Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

Множественное наследование

Даны 8 классов, которые нумеруются от 1 до 8. Классы 2, 3, 4 и 5 наследованы от первого класса. Шестой класс от второго и третьего. Седьмой от четвертого и пятого. Восьмой от шестого и седьмого.

У каждого класса есть параметризированный конструктор с одним параметром строкового типа и закрытое свойство строкового типа для хранения наименования объекта класса. Значение данного свойства определяется в параметризированном конструкторе согласно шаблону:

```
«значение строкового параметра»_«номер класса»
```

У каждого класса есть метод в открытом разделе с одинаковым наименованием, который возвращает наименование объекта класса.

В реализации конструкторов со второго по восьмой класс, вызвать конструктор или конструкторы родительских классов. При вызове передать в качестве параметра выражение:

```
«параметр производного класса + «_» + «номер производного класса»
```

Например, для конструктора второго класса

```
cl_2 :: cl_2 ( string s_name ) : cl_1 ( s_name + “_2” )
```

В основной функции реализовать алгоритм:

1. Объявить один указатель на объект класса x.
2. Объявить переменную строкового типа.
3. Ввести значение строковой переменной. Вводимое значение является идентификатором.
4. Создать объект класса 8 посредством параметризированного конструктора, передав в качестве аргумента строковую переменную.

5. Адрес созданного объекта присвоить указателю на объект класса x.

6. Используя только указатель на объект класса x вывести имена всех объектов в составе объекта класса 8 и имя самого объекта класса 8. Вывод выполнить построчно, упорядочивая согласно возрастанию номеров класса. Наименования объектов первого класса вывести последовательно для производных объектов 2,3.4 и 5 класса.

Наследственность реализовать так, чтобы всего объектов было 10 и обеспечить вывод по аналогии приведенному примеру вывода.

1.1 Описание входных данных

Первая строка:

«идентификатор»

Пример ввода

```
Object
```

1.2 Описание выходных данных

Построчно (одиннадцать строк):

«наименование объекта»

Пример вывода:

```
Object_8_6_2_1
Object_8_6_3_1
Object_8_1
Object_8_1
Object_8_6_2
Object_8_6_3
Object_8_7_4
Object_8_7_5
Object_8_6
```

Object_8_7
Object_8

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- объект object класса MyClass предназначен для демонстрации множественного наследования;
- функция main для основного алгоритма функции.

Класс cl1:

- свойства/поля:
    - поле наименования объекта:
        - наименование — obj1;
        - тип — string;
        - модификатор доступа — private;
- функционал:
    - метод cl1 — параметризированный конструктор , задаёт значение закрытому полю;
    - метод getback — возвращает значение закрытого поля.

Класс cl2:

- свойства/поля:
    - поле наименования объекта:
        - наименование — obj1;
        - тип — string;
        - модификатор доступа — private;
- функционал:
    - метод cl2 — параметризированный конструктор , задаёт значение закрытому полю;
    - метод getback — возвращает значение закрытого поля.

Класс cl3:

- свойства/поля:
    - поле наименования объекта:
        - наименование — obj1;
        - тип — string;
        - модификатор доступа — private;
- функционал:
    - метод cl3 — параметризированный конструктор , задаёт значение закрытому полю;
    - метод getback — возвращает значение закрытого поля.

Класс cl4:

- свойства/поля:
    - поле наименования объекта:
        - наименование — obj1;
        - тип — string;
        - модификатор доступа — private;
- функционал:
    - метод cl4 — параметризированный конструктор , задаёт значение закрытому полю;
    - метод getback — возвращает значение закрытого поля.

Класс cl5:

- свойства/поля:
    - поле наименования объекта:
        - наименование — obj1;
        - тип — string;
        - модификатор доступа — private;
- функционал:
    - метод cl5 — параметризированный конструктор , задаёт значение

закрытому полю;

- метод getback — возвращает значение закрытого поля.

Класс cl6:

- свойства/поля:
  - поле наименования объекта:
    - наименование — obj1;
    - тип — string;
    - модификатор доступа — private;
- функционал:
  - метод cl6 — параметризированный конструктор , задаёт значение закрытому полю;
  - метод getback — возвращает значение закрытого поля.

Класс cl7:

- свойства/поля:
  - поле наименования объекта:
    - наименование — obj1;
    - тип — string;
    - модификатор доступа — private;
- функционал:
  - метод cl7 — параметризированный конструктор , задаёт значение закрытому полю;
  - метод getback — возвращает значение закрытого поля.

