**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

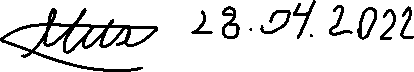
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Программирование на основе классов и шаблонов»

Отчет по лабораторной работе №5

«Множества»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-21Б |  | преподаватель каф. ИУ5 |
| Михалёв Я.М. |  | Козлов А.Д. |
| Подпись и дата: |  | Подпись и дата: |
|  |  |  |



Москва, 2022 г.

**Постановка задачи**

Создать класс с использованием динамической памяти для реализации множества строк. Класс с именем **MySet** включает следующие члены:

* **Члены - данные (private)**
  + **size** – количество элементов в множестве
  + **elements** – указатель на строку, содержащий адрес динамического массива строк
* **Конструкторы и деструктор**
  + Конструктор без параметров для создания пустого множества
  + Конструктор с одним строковым параметром для создания множества размером 1
  + Конструктор, копирующий множество
  + Деструктор
* **Методы доступа**
  + **Size** для возврата количества элементов в множестве
  + **IsElement,** который даёт **true**, если строка-параметр есть в множестве, иначе даёт **false**
* **Методы изменения**
  + **AddElement** для добавления строки в множество, если её там ещё нет
  + **DeleteElement** для удаления строки из множества, если она там есть
* **Операторы**
  + **[ ]** для возврата ссылки на элемент множества
  + **операторы присваивания =, -=, +=, \*=** где - означает разность, + **-** объединение и \* - пересечение. (См. примеры ниже.)

**Функции – не члены класса (не друзья класса!)**

* + Функция **печати** элементов множества - не более 5 элементов в строке
  + **Операторы +** (объединение), **-** (разность), **\*** (пересечение) и **==** (сравнение: истина, если элементы двух множеств совпадают).   
    Например:  
    {1, 4, 5, 6} + {1, 2, 3, 4} => {1, 2, 3, 4, 5, 6}  
    {1, 4, 5, 6} \* {1, 2, 3, 4} => {1, 4}   
    {1, 4, 5, 6} - {1, 2, 3, 4} => {5, 6}

Нельзя добавлять методы-члены класса в раздел **public** и данные-члены в раздел private. При желании можно добавить методы в раздел **private**.

**Разработка алгоритма**

**Класс MySet:**

Члены - данные:

* int size – коэффициент
* string\* elements;

Конструкторы деструктор:

* Конструктор без параметров
* Конструктор с параметрами const string a
* Конструктор копирования
* Деструктор

Перегруженные операторы

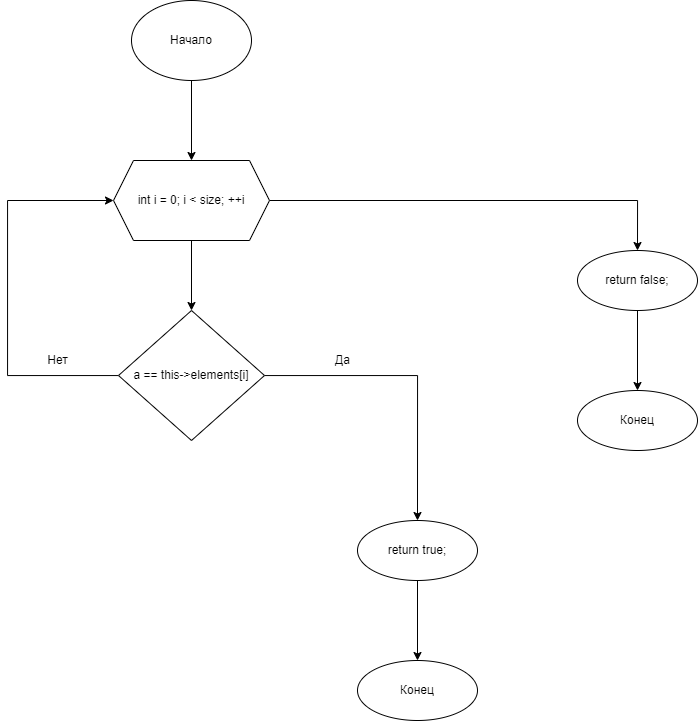
* operator =
* operator +=
* operator -=
* operator \*=
* operator +
* operator –
* operator =
* operator []
* operator ==
* operator 1=
* operator <<

