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 используются:

- объект стандартного потока ввода std::cin (используется для ввода с клавиатуры);
- объект стандартного потока вывода std::cout (используется для вывода на экран);
- объекты классов cl_1, cl_2, cl_3, cl_4, cl_5, cl_6, cl_7, cl_8.

Класс cl_1:

- Свойства (поля):
  - Поля, хранящее наименование объекта:
    - Наименование - name;
    - Тип - строка;
    - Модификатор доступа - закрытый;
  - Методы:
    - Параметризованный конструктор:
      - Функционал - создание объекта с заданным наименованием;
      - Возвращаемое значение - объект класса cl_1;
      - Модификатор доступа - открытый;
      - Параметры:
        - name - наименование объекта (строка);
    - Метод print:
      - Функционал - вывод наименования объекта;
      - Возвращаемое значение - отсутствует (void);
      - Модификатор доступа - открытый;
      - Параметры - отсутствуют.

Класс cl_2:

- Свойства (поля):
  - Поля, хранящее наименование объекта:
    - Наименование - name;
    - Тип - строка;
    - Модификатор доступа - закрытый;
  - Методы:
    - Параметризованный конструктор:
      - Функционал - создание объекта с заданным наименованием;
      - Возвращаемое значение - объект класса cl_2;
      - Модификатор доступа - открытый;
      - Параметры:
        - name - наименование объекта (строка);
    - Метод print:
      - Функционал - вывод наименования объекта;
      - Возвращаемое значение - отсутствует (void);
      - Модификатор доступа - открытый;
      - Параметры - отсутствуют.

Класс cl_3:

- Свойства (поля):
  - Поля, хранящее наименование объекта:
    - Наименование - name;
    - Тип - строка;
    - Модификатор доступа - закрытый;
  - Методы:
    - Параметризованный конструктор:

        - Функционал - создание объекта с заданным наименованием;
        - Возвращаемое значение - объект класса cl_3;
        - Модификатор доступа - открытый;
        - Параметры:
            - name - наименование объекта (строка);
    - Метод print:
        - Функционал - вывод наименования объекта;
        - Возвращаемое значение - отсутствует (void);
        - Модификатор доступа - открытый;
        - Параметры - отсутствуют.

Класс cl_4:

- Свойства (поля):
    - Поля, хранящее наименование объекта:
        - Наименование - name;
        - Тип - строка;
        - Модификатор доступа - закрытый;
    - Методы:
        - Параметризованный конструктор:
            - Функционал - создание объекта с заданным наименованием;
            - Возвращаемое значение - объект класса cl_4;
            - Модификатор доступа - открытый;
            - Параметры:
                - name - наименование объекта (строка);
        - Метод print:
            - Функционал - вывод наименования объекта;

            - Возвращаемое значение - отсутствует (void);
            - Модификатор доступа - открытый;
            - Параметры - отсутствуют.

Класс cl_5:

- Свойства (поля):
    - Поля, хранящее наименование объекта:
        - Наименование - name;
        - Тип - строка;
        - Модификатор доступа - закрытый;
    - Методы:
        - Параметризованный конструктор:
            - Функционал - создание объекта с заданным наименованием;
            - Возвращаемое значение - объект класса cl_5;
            - Модификатор доступа - открытый;
            - Параметры:
                - name - наименование объекта (строка);
        - Метод print:
            - Функционал - вывод наименования объекта;
            - Возвращаемое значение - отсутствует (void);
            - Модификатор доступа - открытый;
            - Параметры - отсутствуют.

Класс cl_6:

- Свойства (поля):
    - Поля, хранящее наименование объекта:
        - Наименование - name;
        - Тип - строка;

        - Модификатор доступа - закрытый;
    - Методы:
        - Параметризованный конструктор:
            - Функционал - создание объекта с заданным наименованием;
            - Возвращаемое значение - объект класса cl_6;
            - Модификатор доступа - открытый;
            - Параметры:
                - name - наименование объекта (строка);
        - Метод print:
            - Функционал - вывод наименования объекта;
            - Возвращаемое значение - отсутствует (void);
            - Модификатор доступа - открытый;
            - Параметры - отсутствуют.

Класс cl_7:

- Свойства (поля):
    - Поля, хранящее наименование объекта:
        - Наименование - name;
        - Тип - строка;
        - Модификатор доступа - закрытый;
    - Методы:
        - Параметризованный конструктор:
            - Функционал - создание объекта с заданным наименованием;
            - Возвращаемое значение - объект класса cl_7;
            - Модификатор доступа - открытый;
            - Параметры:

            - name - наименование объекта (строка);
        - Метод print:
            - Функционал - вывод наименования объекта;
            - Возвращаемое значение - отсутствует (void);
            - Модификатор доступа - открытый;
            - Параметры - отсутствуют.