Класс MyClass:

- свойства/поля:
  - поле наименования объекта:
    - наименование — obj1;
    - тип — string;

        - модификатор доступа — private;
- функционал:
    - метод MyClass — параметризированный конструктор , задаёт значение закрытому полю;
    - метод getback — возвращает значение закрытого поля.

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
1	cl1			базовый класс	
		cl2	public		2
		cl3	public		3
		cl4	public		4
		cl5	public		5
2	cl2			производный класс от класса cl1	
		cl6	public		6
3	cl3			производный класс от класса cl1	
		cl6	public		6
4	cl4			производный класс от класса cl1	
		cl7	public		7
5	cl5			производный класс от класса cl1	
		cl7	public		7
6	cl6			производный класс от классов cl2 и cl3	
		MyClass	public		8
7	cl7			производный класс от классов cl4 и cl5	
		MyClass	public		8
8	MyClass			производный класс от классов cl6 и cl7	

3 ОПИСАНИЕ АЛГОРИТМОВ

Согласно этапам разработки, после определения необходимого инструментария в разделе «Метод», составляются подробные описания алгоритмов для методов классов и функций.

3.1 Алгоритм конструктора класса cl1

Функционал: параметризированный конструктор , задаёт значение закрытому полю.

Параметры: string obj1.

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса cl1

№	Предикат	Действия	№ перехода
1		присвоение значению закрытого поля obj1 значения obj1+"_1"	∅

3.2 Алгоритм метода getback класса cl1

Функционал: возвращает значение закрытого поля.

Параметры: нет.

Возвращаемое значение: string.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода getback класса cl1

№	Предикат	Действия	№ перехода
1		возврат значения закрытого поля obj1	∅

3.3 Алгоритм конструктора класса cl2

Функционал: параметризированный конструктор , задаёт значение закрытому полю.

Параметры: string obj1.

Алгоритм конструктора представлен в таблице 4.

Таблица 4 – Алгоритм конструктора класса cl2

№	Предикат	Действия	№ перехода
1		присвоение значению закрытого поля obj1 значения obj1+"_2"	∅

3.4 Алгоритм метода getback класса cl2

Функционал: возвращает значение закрытого поля.

Параметры: нет.

Возвращаемое значение: string.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода getback класса cl2

№	Предикат	Действия	№ перехода
1		возврат значения закрытого поля obj1	∅

3.5 Алгоритм конструктора класса cl3

Функционал: параметризированный конструктор , задаёт значение закрытому полю.

Параметры: strin obj1.

Алгоритм конструктора представлен в таблице 6.

Таблица 6 – Алгоритм конструктора класса cl3

№	Предикат	Действия	№ перехода
1		присвоение значению закрытого поля obj1	∅

3.6 Алгоритм метода getback класса cl3

Функционал: возвращает значение закрытого поля.

Параметры: нет.

Возвращаемое значение: string.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода getback класса cl3

№	Предикат	Действия	№ перехода
1		возврат значения закрытого поля obj1	∅

3.7 Алгоритм конструктора класса cl4

Функционал: параметризированный конструктор , задаёт значение закрытому полю.

Параметры: string obj1.

Алгоритм конструктора представлен в таблице 8.

Таблица 8 – Алгоритм конструктора класса cl4

№	Предикат	Действия	№ перехода
1		присвоение значению закрытого поля obj1 значения obj1+"_4"	∅

3.8 Алгоритм метода getback класса cl4

Функционал: возвращает значение закрытого поля.

Параметры: нет.

Возвращаемое значение: string.

Алгоритм метода представлен в таблице 9.

Таблица 9 – Алгоритм метода getback класса cl4

№	Предикат	Действия	№ перехода
1		возврат значения закрытого поля obj1	∅

3.9 Алгоритм конструктора класса cl5

Функционал: параметризированный конструктор , задаёт значение закрытому полю.

Параметры: string obj1.

Алгоритм конструктора представлен в таблице 10.

Таблица 10 – Алгоритм конструктора класса cl5

№	Предикат	Действия	№ перехода
1		присвоение значениб закрытого поля obj1 значения obj1+"_5"	∅

3.10 Алгоритм метода getback класса cl5

Функционал: возвращает значение закрытого поля.

Параметры: нет.

Возвращаемое значение: string.

Алгоритм метода представлен в таблице 11.

Таблица 11 – Алгоритм метода getback класса cl5

№	Предикат	Действия	№ перехода
1		возврат значения закрытого поля obj1	∅

3.11 Алгоритм конструктора класса cl6

Функционал: параметризированный конструктор , задаёт значение закрытому полю.