Методы:

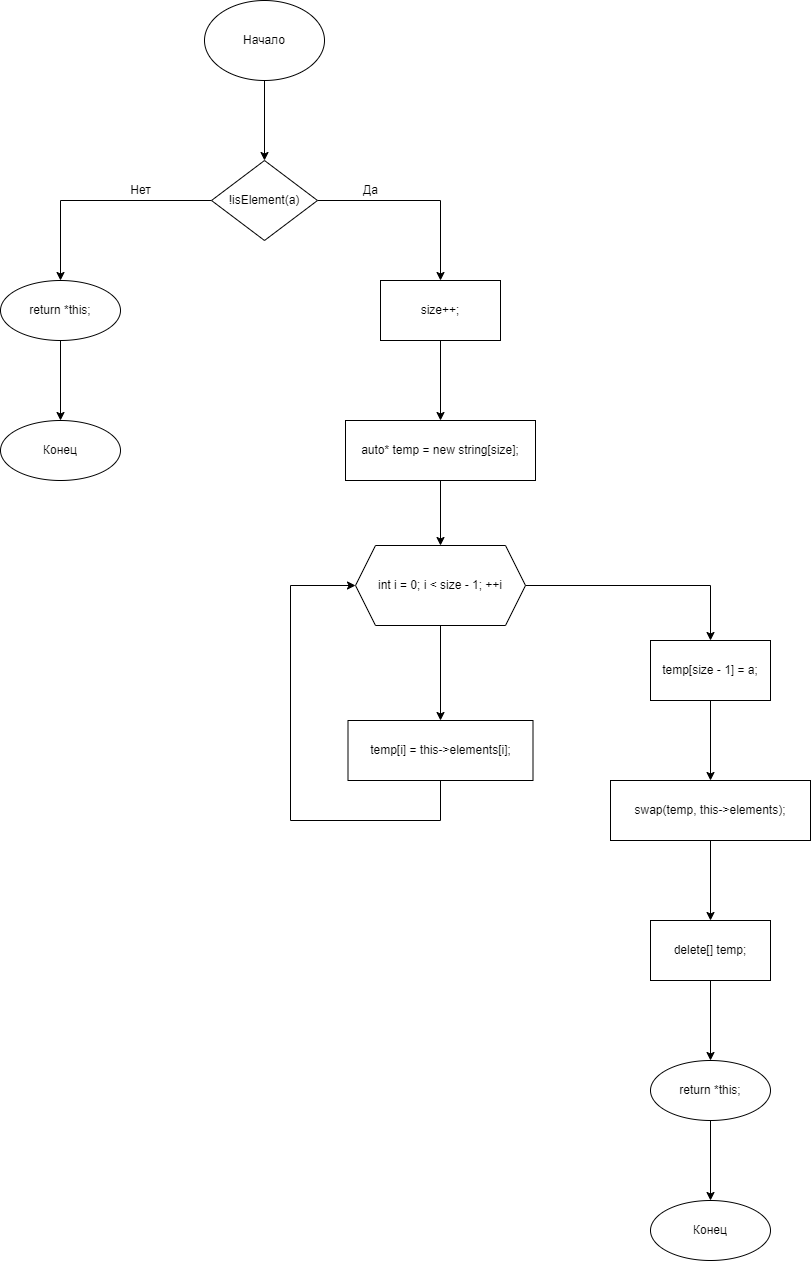
* MySet AddElement(string a) – добавляет элемент a
* MySet deleteElement(string a) – удаляет элемент a
* bool isElement(string a) – возвращает true – если элемент находится во множестве