Класс cl_8:

- Свойства (поля):
    - Поля, хранящее наименование объекта:
        - Наименование - name;
        - Тип - строка;
        - Модификатор доступа - закрытый;
    - Методы:
        - Параметризованный конструктор:
            - Функционал - создание объекта с заданным наименованием;
            - Возвращаемое значение - объект класса cl_8;
            - Модификатор доступа - открытый;
            - Параметры:
                - name - наименование объекта (строка);
        - Метод print:
            - Функционал - вывод наименования объекта;
            - Возвращаемое значение - отсутствует (void);
            - Модификатор доступа - открытый;
            - Параметры - отсутствуют.

Таблица иерархии классов:

Таблица 1 – Иерархия наследования классов

Номер	**Имя класса**	**Классы-наследники**	**Модификатор доступа при наследовании**	**Описание**	**Номер**
1	cl_1			Базовый класс в иерархии наследования	
		cl_2	public		2
		cl_3	public		3
		cl_4	public		4
		cl_5	public		5
2	cl_2			Наследник класса cl_1	
		cl_6	public		6
3	cl_3			Наследник класса cl_1	
		cl_6	public		6
4	cl_4			Наследник класса cl_1	
		cl_7	public		7
5	cl_5			Наследник класса cl_1	
		cl_7	public		7
6	cl_6			Наследник классов cl_2 и cl_3	
		cl_8	public		8
7	cl_7			Наследник классов cl_4 и cl_5	
		cl_8	public		8
8	cl_8			Наследник классов cl_6 и cl_7	

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 функции main

Функционал: основной алгоритм программы.

Параметры: отсутствуют.

Возвращаемое значение: целочисленный код завершения работы программы.

Алгоритм функции представлен в таблице 2.

Таблица 2 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Объявление строковой переменной name	2
2		Ввод значения переменной name	3
3		Инициализация указателя на объект класса cl_8 obj адресом нового объекта, созданного с помощью оператора new с использованием параметризованного конструктора со значением переменной name в качестве параметра	4
4		Вызов метода print объекта, на который указывает указатель obj, приведённый к типу указателя на объект класса cl_6, после чего приведённый к типу указателя на объект класса cl_2, после чего приведённый к типу указателя на объект класса cl_1	5
5		Вывод строки "\n"	6
6		Вызов метода print объекта, на который указывает указатель obj, приведённый к типу указателя на объект класса cl_3, после чего приведённый к типу указателя на объект класса cl_1	7

№	Предикат	Действия	№ перехода
7		Вывод строки "\n"	8
8		Вызов метода print объекта, на который указывает указатель obj, приведённый к типу указателя на объект класса cl_4, после чего приведённый к типу указателя на объект класса cl_1	9
9		Вывод строки "\n"	10
10		Вызов метода print объекта, на который указывает указатель obj, приведённый к типу указателя на объект класса cl_5, после чего приведённый к типу указателя на объект класса cl_1	11
11		Вывод строки "\n"	12
12		Вызов метода print объекта, на который указывает указатель obj, приведённый к типу указателя на объект класса cl_6, после чего приведённый к типу указателя на объект класса cl_2	13
13		Вывод строки "\n"	14
14		Вызов метода print объекта, на который указывает указатель obj, приведённый к типу указателя на объект класса cl_6, после чего приведённый к типу указателя на объект класса cl_3	15
15		Вывод строки "\n"	16
16		Вызов метода print объекта, на который указывает указатель obj, приведённый к типу указателя на объект класса cl_7, после чего приведённый к типу указателя на объект класса cl_4	17
17		Вывод строки "\n"	18
18		Вызов метода print объекта, на который указывает указатель obj, приведённый к типу указателя на объект класса cl_7, после чего приведённый к типу указателя на объект класса cl_5	19
19		Вывод строки "\n"	20
20		Вызов метода print объекта, на который указывает указатель obj, приведённый к типу указателя на объект класса cl_6	21
21		Вывод строки "\n"	22
22		Вызов метода print объекта, на который указывает указатель obj,	23

№	Предикат	Действия	№ перехода
		приведённый к типу указателя на объект класса cl_7	
23		Вывод строки "\n"	24
24		Вызов метода print объекта, на который указывает указатель obj	25
25		Удаленние объекта, на который указывает указатель obj с помощью оператора delete	∅

3.2 Алгоритм конструктора класса cl_1

Функционал: создание объекта с заданным наименованием.