Параметры: string obj1.

Алгоритм конструктора представлен в таблице 12.

Таблица 12 – Алгоритм конструктора класса cl6

№	Предикат	Действия	№ перехода
1		присвоение значению закрытого поля obj1 значения obj+"_6"	∅

3.12 Алгоритм метода getback класса cl6

Функционал: возвращает значение закрытого поля.

Параметры: нет.

Возвращаемое значение: string.

Алгоритм метода представлен в таблице 13.

Таблица 13 – Алгоритм метода getback класса cl6

№	Предикат	Действия	№ перехода
1		возврат значения закрытого поля obj1	∅

3.13 Алгоритм конструктора класса cl7

Функционал: параметризированный конструктор , задаёт значение закрытому полю.

Параметры: string obj1.

Алгоритм конструктора представлен в таблице 14.

Таблица 14 – Алгоритм конструктора класса cl7

№	Предикат	Действия	№ перехода
1		присвоение значению закрытого поля obj1 значения obj1+"_7"	∅

3.14 Алгоритм метода getback класса cl7

Функционал: возвращает значение закрытого поля.

Параметры: нет.

Возвращаемое значение: string obj1.

Алгоритм метода представлен в таблице 15.

Таблица 15 – Алгоритм метода getback класса cl7

№	Предикат	Действия	№ перехода
1		возврат значения закрытого поля obj1	∅

3.15 Алгоритм конструктора класса MyClass

Функционал: параметризированный конструктор , задаёт значение закрытому полю.

Параметры: string obj1.

Алгоритм конструктора представлен в таблице 16.

Таблица 16 – Алгоритм конструктора класса MyClass

№	Предикат	Действия	№ перехода
1		присвоение значению закрытого поля obj1 значения obj1+"_8"	∅

3.16 Алгоритм метода getback класса MyClass

Функционал: возвращает значение закрытого поля.

Параметры: нет.

Возвращаемое значение: strin g.

Алгоритм метода представлен в таблице 17.

Таблица 17 – Алгоритм метода getback класса MyClass

№	Предикат	Действия	№ перехода
1		возврат значения закрытого поля obj1	∅

3.17 Алгоритм функции main

Функционал: основной алгоритм программы.

Параметры: нет.

Возвращаемое значение: int.

Алгоритм функции представлен в таблице 18.

Таблица 18 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление указателя obj2	2
2		объявление переменной obj1 типа string	3
3		ввод значения переменной obj1	4
4		создание объекта object класса MyClass с помощью параметризированного конструктора параметром obj1	5
5		присвоение указателю obj2 ссылки на объект object	6
6		вызов метода getback() через указатель obj2 для объекта класса cl1, созданного объектом класса cl2, и вывод его значения на экран	7
7		вызов метода getback() через указатель obj2 для объекта класса cl1, созданного объектом класса cl3, вывод его значения на экран	8
8		вызов метода getback() через указатель obj2 для объекта класса cl1, созданного объектом класса cl4, вывод его значения на экран	9
9		вызов метода getback() через указатель obj2 для объекта класса cl1, созданного объектом класса cl5, вывод его значения на экран	10

№	Предикат	Действия	№ перехода
10		вызов метода getback() через указатель obj2 для объекта класса cl2, вывод его значения на экран	11
11		вызов метода getback() через указатель obj2 для объекта класса cl3, вывод его значения на экран	12
12		вызов метода getback() через указатель obj2 для объекта класса c4, вывод его значения на экран	13
13		вызов метода getback() через указатель obj2 для объекта класса cl5, вывод его значения на экран	14
14		вызов метода getback() через указатель obj2 для объекта класса cl6, вывод его значения на экран	15
15		вызов метода getback() через указатель obj2 для объекта класса cl7, вывод его значения на экран	16
16		вывод метода getback() через указатель obj2 , вывод его значения на экран	17
17		return 0	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-4.