**Схема алгоритма**

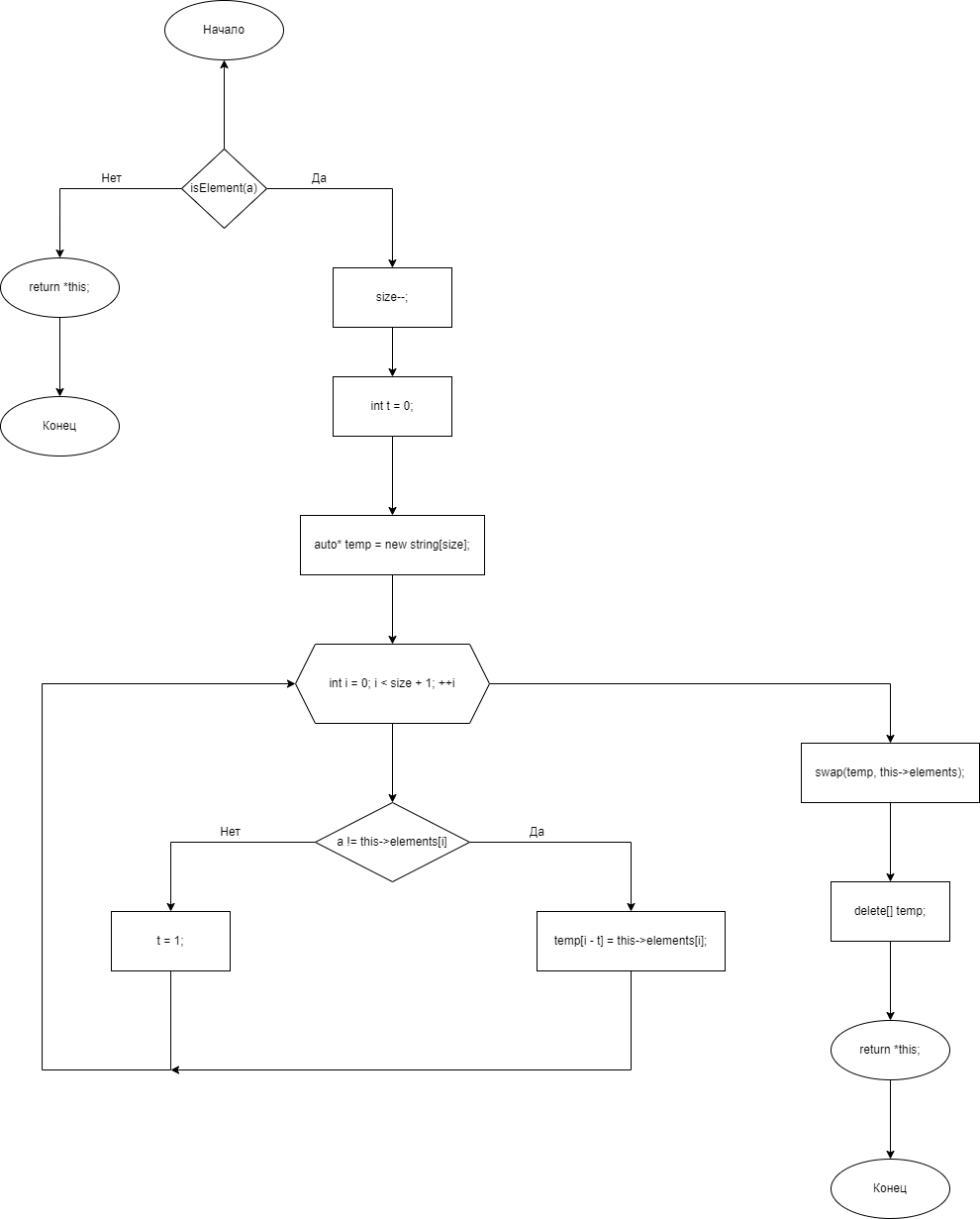
isElement



addElement



deleteElement



**Текст программы**

Main.cpp

#include <iostream>

#include <string>

#include <iomanip>

**using** **namespace** std**;**

class MySet

**{**

private**:**

int size**;**

string**\*** elements**;**

public**:**

MySet**()** **{**

**this->**size **=** 0**;**

**this->**elements **=** **nullptr;**

**}**

MySet**(**const string a**)** **{**

**this->**size **=** 1**;**

**this->**elements **=** **new** string**[**1**];**

**this->**elements**[**0**]** **=** a**;**

**}**

MySet**(**const MySet **&**mySet**)** **{**

**this->**size **=** mySet**.**size**;**

**this->**elements **=** **new** string**[**size**];**

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

**this->**elements**[**i**]** **=** mySet**.**elements**[**i**];**

**}**

**}**

**~**MySet**()** **{**

**delete[]** **this->**elements**;**

**}**

int Size**()** **{**

**return** size**;**

**}**

bool isElement**(**string a**)** **{**

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

**if** **(**a **==** **this->**elements**[**i**])** **{**

**return** **true;**

**}**

**}**

**return** **false;**

**}**

MySet addElement**(**string a**)** **{**

**if** **(!**isElement**(**a**))** **{**

size**++;**

auto**\*** temp **=** **new** string**[**size**];**

**for** **(**int i **=** 0**;** i **<** size **-** 1**;** **++**i**)** **{**

temp**[**i**]** **=** **this->**elements**[**i**];**

**}**

temp**[**size **-** 1**]** **=** a**;**

swap**(**temp**,** **this->**elements**);**

**delete[]** temp**;**

**}**

**return** **\*this;**

**}**

MySet deleteElement**(**string a**)** **{**

**if** **(**isElement**(**a**))** **{**

size**--;**

