек

delete person1;

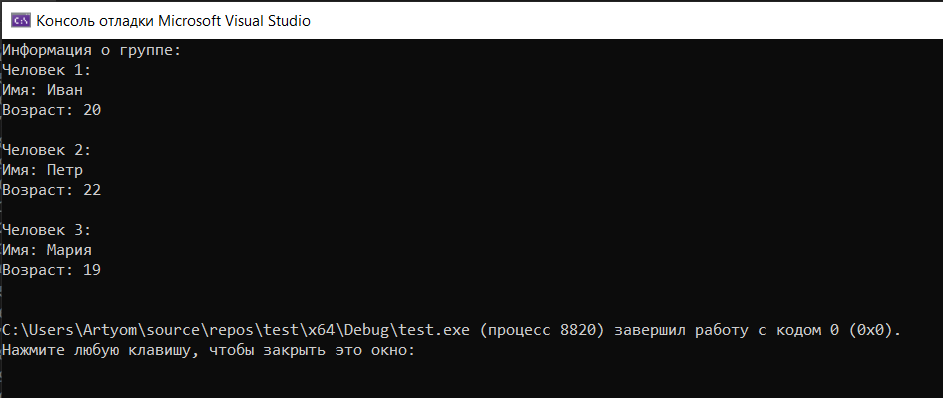
delete student1;

delete person2;

return 0;

}

**Результат работы кода:**

****

**Контрольные вопросы:**

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

Класс-группа – это класс, который содержит в себе коллекцию (группу) других объектов. Он предоставляет методы для управления этой коллекцией, такие как добавление, удаление, поиск, перебор элементов.

Примеры:

• Список (List)

• Массив (Array)

• Множество (Set)

• Дерево (Tree)

• Граф (Graph)

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

class List {

private:

Array elements; // Внутренний массив для хранения элементов

int size; // Текущий размер списка

int capacity; // Максимальная вместимость массива

public:

List(); // Конструктор

~List(); // Деструктор

void add(Object element); // Добавить элемент в конец списка

void insert(int index, Object element); // Вставить элемент по индексу

Object remove(int index); // Удалить элемент по индексу

Object get(int index); // Получить элемент по индексу

int getSize(); // Получить размер списка

bool isEmpty(); // Проверить, пуст ли список

}

```

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

// Конструктор без параметров

List::List() : size(0), capacity(10), elements(new Object[capacity]) {}

// Конструктор с параметром (начальный размер)

List::List(int initialCapacity) : size(0), capacity(initialCapacity), elements(new Object[capacity]) {}

// Конструктор копирования

List::List(const List& other) : size(other.size), capacity(other.capacity), elements(new Object[capacity]) {

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

elements[i] = other.elements[i]; // Предполагается, что Object имеет оператор присваивания

}

}

4. Привести пример деструктора для класса-группы Список.

```c++

List::~List() {

delete[] elements; // Освобождаем память, выделенную под массив

}

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

void List::printElements() {

std::cout << "Список: ";

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

std::cout << elements[i] << " "; // Предполагается, что Object имеет оператор <<

}

std::cout << std::endl;

}

6. Какой вид иерархии дает группа?

Группа обычно дает \*отношение "has-a"\* (имеет). То есть, класс группы \*имеет\* объекты другого класса. Это композиция или агрегация.

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

Если в группе могут храниться объекты разных классов (например, List<Shape> может содержать Circle, Square, Triangle), то во главе иерархии должен стоять абстрактный класс (например, Shape). Это позволяет:

\* Полиморфизм: Обращаться к объектам группы единообразно через указатели или ссылки на абстрактный базовый класс.

\* Гибкость: Легко добавлять новые типы объектов в группу, просто наследуя их от абстрактного класса.

\* Обобщенность: Определять общие свойства и методы для всех объектов в группе.

8. Что такое событие? Для чего используются события?

Событие – это сигнал о том, что произошло какое-то действие или изменение состояния в системе.

События используются для:

\* Реагирования на действия пользователя (например, нажатие кнопки, перемещение мыши).

\* Асинхронного взаимодействия между компонентами системы.

\* Обработки ошибок и исключительных ситуаций.

\* Уведомления других частей программы об изменениях данных.

9. Какие характеристики должно иметь событие-сообщение?

Событие-сообщение должно иметь:

\* Идентификатор типа события: (например, int what в вашей структуре). Позволяет определить, что за событие произошло.

\* Данные, относящиеся к событию: (например, координаты мыши при клике, код нажатой клавиши). Передаются через структуру или класс события.

\* Источник события: (не всегда явно, но часто необходимо знать, кто инициировал событие).

\* Время возникновения события: (иногда полезно для отладки и анализа).

10. Привести пример структуры, описывающей событие.

c++

struct TEvent {

int what; // Тип события

union {

MouseEventType mouse; // Данные для события мыши

KeyDownEvent keyDown; // Данные для события клавиатуры

MessageEvent message; // Данные для общего сообщения

};

};

11. Задана структура события (см. код в конце изображения)

Структура TEvent содержит:

• int what: Идентификатор типа события (например, EV\_MOUSE\_CLICK, EV\_KEY\_PRESSED).