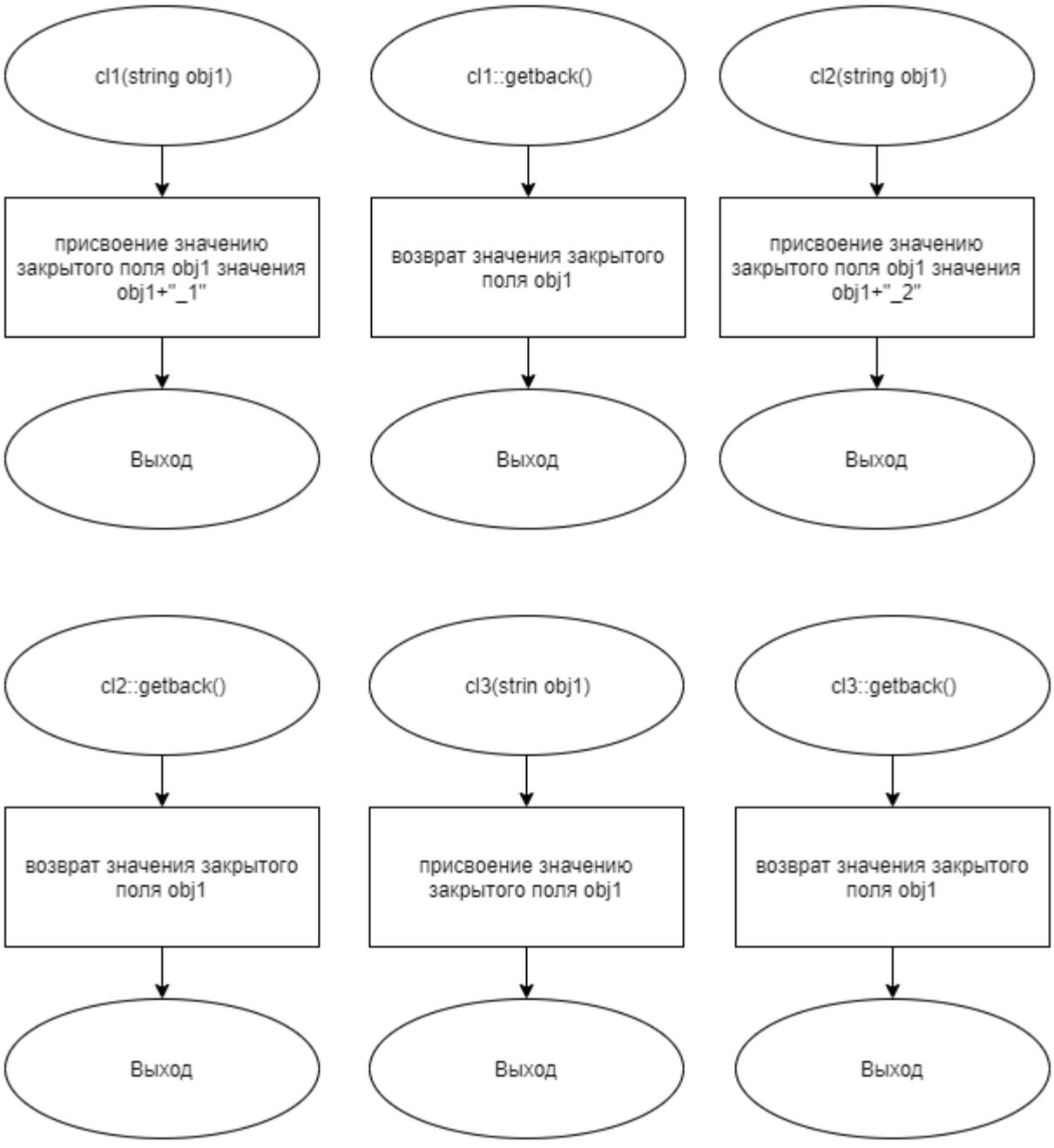

Рисунок 1 – Блок-схема алгоритма

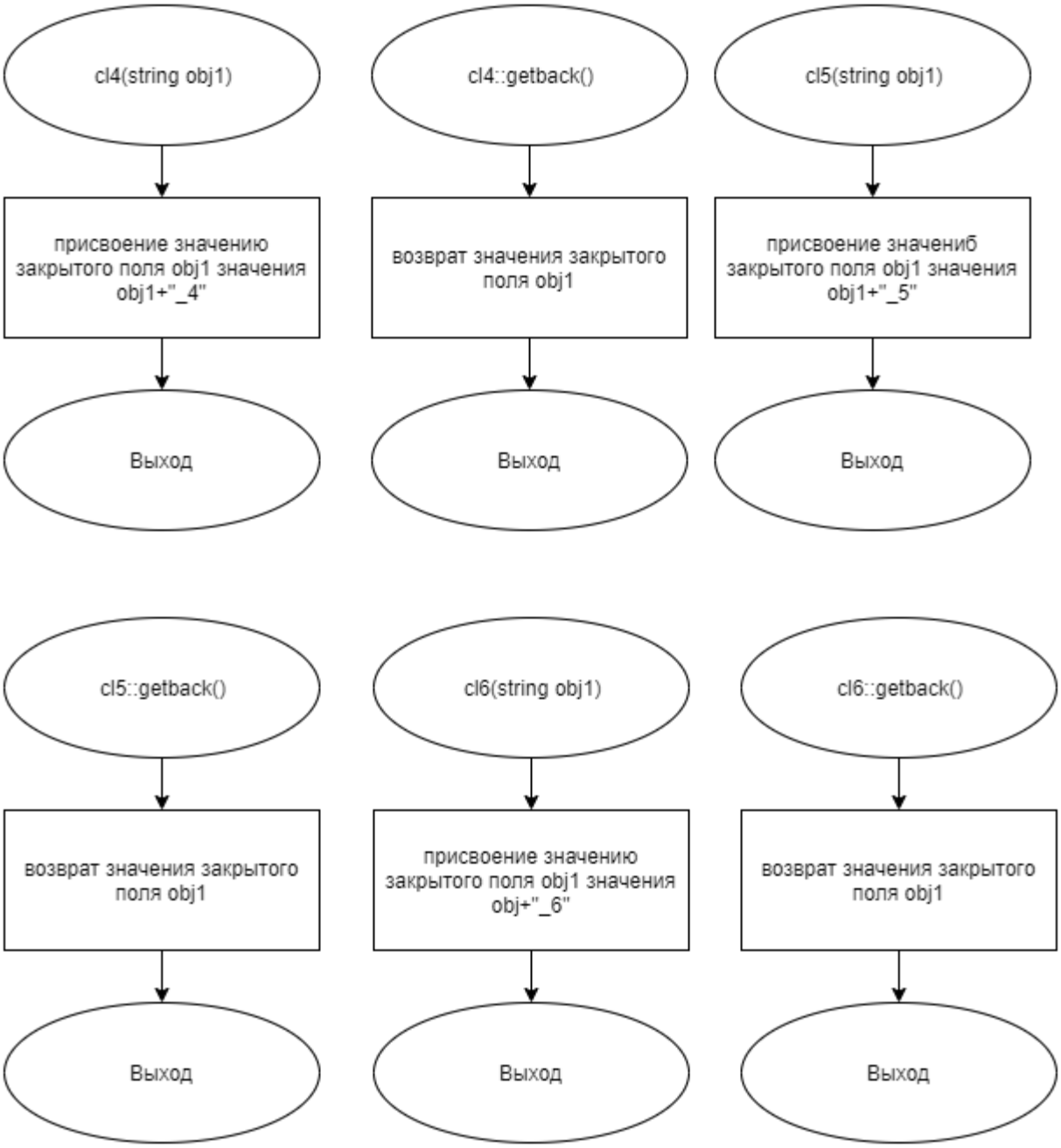

Рисунок 2 – Блок-схема алгоритма

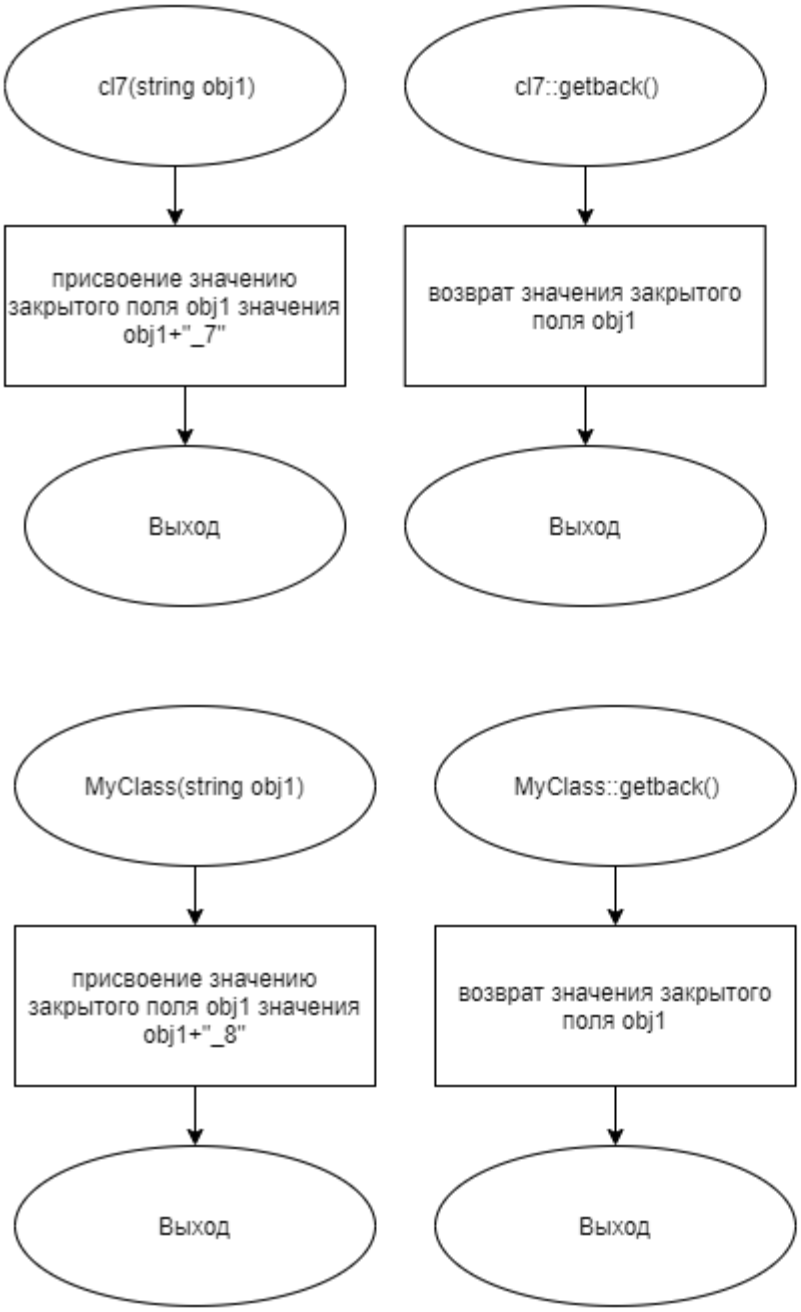

Рисунок 3 – Блок-схема алгоритма

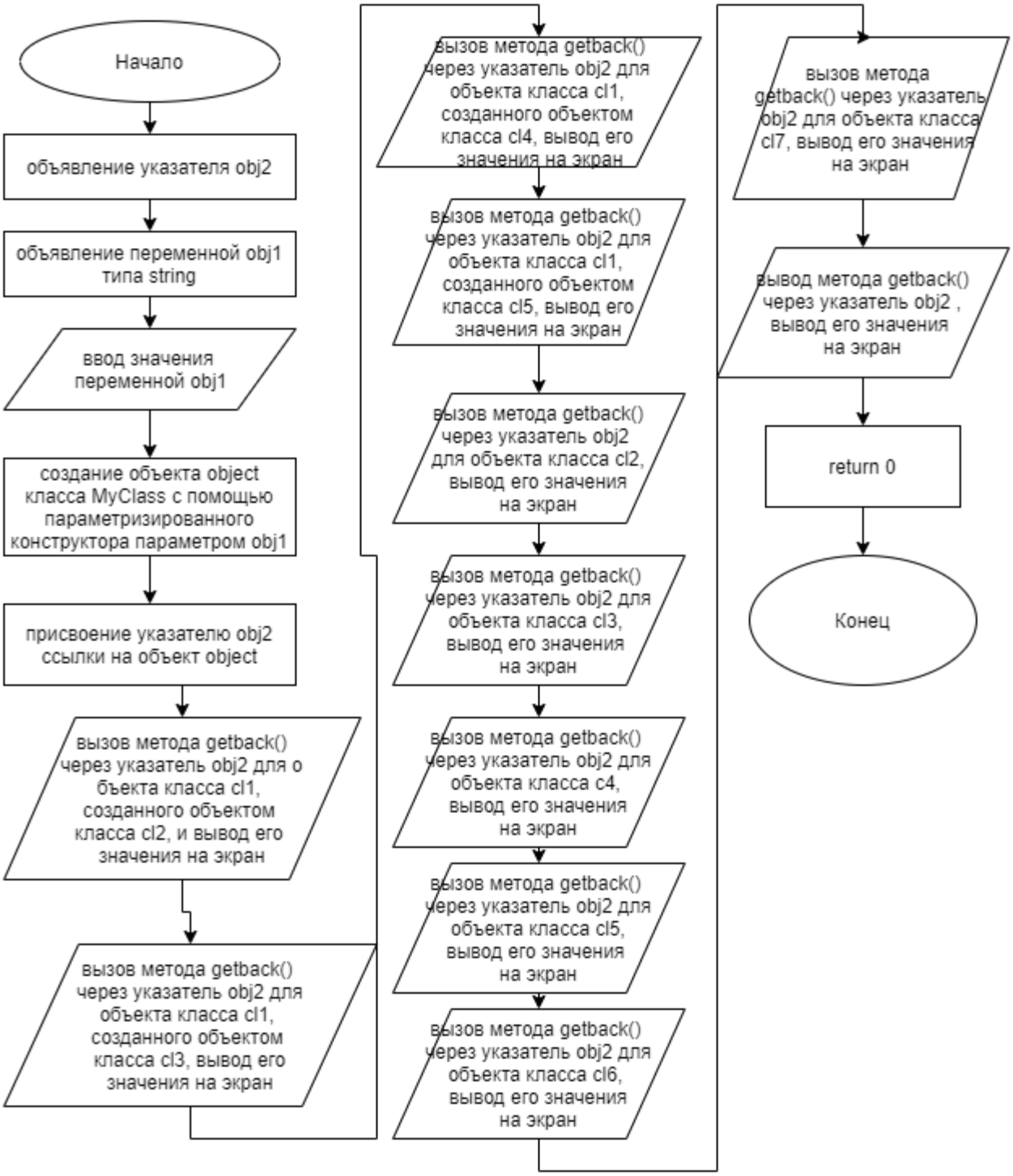

Рисунок 4 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл cl1.cpp

Листинг 1 – cl1.cpp

```
#include  "cl1.h"
using namespace std;

cl1::cl1(string obj1){
   this -> obj1=obj1+"_1";
}
string cl1::getback(){
   return obj1;
}
```

5.2 Файл cl1.h

Листинг 2 – cl1.h

```
#ifndef __CL1__H
#define __CL1__H
#include <iostream>
#include <string>
using namespace std;

class cl1{
   private:
      string obj1;
   public:
