Министерство высшего образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ (ПНИПУ)

Электротехнический факультет

Дисциплина: основы алгоритмизации и программирования, 2 семестр

ОТЧЁТ

Тема: «Лабораторная работа №8»

Выполнил

Студент РИС-24-1б

Конькова С. С.

Проверил

Доцент кафедры ИТАС

Полякова О. А.

Пермь 2025**Содержание**

1. Постановка задачи (общая и конкретного варианта).
2. Описание класса.
3. Код программы.
4. Объяснение результатов работы программы.
5. Ответы на контрольные вопросы.

**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.

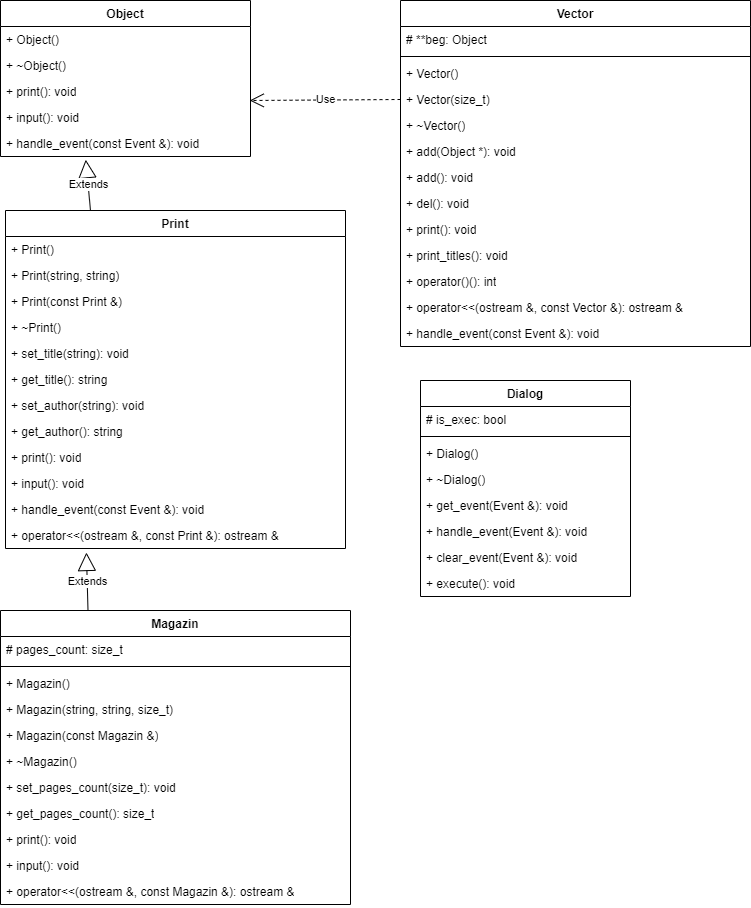
Команды:

1. Создать группу (формат команды: *m* количество элементов группы).
2. Добавить элемент в группу (формат команды: *+)*
3. Удалить элемент из группы (формат команды *-)*
4. Вывести информацию об элементах группы (формат команды: *s*)
5. Вывести информацию о названиях всех элементов группы (формат команды: *z*)
6. Конец работы (формат команды: *q*)

**Анализ**

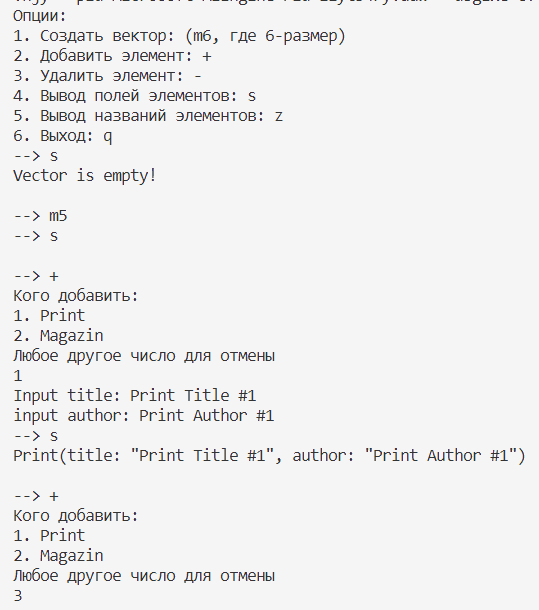
1. Добавление (+) Выбор: Person (имя, возраст) или Student (+ предмет, оценка). Объект сохраняется в вектор.
2. Удаление (-) Удаляет последний элемент, вызывая деструктор.
3. Просмотр (s) Для каждого объекта вызывается Show()
4. Средний возраст (z) Считает и выводит среднее значение.
5. Завершение (q)
6. Освобождает память, вызывая деструкторы всех объектов.