• union: Объединение, которое позволяет хранить данные для разных типов событий в одном и том же месте в памяти. В зависимости от типа события, поле union содержит данные для мыши (MouseEventType mouse), клавиатуры (KeyDownEvent keyDown) или общего сообщения (MessageEvent message).

ChatGPT4 | Midjourney, [26.04.2025 14:58]

Вот ответы на оставшиеся вопросы из изображения:

12. Какие значения, и в каких случаях присваиваются полю command?

Полю command присваиваются целочисленные значения, представляющие код команды или тип операции, которую нужно выполнить. Конкретные значения зависят от контекста системы. Например:

• Команды меню: command = MENU\_OPEN, command = MENU\_SAVE.

• Сетевые команды: command = NETWORK\_CONNECT, command = NETWORK\_DISCONNECT.

• Внутренние команды: command = DO\_SOMETHING, command = CALCULATE\_RESULT.

Значение присваивается тогда, когда возникает событие, требующее выполнения определенной команды. Например, пользователь нажал на пункт меню "Открыть", тогда полю command присваивается значение MENU\_OPEN.

13. Для чего используются поля a и message?

• message: Используется для передачи текстового сообщения или кода сообщения, связанного с событием. Например, если событие - это вывод сообщения в лог, то message будет содержать текст этого сообщения. Если это системное сообщение, то message может быть кодом сообщения (например, WM\_CLOSE в Windows).

• a: Используется для передачи дополнительной целочисленной информации, связанной с событием. Например:

\* Код ошибки.

\* Идентификатор объекта, связанного с событием.

\* Дополнительные параметры команды.

14. Какие методы необходимы для организации обработки сообщений?

Необходимы как минимум следующие методы (или аналогичные):

• GetEvent(): Получает следующее событие из очереди событий (обычно это главный цикл обработки событий).

• HandleEvent(TEvent event): Обрабатывает событие, определяя его тип и вызывая соответствующий обработчик.

• ClearEvent(TEvent event): Освобождает ресурсы, связанные с событием (например, удаляет выделенную память, если это необходимо).

• PostEvent(TEvent event) / AddEvent(TEvent event): Добавляет событие в очередь событий.

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

```c++

while (!endState) { // Или пока не будет команды выхода

TEvent event = GetEvent(); // Получаем следующее событие

if (Valid(event)) { // Проверяем, что событие валидно

HandleEvent(event); // Обрабатываем событие

ClearEvent(event); // Освобождаем ресурсы, связанные с событием

} else {

// Обработка невалидного события (например, запись в лог)

}

}

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

ClearEvent() освобождает ресурсы, связанные с событием. Например:

• Если в событии содержатся указатели на динамически выделенную память (например, на строку сообщения), то эта память освобождается с помощью delete.

• Если событие связано с открытым файлом, то файл закрывается.

• Сбрасываются флаги, указывающие на то, что событие обработано.

Способ реализации зависит от структуры события и того, какие ресурсы оно использует.

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

HandleEvent() определяет тип события (обычно на основе поля what) и вызывает соответствующий обработчик для этого события.

Реализация:

• Switch-case конструкция: Самый простой вариант.

void HandleEvent(TEvent event) {

switch (event.what) {

case MENU\_OPEN: HandleMenuOpen(event); break;

case BUTTON\_CLICKED: HandleButtonClick(event); break;

case KEY\_PRESSED: HandleKeyPressed(event); break;

default: std::cout << "Неизвестное событие" << std::endl;

}

}

\* Таблица обработчиков (function pointers): Более гибкий вариант. Создается таблица, связывающая тип события с указателем на функцию-обработчик.

18. Какую функцию выполняет метод GetEvent()?

GetEvent() получает следующее событие из очереди событий. Обычно он ждет, пока в очереди не появится событие, и возвращает его. Если очередь пуста, он может блокировать поток или возвращать специальное значение (например, nullptr).

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

Поле EndState используется как флаг, указывающий на то, что главный цикл обработки событий должен завершиться. Когда EndState становится true, цикл while (!endState) завершается, и программа завершает свою работу.

Поле EndState обычно содержится в классе, управляющем главным циклом обработки событий (например, в классе Application, EventLoop, Dialog и т.д.).

20. Для чего используется функция Valid()?

Функция Valid() проверяет, является ли событие валидным. Это необходимо, чтобы предотвратить обработку неверных или поврежденных событий.

Примеры проверок, которые может выполнять Valid():

• Проверка типа события на соответствие допустимым значениям.

• Проверка данных, содержащихся в событии, на корректность (например, координаты мыши находятся в пределах экрана).

• Проверка, что событие не было сгенерировано вредоносным кодом.

• Проверка, что событие не дубликат и уже было обработано

**Ссылка на GitHub:**

https://github.com/Drazda1488/oop\_7lab