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 объект класса cl_8 класса obj предназначен для ;
- функция main для Основная функция;
- Объект стандартного потока ввода/вывода cin/cout.

Класс cl_1:

- свойства/поля:
  - поле Имя объекта:
    - наименование — s_name;
    - тип — Строковый;
    - модификатор доступа — private;
- функционал:
  - метод print — Вывод объекта;
  - метод cl_1 — Параметризированный конструктор.

Класс cl_2:

- свойства/поля:
  - поле Название объекта:
    - наименование — s_name;
    - тип — Строковое;
    - модификатор доступа — private;
- функционал:
  - метод print — Вывод объекта cl_2;
  - метод cl_2 — Параметризированный конструктор.

Класс cl_3:

- свойства/поля:
  - поле Название объекта:

        - наименование — s_name;
        - тип — Строковый;
        - модификатор доступа — private;
- функционал:
    - метод cl_3 — Параметризированный конструктор;
    - метод print — Вывод объекта.

Класс cl_4:

- свойства/поля:
    - поле Название объекта:
        - наименование — s_name;
        - тип — Строковый;
        - модификатор доступа — private;
- функционал:
    - метод cl_4 — Параметризированный конструктор;
    - метод print — Вывод объекта.

Класс cl_5:

- свойства/поля:
    - поле Название объекта:
        - наименование — s_name;
        - тип — Строковый;
        - модификатор доступа — private;
- функционал:
    - метод cl_5 — Параметризированный конструктор;
    - метод print — Вывод объекта.

Класс cl_6:

- свойства/поля:
    - поле Название объекта:

        - наименование — s_name;
        - тип — Строковы;
        - модификатор доступа — private;
- функционал:
    - метод cl_6 — Параметризированный конструктор;
    - метод print — Вывод объекта.

Класс cl_7:

- свойства/поля:
    - поле Название объекта:
        - наименование — s_name;
        - тип — Строковый;
        - модификатор доступа — private;
- функционал:
    - метод cl_7 — Параметризированный конструктор;
    - метод print — Вывод объекта.

Класс cl_8:

- свойства/поля:
    - поле Название объекта:
        - наименование — s_name;
        - тип — Строковый;
        - модификатор доступа — private;
- функционал:
    - метод cl_8 — Параметризированный конструктор;
    - метод print — Вывод объекта.

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
1	cl_1				
		cl_2	public		2
		cl_3	public		3
		cl_4	virtual public		4
		cl_5	virtual public		5
2	cl_2				
		cl_6	public		6
3	cl_3				
		cl_6	public		6
4	cl_4				
		cl_7	public		7
5	cl_5				
		cl_7	public		7
6	cl_6				
		cl_8	public		8
7	cl_7				
		cl_8	public		8
8	cl_8				

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 метода print класса cl_1

Функционал: Вывод объекта.

Параметры: нет.

Возвращаемое значение: Отутствует.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода print класса cl_1

№	Предикат	Действия	№ перехода
1		Вывод объекта	∅

3.2 Алгоритм конструктора класса cl_1

Функционал: Параметризированный конструктор.

Параметры: s_name.

Алгоритм конструктора представлен в таблице 3.

Таблица 3 – Алгоритм конструктора класса cl_1

№	Предикат	Действия	№ перехода
1		Присвоение переменной s_name значения name	∅

3.3 Алгоритм метода print класса cl_2

Функционал: Вывод объекта cl_2.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода print класса cl_2

№	Предикат	Действия	№ перехода
1		Вывод объекта	∅

3.4 Алгоритм конструктора класса cl_2

Функционал: Параметризированный конструктор.

Параметры: нет.

Алгоритм конструктора представлен в таблице 5.

Таблица 5 – Алгоритм конструктора класса cl_2

№	Предикат	Действия	№ перехода
1		Присвоение переменной s_name значения name	∅

3.5 Алгоритм конструктора класса cl_3

Функционал: Параметризированный конструктор.

Параметры: string.

Алгоритм конструктора представлен в таблице 6.

Таблица 6 – Алгоритм конструктора класса cl_3

№	Предикат	Действия	№ перехода
1		Присвоение s_name значения	∅

3.6 Алгоритм метода print класса cl_3

Функционал: Вывод объекта.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода print класса cl_3

№	Предикат	Действия	№ перехода
1		Вывод объекта	∅

3.7 Алгоритм конструктора класса cl_4

Функционал: Параметризированный конструктор.