      cl1(string obj1);
      string getback();
};


#endif
```

5.3 Файл cl2.cpp

Листинг 3 – cl2.cpp

```
#include "cl2.h"
using namespace std;

cl2::cl2(string obj1):
cl1(obj1+"_2")
{
   this -> obj1=obj1+"_2";
}
string cl2::getback()
{
   return obj1;
}
```

5.4 Файл cl2.h

Листинг 4 – cl2.h

```
#ifndef __CL2__H
#define __CL2__H
#include "cl1.h"
using namespace std;

class cl2:
public cl1{
   private:
      string obj1;
   public:
      cl2(string obj1);
      string getback();
};


#endif
```

5.5 Файл cl3.cpp

Листинг 5 – cl3.cpp

```
#include "cl3.h"
```

```
using namespace std;

cl3::cl3(string obj1):
cl1(obj1+"_3"){
   this -> obj1=obj1+"_3";
}
string cl3::getback()
{
   return obj1;
}
```

5.6 Файл cl3.h

Листинг 6 – cl3.h

```
#ifndef __CL3__H
#define __CL3__H
#include "cl1.h"
using namespace std;

class cl3:
public cl1{
   private:
      string obj1;
   public:
      cl3(string obj1);
      string getback();
};


#endif
```

5.7 Файл cl4.cpp

Листинг 7 – cl4.cpp

```
#include "cl4.h"
using namespace std;

cl4::cl4(string obj1):
cl1(obj1+"_4"){
   this -> obj1=obj1+"_4";
}
string cl4::getback()
{
```

```
   return obj1;
}
```

5.8 Файл cl4.h

Листинг 8 – cl4.h

```
#ifndef __CL4__H
#define __CL4__H
#include "cl1.h"
using namespace std;

class cl4:
virtual public cl1{
   private:
      string obj1;
   public:
      cl4(string obj1);
      string getback();
};


#endif
```

5.9 Файл cl5.cpp

Листинг 9 – cl5.cpp

```
#include "cl5.h"
using namespace std;

cl5::cl5(string obj1):
cl1(obj1+"_5"){
   this ->obj1=obj1+"_5";
}
string cl5::getback()
{
   return obj1;
}
```

5.10 Файл cl5.h

Листинг 10 – cl5.h

```
#ifndef __CL5__H
#define __CL5__H
#include "cl1.h"
using namespace std;

class cl5:
virtual public cl1{
   private:
      string obj1;
   public:
      cl5(string obj1);
      string getback();
};


#endif
```

5.11 Файл cl6.cpp

Листинг 11 – cl6.cpp

```
#include "cl6.h"
using namespace std;
cl6::cl6(string obj1):
cl2(obj1+"_6"), cl3(obj1+"_6"){
   this ->obj1=obj1+"_6";
}
string cl6::getback()
{
   return obj1;
}
```