int t **=** 0**;**

auto**\*** temp **=** **new** string**[**size**];**

**for** **(**int i **=** 0**;** i **<** size **+** 1**;** **++**i**)** **{**

**if** **(**a **!=** **this->**elements**[**i**])** **{**

temp**[**i **-** t**]** **=** **this->**elements**[**i**];**

**}**

**else** **{**

t **=** 1**;**

**}**

**}**

swap**(**temp**,** **this->**elements**);**

**delete[]** temp**;**

**}**

**return** **\*this;**

**}**

string **operator[](**int i**)** **{**

**return** **this->**elements**[**i**];**

**}**

MySet **operator=(**MySet set**)** **{**

auto**\*** temp **=** **new** string**[**set**.**size**];**

size **=** set**.**size**;**

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

temp**[**i**]** **=** set**.**elements**[**i**];**

**}**

swap**(**temp**,** **this->**elements**);**

**delete[]** temp**;**

**return** **\*this;**

**}**

MySet **operator+=(**MySet set**)** **{**

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

addElement**(**set**.**elements**[**i**]);**

**}**

**return** **\*this;**

**}**

MySet **operator-=(**MySet set**)** **{**

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

deleteElement**(**set**[**i**]);**

**}**

**return** **\*this;**

**}**

MySet **operator\*=(**MySet set**)** **{**

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

**if** **(!**set**.**isElement**(**elements**[**i**]))** **{**

deleteElement**(**elements**[**i**]);**

**}**

**}**

**return** **\*this;**

**}**

friend bool **operator==(**MySet a**,** MySet b**)** **{**

**if** **(**a**.**Size**()** **!=** b**.**Size**())** **{**

**return** **false;**

**}**

**else** **{**

**for** **(**int i **=** 0**;** i **<** a**.**Size**();** **++**i**)** **{**

**if** **(!**a**.**isElement**(**b**[**i**]))** **{**

**return** **false;**

**}**

**}**

**return** **true;**

**}**

**}**

friend bool **operator!=(**MySet a**,** MySet b**)** **{**