Параметры: s_name.

Алгоритм конструктора представлен в таблице 8.

Таблица 8 – Алгоритм конструктора класса cl_4

№	Предикат	Действия	№ перехода
1		Присвоение переменной s_name значения name	∅

3.8 Алгоритм метода print класса cl_4

Функционал: Вывод объекта.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 9.

Таблица 9 – Алгоритм метода print класса cl_4

№	Предикат	Действия	№ перехода
1		Вывод объекта	∅

3.9 Алгоритм конструктора класса cl_5

Функционал: Параметризированный конструктор.

Параметры: s_name.

Алгоритм конструктора представлен в таблице 10.

Таблица 10 – Алгоритм конструктора класса cl_5

№	Предикат	Действия	№ перехода
1		Присвоение переменной s_name значения name	∅

3.10 Алгоритм метода print класса cl_5

Функционал: Вывод объекта.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 11.

Таблица 11 – Алгоритм метода print класса cl_5

№	Предикат	Действия	№ перехода
1		Вывод объекта	∅

3.11 Алгоритм конструктора класса cl_6

Функционал: Параметризированный конструктор.

Параметры: s_name.

Алгоритм конструктора представлен в таблице 12.

Таблица 12 – Алгоритм конструктора класса cl_6

№	Предикат	Действия	№ перехода
1		Присвоение s_name значению name	∅

3.12 Алгоритм метода print класса cl_6

Функционал: Вывод объекта.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 13.

Таблица 13 – Алгоритм метода print класса cl_6

№	Предикат	Действия	№ перехода
1		Вывод объекта	∅

3.13 Алгоритм конструктора класса cl_7

Функционал: Параметризированный конструктор.

Параметры: s_name.

Алгоритм конструктора представлен в таблице 14.

Таблица 14 – Алгоритм конструктора класса cl_7

№	Предикат	Действия	№ перехода
1		Присвоение переменной s_name значения name	∅

3.14 Алгоритм метода print класса cl_7

Функционал: Вывод объекта.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 15.

Таблица 15 – Алгоритм метода print класса cl_7

№	Предикат	Действия	№ перехода
1		Вывод объекта	∅

3.15 Алгоритм конструктора класса cl_8

Функционал: Параметризированный конструктор.

Параметры: нет.

Алгоритм конструктора представлен в таблице 16.

Таблица 16 – Алгоритм конструктора класса cl_8

№	Предикат	Действия	№ перехода
1		Присвоение переменной s_name значения name	∅

3.16 Алгоритм метода print класса cl_8

Функционал: Вывод объекта.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 17.

Таблица 17 – Алгоритм метода print класса cl_8

№	Предикат	Действия	№ перехода
1		Вывод объекта	∅

3.17 Алгоритм функции main

Функционал: Основная функция.

Параметры: нет.

Возвращаемое значение: Целое.

Алгоритм функции представлен в таблице 18.

Таблица 18 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Объявление одного указателя на объект класса x	2
2		Объявление переменной строкового типа	3
3		Ввод значение строковой переменной. Вводимое значение является идентификатором	4
4		Создать объект класса 8 посредством параметризированного конструктора, передав в качестве аргумента строковую переменную	5
5		Адрес созданного объекта присвоить указателю на объект класса x	6
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	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-9.