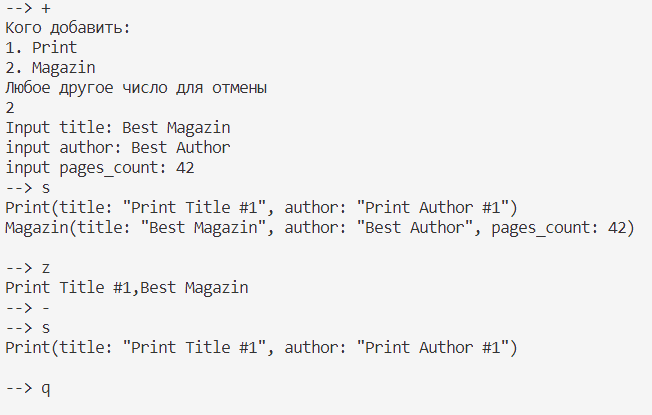
**UML Диаграмма**



**Код программы**

**Результат программы**





**Объяснение результатов.**

Программа выводит меню, и начинает обрабатывать ввод пользователя, пока пользователь не введет символ *q*. При первом введении символа *s* пользователь получил сообщение, об отсутствии списка. При последующих введениях этого же символа выводился список всех сохраненных данных. При введении символа *+* пользователю выводилось меню выбора, данные какого типа вводить, после и сам ввод данных. При введении символа *z* пользователю выводился список всех сохраненных заголовков. При введении символа *–* из списка сохраненных структур удалялась последняя структура. При введении символа *mN* с числом *N* создавался новый вектор.

**Ответы на контрольные вопросы**

1. Вопрос (В): Что такое класс-группа? Привести примеры таких классов.

Ответ (О): Класс-группа – это объект, в который включены другие объекты.

Пример (П):

*class TClass : public TGroup // класс-группа*

*{*

*TVal v;*

*TKey k;*

*};*

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

П:

*class List*

*{*

*public:*

*Iterator \*beg, \*end;*

*Node \*head, \*tail;*

*…*

*};*

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

П:

*List::List() : head(nullptr), tail(nullptr), beg(nullptr), end(nullptr) {}*

*List::List(Node \*head, Node \*tail) : head(head), tail(tail), beg(head), end(nullptr) {}*

*List::List(const List& other) : List(other.head, other.tail) {}*

1. В: Привести пример деструктора для класса-группы Список.

П:

*List::~List()*

*{*

*if (head != nullptr)*

*delete head;*

*}*

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

П:

*// Я ничерта не понимаю эти вопросы*

*void get\_elements(List list)*

*{*

*for (List::Iterator it = list.beg; it != list.end; it++)*

*\*it; // Получил доступ*

*}*

1. В: Какой вид иерархии дает группа?

О: Иерархия объектов.

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

О: Чтоб нельзя было создать его экземпляр / чтоб можно было определять его наследников.

1. В: Что такое событие? Для чего используются события?

О: Событие – набор информации. События используют для передачи информации.

1. В: Какие характеристики должно иметь событие-сообщение?

О: Оно должно иметь это самое сообщение.

1. В: Привести пример структуры, описывающей событие.

П:

*template<class T>*

*struct Event*

*{*

*EventType type; // EventType – enum*

*T data;*

*};*

*// И все, и больше никаких int a, в которые пытаются запихнуть строку, никаких int what вместо bool’а.*

1. В: Задана структура события:

*struct TEvent*

*{*

*int what;*

*union*

*{*

*MouseEventType mouse;*

*KeyDownEvent keyDown;*

*MessageEvent message;*

*}*

*};*

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

О: Это должен быть тип события, а не пришло/не пришло.

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*?

О: Это должен быть тип события, а не пришло/не пришло.

1. В: Задана структура события:

*struct TEvent*

*{*

*int what;*

*union*

*{*

*int command;*

*struct*

*{*

*int message;*

*int a;*

*};*

*}*

*};*

Для чего используются поля a и message?

О: *message* – данные, *a* – какой-то бредовый костыль без контекста.

П:

1. В: Какие методы необходимы для организации обработки сообщений?

О: Получение события, обработчики событий

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

О: Он не имеет вида, это просто цикл, получающий новые события, пока одно из не скажет остановиться.

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

О: Очищает ОДИН ЕДИНСТВЕННЫЙ ЭКЗЕМПЛЯР путем установки всех значений по умолчанию.

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

О: Обрабатывает событие так, как задал разработчик.

1. В: Какую функцию выполняет метод *GetEvent()*?

О: Получает новое событие. Тем образом, который предусмотрел разработчик, будь то ввод из консоли, из файла, нажатие кнопки…

1. В: Для чего используется поле EndState? Какой класс (объект) содержит это поле?

О: Костыль для определения этапа процесса обработки.

1. В: Для чего используется функция Valid()?

О: Проверка события на пригодность. Тоже костыль.