7ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа

Программа, управляемая событиями

Выполнил студент группы РИС-23-3Б

Мазунин М.А.

Проверила доцент кафедры ИТАС

О. А. Полякова

2024 г.

**1. Постановка задачи**

1. Определить шаблон класса-контейнера (см. лабораторную работу №6).
2. Реализовать конструкторы, деструктор, операции ввода-вывода, операцию присваивания.
3. Перегрузить операции, указанные в варианте.
4. Инстанцировать шаблон для стандартных типов данных (int, float, double).
5. Написать тестирующую программу, иллюстрирующую выполнение операций для контейнера, содержащего элементы стандартных типов данных.
6. Реализовать пользовательский класс (см. лабораторную работу №3).
7. Перегрузить для пользовательского класса операции ввода-вывода.
8. Перегрузить операции необходимые для выполнения операций контейнерного класса.
9. Инстанцировать шаблон для пользовательского класса.
10. Написать тестирующую программу, иллюстрирующую выполнение операций для контейнера, содержащего элементы пользовательского класса.

**2. Анализ задачи**

Класс TEvent - Класс, описывающий событие

Класс Object - Абстрактный класс, стоящий во главе иерархии наследования

Класс Print - Класс, описывающий печатное издание. Наследуется от Object

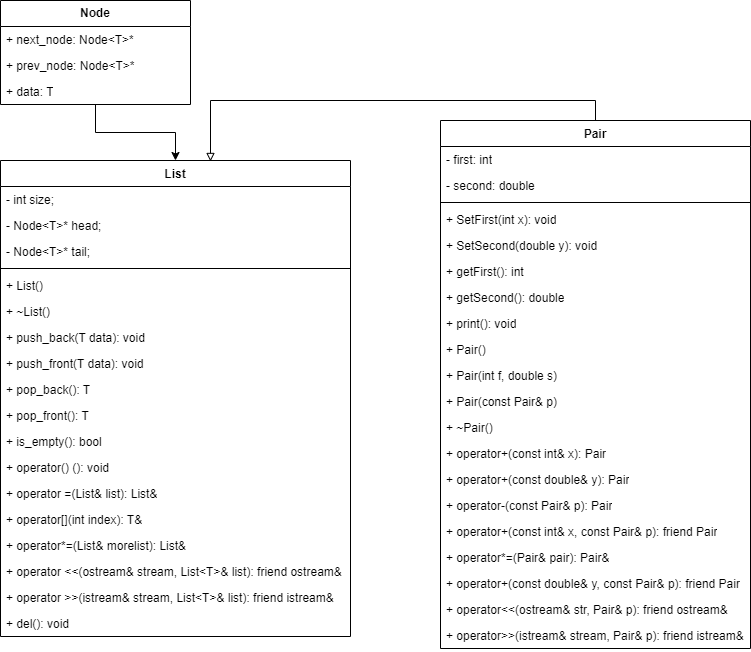
Класс Magazine - Класс, описывающий журнал. Наследуется от Print