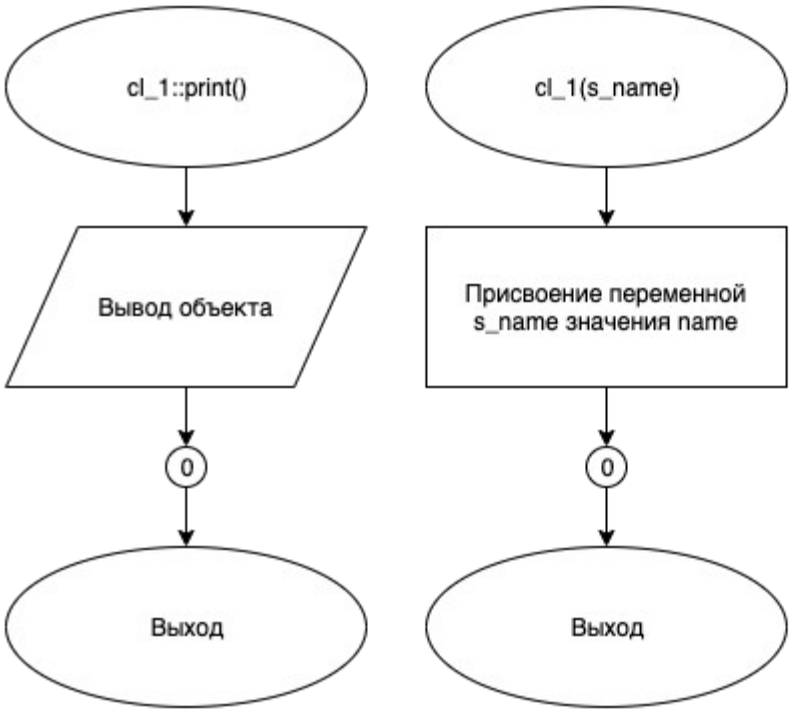

Рисунок 1 – Блок-схема алгоритма

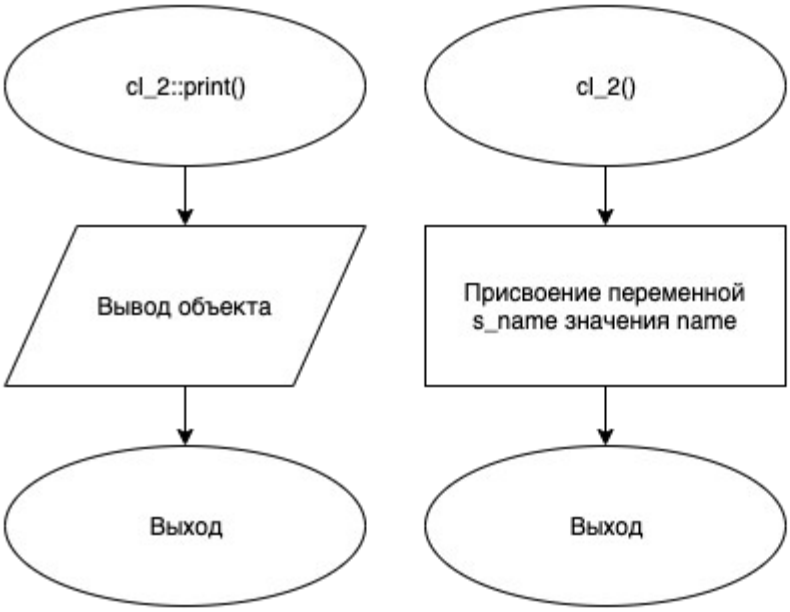

Рисунок 2 – Блок-схема алгоритма

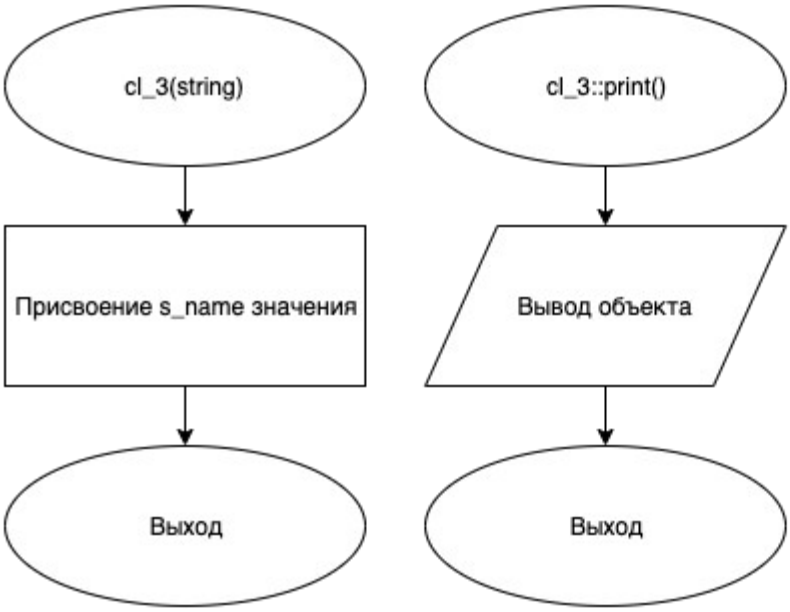

Рисунок 3 – Блок-схема алгоритма

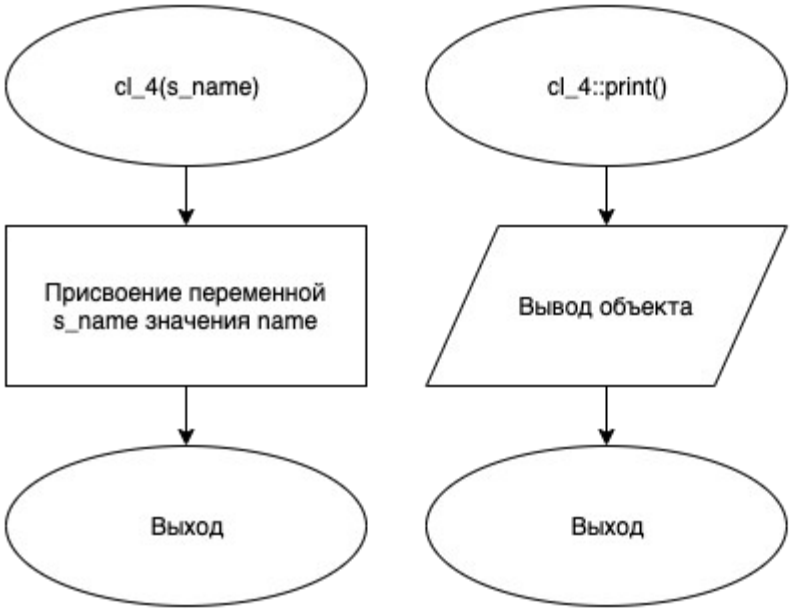

Рисунок 4 – Блок-схема алгоритма

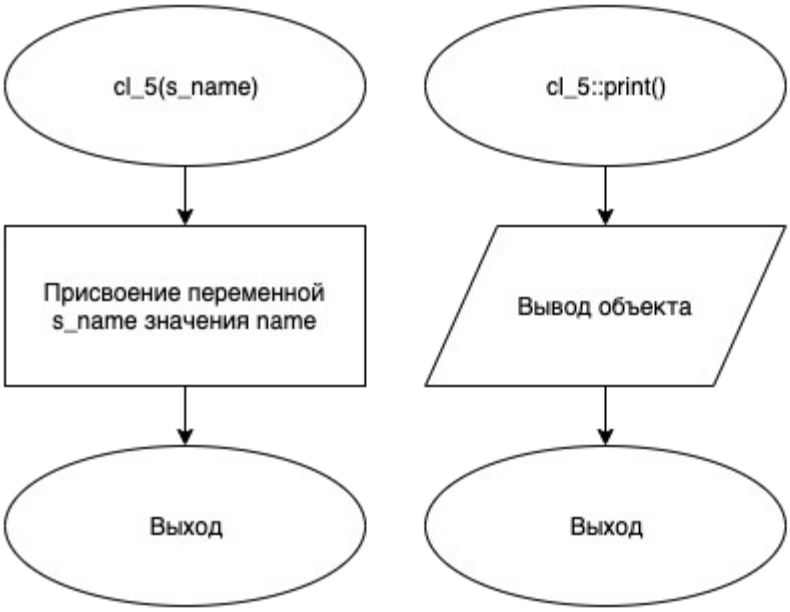

Рисунок 5 – Блок-схема алгоритма

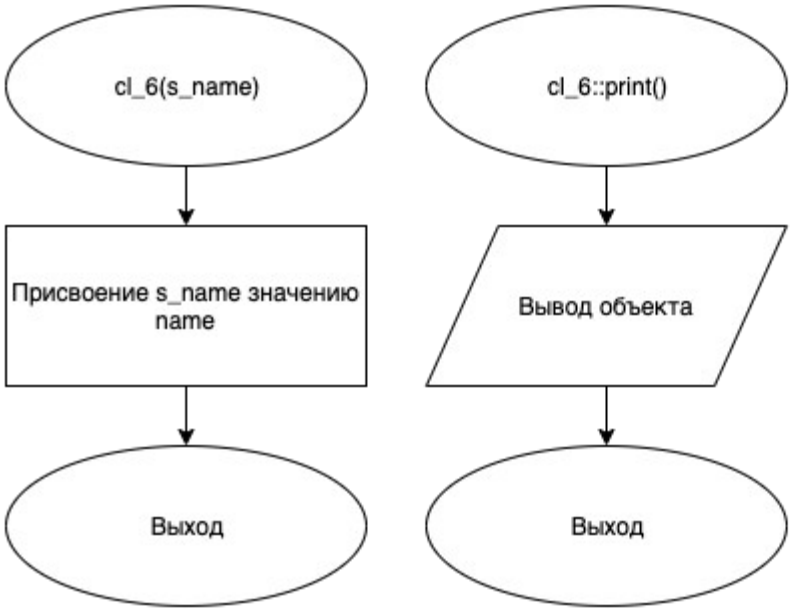

Рисунок 6 – Блок-схема алгоритма

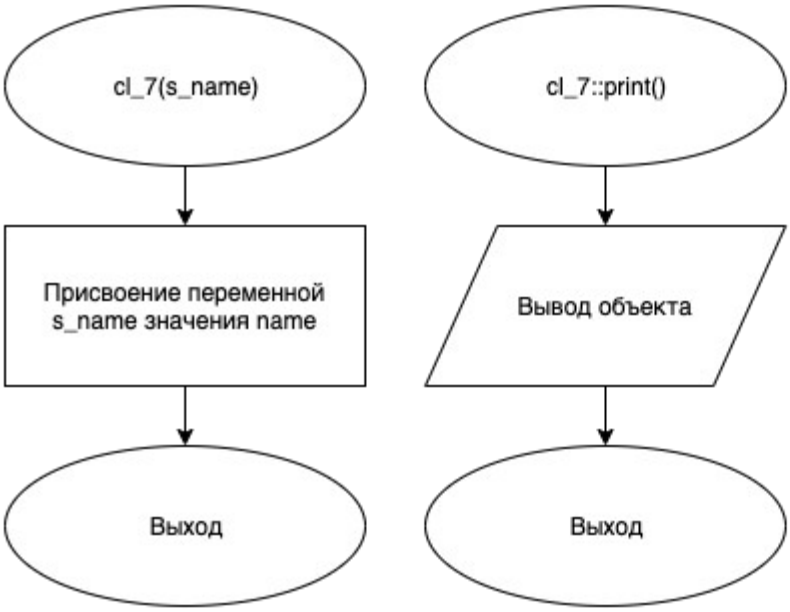