5.12 Файл cl6.h

Листинг 12 – cl6.h

```
#ifndef __CL6__H
#define __CL6__H
#include "cl2.h"
```

```
#include "cl3.h"
using namespace std;

class cl6:
public cl2, public cl3
{
   private:
      string obj1;
   public:
      cl6(string obj1);
      string getback();
};


#endif
```

5.13 Файл cl7.cpp

Листинг 13 – cl7.cpp

```
#include "cl7.h"
using namespace std;

cl7::cl7(string obj1):
cl1(obj1+"_7"), cl4(obj1+"_7"),cl5(obj1+"_7")
{
   this ->obj1=obj1+"_7";
}
string cl7::getback()
{
   return obj1;
}
```

5.14 Файл cl7.h

Листинг 14 – cl7.h

```
#ifndef __CL7__H
#define __CL7__H
#include "cl4.h"
#include "cl5.h"
using namespace std;

class cl7:
public cl4 , public cl5
```

```
{
   private:
      string obj1;
   public:
      cl7(string obj1);
      string getback();
};


#endif
```

5.15 Файл main.cpp

Листинг 15 – main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include "MyClass.h"

int main(){
   MyClass* obj2;
   string obj1;
   cin>>obj1;
   MyClass object(obj1);
   obj2=&object;
   cout<<((cl1*)(cl2*)obj2)->getback()<<endl;
   cout<<((cl1*)(cl3*)obj2)->getback()<<endl;
   cout<<((cl1*)(cl4*)obj2)->getback()<<endl;
   cout<<((cl1*)(cl5*)obj2)->getback()<<endl;
   cout<<((cl2*)obj2)->getback()<<endl;
   cout<<((cl3*)obj2)->getback()<<endl;
   cout<<((cl4*)obj2)->getback()<<endl;
   cout<<((cl5*)obj2)->getback()<<endl;
   cout<<((cl6*)obj2)->getback()<<endl;
   cout<<((cl7*)obj2)->getback()<<endl;
   cout<<obj2->getback();
   return (0);
}
```

5.16 Файл MyClass.cpp

Листинг 16 – MyClass.cpp

```
#include "MyClass.h"
```

```
using namespace std;

MyClass::MyClass(string obj1):
cl7::cl1(obj1+"_8"), cl6(obj1+"_8"),cl7(obj1+"_8")
{
   this -> obj1=obj1+"_8";
}
string MyClass::getback()
{
   return obj1;
}
```

5.17 Файл MyClass.h

Листинг 17 – MyClass.h

```
#ifndef __MyClass__H
#define __MyClass__H
#include "cl6.h"
#include "cl7.h"
using namespace std;

class MyClass:
public cl6, public cl7
{
   private:
      string obj1;
   public:
      MyClass(string obj1);
      string getback();
};

#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 19.

Таблица 19 – Результат тестирования программы

Входные данные	**Ожидаемые выходные данные**	**Фактические выходные данные**

`Object`	`Object_8_6_2_1`
	`Object_8_6_3_1`
	`Object_8_1`
	`Object_8_1`
	`Object_8_6_2`
	`Object_8_6_3`
	`Object_8_7_4`
	`Object_8_7_5`
	`Object_8_6`
	`Object_8_7`
	`Object_8`
		`Object_8_6_2_1`
		`Object_8_6_3_1`
		`Object_8_1`
		`Object_8_1`
		`Object_8_6_2`
		`Object_8_6_3`
		`Object_8_7_4`
		`Object_8_7_5`
		`Object_8_6`
		`Object_8_7`
		`Object_8`

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 19 Единая система программной документации.

2. Методическое пособие студента для выполнения практических заданий, контрольных и курсовых работ по дисциплине «Объектно-ориентированное программирование» [Электронный ресурс] – URL: https://mirea.aco-avrora.ru/student/files/methodichescoe_posobie_dlya_laboratornyh_rabot_3.pdf (дата обращения 05.05.2021).

3. Приложение к методическому пособию студента по выполнению заданий в рамках курса «Объектно-ориентированное программирование» [Электронный ресурс]. URL: https://mirea.aco-avrora.ru/student/files/Prilozheniye_k_methodichke.pdf (дата обращения 05.05.2021).

4. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2019. — 624 с.

5. Видео лекции по курсу «Объектно-ориентированное программирование» [Электронный ресурс]. АСО «Аврора».

6. Антик М.И. Дискретная математика [Электронный ресурс]: Учебное пособие /Антик М.И., Казанцева Л.В. — М.: МИРЭА — Российский технологический университет, 2018 — 1 электрон. опт. диск (CD-ROM).