Класс Vector - Класс-контейнер

Класс Dialog - Класс, обеспечивающий интерфейс программы. Наследуется от Vector

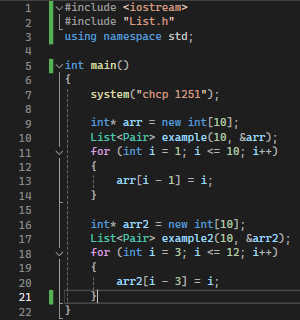
**3. Разбор работы алгоритма**

UML диаграмма



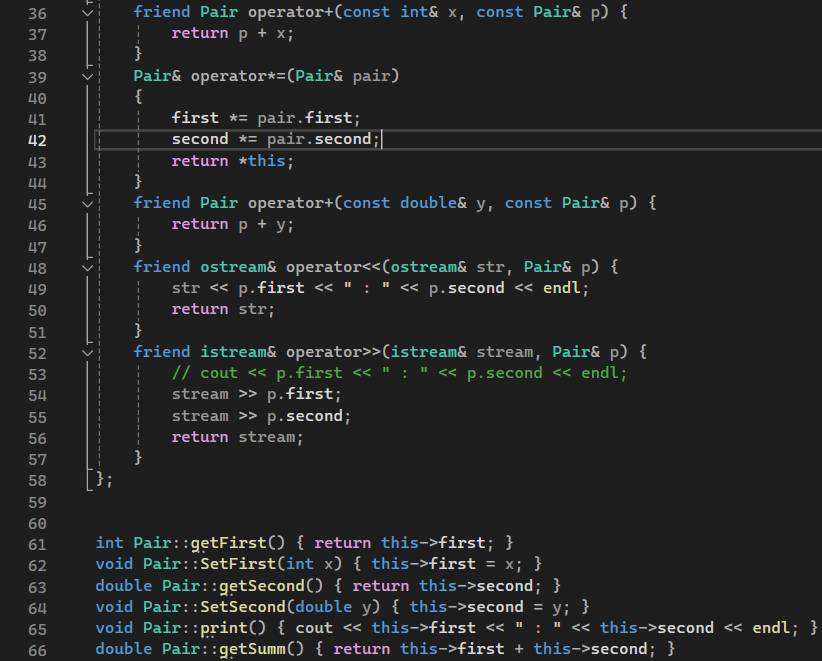
Код программы

Class\_7.cpp

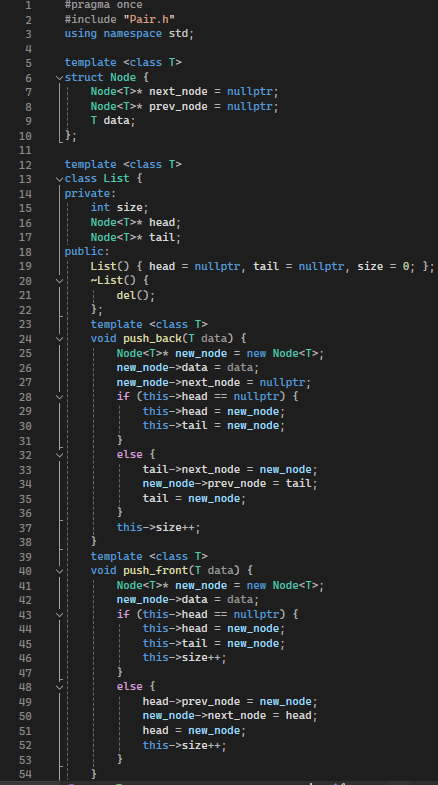


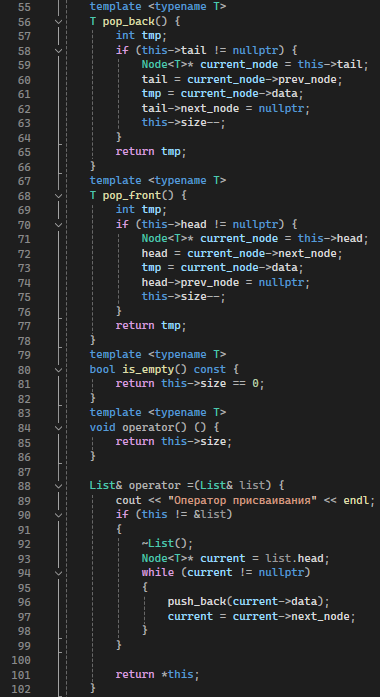
Pair.h

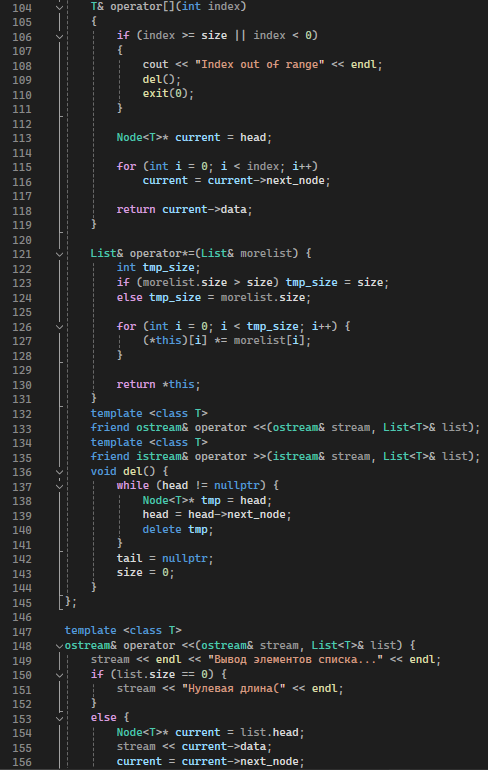


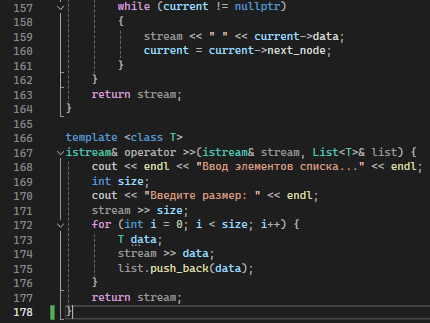


List.h









**4. Контрольные вопросы**

1. Шаблоны нужны для автоматической перегрузки функций под любой тип данных.

2. Перед функцией применяется конструкция template <typename T>, где T используется как переменная любого типа.

3. Перед функцией применяется конструкция template <class T>, где T используется как переменная любого типа.

4. Параметры шаблона функции – это переменные, которые будут использоваться для перегрузки.

5. Свойства параметра шаблона функции: имя параметра должно быть уникальным, список параметров не может быть пустым, в списке может быть несколько параметров.

6. Параметр шаблона записывается в треугольных скобках в формате “class имя” либо “typename имя”.

7. Параметризованные функции можно перегружать.

8. На основе параметризованного класса можно создавать перегруженные объекты. Каждый параметризованный класс должен иметь уникальное имя.

9. Все компонентные функции параметризованного класса являются параметризованными.

10. Дружественные функции, описанные в параметризованном классе, являются параметризованными.

11. Шаблоны классов могут содержать виртуальные компонентные функции.

12. template <class Type>

class MyClass {

Type value\_;

public:

MyClass(const T &value) : value\_(value) {};

void print();

};

template <class T>

void MyClass<T>::print() { std::cout << this->value\_; }

13. Инстанцирование шаблона – создание объекта параметризованного класса.

14. Генерирование определения класса по шаблону происходит на этапе вызова функции.

**5. Вывод**

Алгоритм выполняет поставленную задачу.

Данный проект можно найти на GitHub