Рисунок 7 – Блок-схема алгоритма

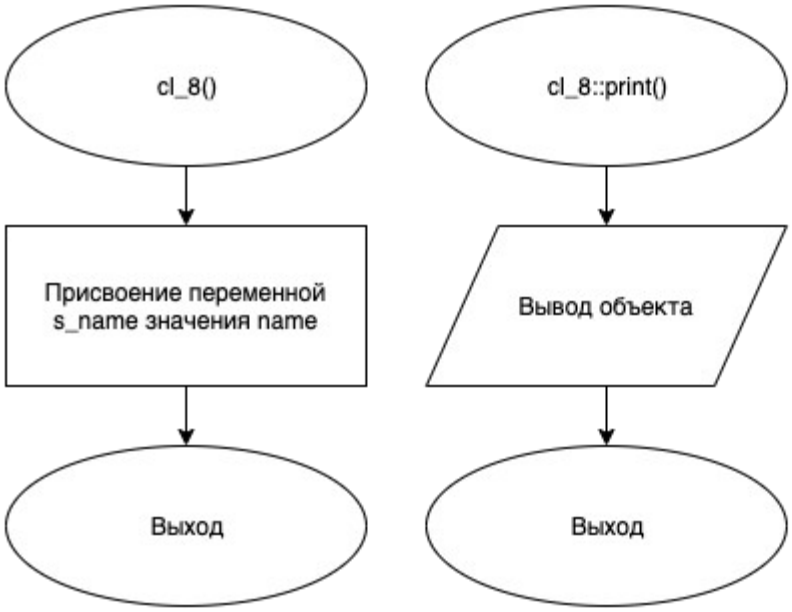

Рисунок 8 – Блок-схема алгоритма

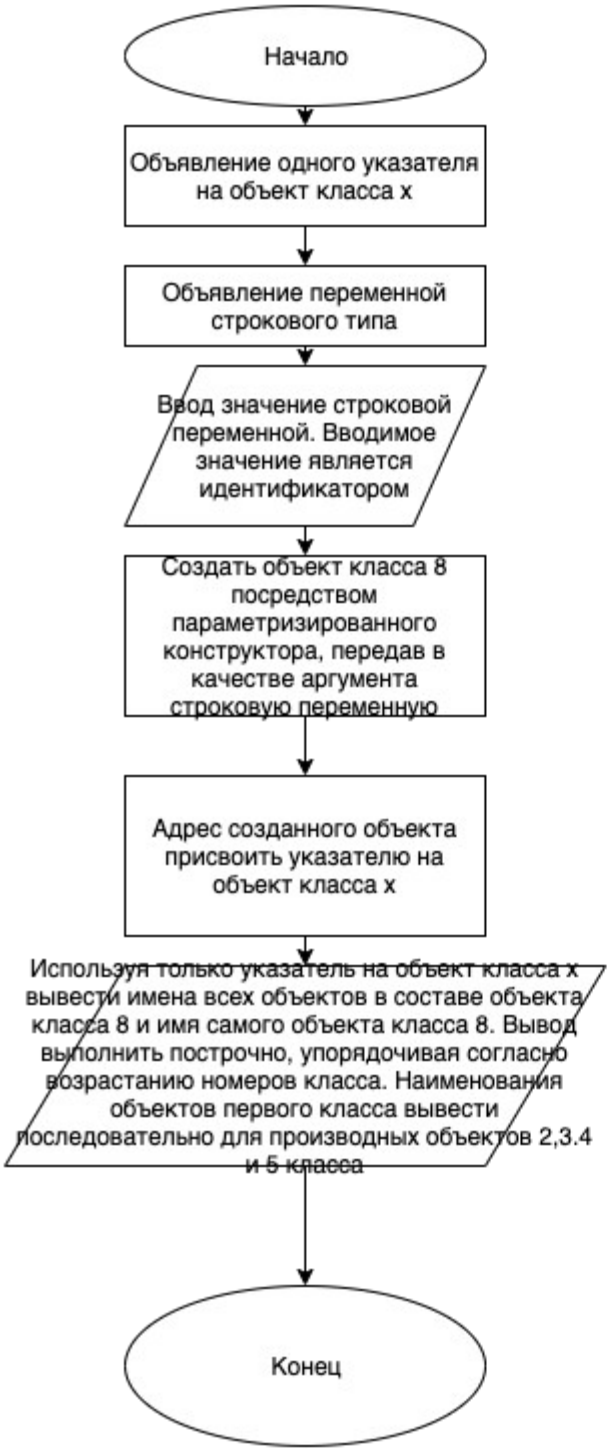

Рисунок 9 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл cl_1.cpp

Листинг 1 – cl_1.cpp

```
#include "cl_1.h"
cl_1::cl_1(string name)
{
   s_name = name;
}

string cl_1::print()
{
   return s_name;
}
```

5.2 Файл cl_1.h

Листинг 2 – cl_1.h

```
#ifndef __CL_1__H
#define __CL_1__H
#include <iostream>