**return** **!(**a **==** b**);**

**}**

friend MySet **operator+(**MySet a**,** MySet b**)** **{**

MySet temp**(**a**);**

temp **+=** b**;**

**return** temp**;**

**}**

friend MySet **operator-(**MySet a**,** MySet b**)** **{**

MySet temp**(**a**);**

temp **-=** b**;**

**return** temp**;**

**}**

friend MySet **operator\*(**MySet a**,** MySet b**)** **{**

MySet temp**(**a**);**

temp **\*=** b**;**

**return** temp**;**

**}**

friend ostream **&operator<<(**ostream**&** out**,** MySet set**)** **{**

**if** **(**set**.**Size**()** **!=** 0**)** **{**

out **<<** "{ "**;**

**if** **(**set**.**Size**()** **!=** 1**)** **{**

**for** **(**int i **=** 0**;** i **<** set**.**Size**()** **-** 1**;** **++**i**)** **{**

cout **<<** set**[**i**]** **<<** ", "**;**

**}**

**}**

cout **<<** set**[**set**.**Size**()** **-** 1**]** **<<** " }"**;**

**}**

**else** **{**

out **<<** "{ }"**;**

**}**

**return** out**;**

**}**

**};**

int main**()**

**{**

MySet a**,** b**(**"1"**),** c**(**"2"**),** d**(**b**),** e**(**b **+** c**),** f**(**d **\*** e**),** g**(**e **-** b**);**

cout **<<** a **<<** endl**;**

cout **<<** b **<<** endl**;**

cout **<<** c **<<** endl**;**

cout **<<** d **<<** endl**;**

cout **<<** e **<<** endl**;**

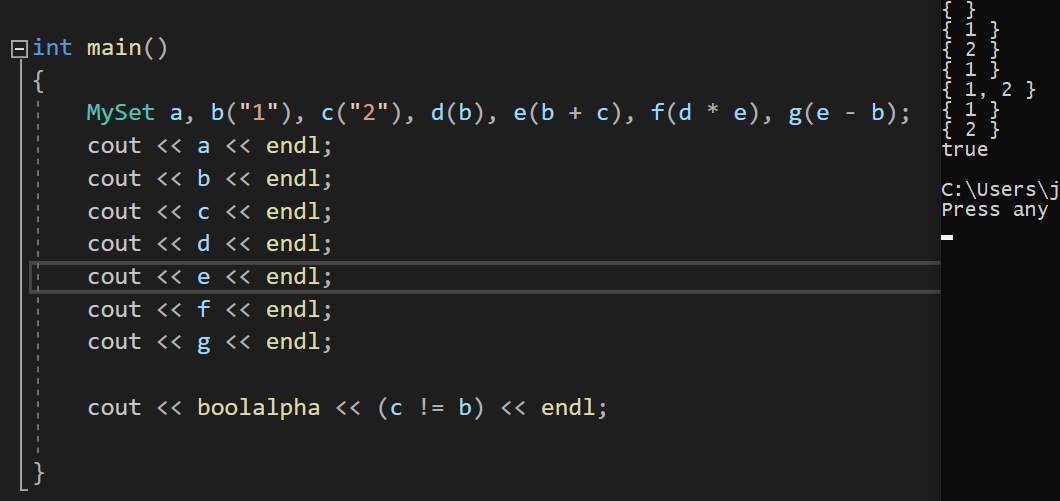
cout **<<** f **<<** endl**;**

cout **<<** g **<<** endl**;**

cout **<<** boolalpha **<<** **(**c **!=** b**)** **<<** endl**;**

**}**

**Анализ результатов**



**Вывод**

Я научился

* Работать с динамическими массивами
* Работать с конструктора и деконструкторами
* Перезагружать операторы