#include <string>
using namespace std;
class cl_1
{
private:
   string s_name;
public:
   cl_1(string s_name);
   string print();
};
#endif
```

5.3 Файл cl_2.cpp

Листинг 3 – cl_2.cpp

```
#include "cl_2.h"
cl_2::cl_2(string name):cl_1(name + "_1")
{
   s_name = name;
}

string cl_2::print()
{
   return s_name;
}
```

5.4 Файл cl_2.h

Листинг 4 – cl_2.h

```
#ifndef __CL_2__H
#define __CL_2__H
#include "cl_1.h"
class cl_2:public cl_1
{
private:
   string s_name;
public:
   cl_2(string s_name);
   string print();
};
#endif
```

5.5 Файл cl_3.cpp

Листинг 5 – cl_3.cpp

```
#include "cl_3.h"
cl_3::cl_3(string name):cl_1(name + "_1")
{
   s_name = name;
}

string cl_3::print()
```

```
{
   cout << endl;
   return s_name;
}
```

5.6 Файл cl_3.h

Листинг 6 – cl_3.h

```
#ifndef __CL_3__H
#define __CL_3__H
#include "cl_1.h"
class cl_3:public cl_1
{
private:
   string s_name;
public:
   cl_3(string s_name);
   string print();
};
#endif
```

5.7 Файл cl_4.cpp

Листинг 7 – cl_4.cpp

```
#include "cl_4.h"
cl_4::cl_4(string name):cl_1(name + "_1")
{
   s_name = name;
}

string cl_4::print()
{
   cout << endl;
   return s_name;
}
```

5.8 Файл cl_4.h

Листинг 8 – cl_4.h

```
#ifndef __CL_4__H
#define __CL_4__H
#include "cl_1.h"
class cl_4:virtual public cl_1
{
private:
   string s_name;
public:
   cl_4(string s_name);
   string print();
};
#endif
```

5.9 Файл cl_5.cpp

Листинг 9 – cl_5.cpp

```
#include "cl_5.h"
cl_5::cl_5(string name):cl_1(name + "_1")
{
   s_name = name;
}

string cl_5::print()
{
   cout << endl;
   return s_name;
}
```

5.10 Файл cl_5.h

Листинг 10 – cl_5.h

```
#ifndef __CL_5__H
#define __CL_5__H
#include "cl_1.h"
class cl_5:virtual public cl_1
{
private:
```

```
   string s_name;
public:
   cl_5(string s_name);
   string print();
};


#endif
```

5.11 Файл cl_6.cpp

Листинг 11 – cl_6.cpp

```
#include "cl_6.h"
cl_6::cl_6(string name):cl_2(name + "_2"),cl_3(name + "_3")
{
   s_name = name;
}

string cl_6::print()
{
   cout << endl;
   return s_name;
}
```

5.12 Файл cl_6.h

Листинг 12 – cl_6.h

```
#ifndef __CL_6__H
#define __CL_6__H
#include "cl_2.h"
#include "cl_3.h"
class cl_6:public cl_2, public cl_3
{
private:
   string s_name;
public:
   cl_6(string s_name);
   string print();
};

#endif
```

5.13 Файл cl_7.cpp

Листинг 13 – cl_7.cpp

```
#include "cl_7.h"
cl_7::cl_7(string  name):cl_4(name  +  "_4"),cl_5(name  +  "_5"),cl_1(name  +
"_7")
{
   s_name = name;
}

string cl_7::print()
{
   cout << endl;
   return s_name;
}
```

5.14 Файл cl_7.h

Листинг 14 – cl_7.h

```
#ifndef __CL_7__H
#define __CL_7__H
#include "cl_4.h"
#include "cl_5.h"
class cl_7:public cl_4, public cl_5
{
private:
   string s_name;
public:
   cl_7(string s_name);
   string print();
};


#endif
```

5.15 Файл cl_8.cpp

Листинг 15 – cl_8.cpp

```
#include "cl_8.h"
cl_8::cl_8(string  name):cl_6(name  +  "_6"),cl_7(name  +  "_7"),  cl_1(name  +
```

```
"_1")
{
   s_name = name;
}

string cl_8::print()
{
   cout << endl;
   return s_name;
}
```

5.16 Файл cl_8.h

Листинг 16 – cl_8.h

```
#ifndef __CL_8__H
#define __CL_8__H
#include "cl_6.h"
#include "cl_7.h"
class cl_8:public cl_6, public cl_7
{
private:
   string s_name;
public:
   cl_8(string s_name);
   string print();
};

#endif
```

5.17 Файл main.cpp

Листинг 17 – main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include "cl_8.h"
int main()
{
   string str;
   cin >> str;
   cl_8 obj(str + "_8");
   cl_8 * ref = & obj;
   cout << ((cl_1 * ) ((cl_2 * ) (ref))) -> print()<< endl;
   cout << ((cl_1 * ) ((cl_3 * ) (ref))) -> print()<< endl;
```

```
        cout << ((cl_1 * ) ((cl_4 * ) (ref))) -> print()<< endl;
        cout << ((cl_1 * ) ((cl_5 * ) (ref))) -> print() << endl;
        cout << ((cl_2 * ) ((cl_6 * ) (ref))) -> print();
        cout << ((cl_3 * ) ((cl_6 * ) (ref))) -> print();
        cout << ((cl_4 * ) ((cl_7 * ) (ref))) -> print();
        cout << ((cl_5 * ) ((cl_7 * ) (ref))) -> print();
        cout << ((cl_6 * ) (ref)) -> print();
        cout << ((cl_7 * ) (ref)) -> print();
        cout << ref -> print();
        return(0);
    }
```

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).