Параметры: name - наименование объекта (строка).

Алгоритм конструктора представлен в таблице 3.

Таблица 3 – Алгоритм конструктора класса cl_1

№	Предикат	Действия	№ перехода
1		Присваивание полю name значения суммы значения параметра name и строки "_1"	∅

3.3 Алгоритм метода print класса cl_1

Функционал: вывод наименования объекта.

Параметры: отсутствуют.

Возвращаемое значение: отсутствует (void).

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода print класса cl_1

№	Предикат	Действия	№ перехода
1		Вывод значения поля name	∅

3.4 Алгоритм конструктора класса cl_2

Функционал: создание объекта с заданным наименованием.

Параметры: name - наименование объекта (строка).

Алгоритм конструктора представлен в таблице 5.

Таблица 5 – Алгоритм конструктора класса cl_2

№	Предикат	Действия	№ перехода
1		Присваивание полю name значения суммы значения параметра name и строки "_2"	∅

3.5 Алгоритм метода print класса cl_2

Функционал: вывод наименования объекта.

Параметры: отсутствуют.

Возвращаемое значение: отсутствует (void).

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода print класса cl_2

№	Предикат	Действия	№ перехода
1		Вывод значения поля name	∅

3.6 Алгоритм конструктора класса cl_3

Функционал: создание объекта с заданным наименованием.

Параметры: name - наименование объекта (строка).

Алгоритм конструктора представлен в таблице 7.

Таблица 7 – Алгоритм конструктора класса cl_3

№	Предикат	Действия	№ перехода
1		Присваивание полю name значения суммы значения параметра name и строки "_3"	∅

3.7 Алгоритм метода print класса cl_3

Функционал: вывод наименования объекта.

Параметры: отсутствуют.

Возвращаемое значение: отсутствует (void).

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода print класса cl_3

№	Предикат	Действия	№ перехода
1		Вывод значения поля name	∅

3.8 Алгоритм конструктора класса cl_4

Функционал: создание объекта с заданным наименованием.

Параметры: name - наименование объекта (строка).

Алгоритм конструктора представлен в таблице 9.

Таблица 9 – Алгоритм конструктора класса cl_4

№	Предикат	Действия	№ перехода
1		Присваивание полю name значения суммы значения параметра name и строки "_4"	∅

3.9 Алгоритм метода print класса cl_4

Функционал: вывод наименования объекта.

Параметры: отсутствуют.

Возвращаемое значение: отсутствует (void).

Алгоритм метода представлен в таблице 10.

Таблица 10 – Алгоритм метода print класса cl_4

№	Предикат	Действия	№ перехода
1		Вывод значения поля name	∅

3.10 Алгоритм конструктора класса cl_5

Функционал: создание объекта с заданным наименованием.

Параметры: name - наименование объекта (строка).

Алгоритм конструктора представлен в таблице 11.

Таблица 11 – Алгоритм конструктора класса cl_5

№	Предикат	Действия	№ перехода
1		Присваивание полю name значения суммы значения параметра name и строки "_5"	∅

3.11 Алгоритм метода print класса cl_5

Функционал: вывод наименования объекта.

Параметры: отсутствуют.

Возвращаемое значение: отсутствует (void).

Алгоритм метода представлен в таблице 12.

Таблица 12 – Алгоритм метода print класса cl_5

№	Предикат	Действия	№ перехода
1		Вывод значения поля name	∅

3.12 Алгоритм конструктора класса cl_6

Функционал: создание объекта с заданным наименованием.

Параметры: name - наименование объекта (строка).

Алгоритм конструктора представлен в таблице 13.

Таблица 13 – Алгоритм конструктора класса cl_6

№	Предикат	Действия	№ перехода
1		Присваивание полю name значения суммы значения параметра name и строки "_6"	∅

3.13 Алгоритм метода print класса cl_6

Функционал: вывод наименования объекта.

Параметры: отсутствуют.

Возвращаемое значение: отсутствует (void).

Алгоритм метода представлен в таблице 14.

Таблица 14 – Алгоритм метода print класса cl_6

№	Предикат	Действия	№ перехода
1		Вывод значения поля name	∅

3.14 Алгоритм конструктора класса cl_7

Функционал: создание объекта с заданным наименованием.

Параметры: name - наименование объекта (строка).

Алгоритм конструктора представлен в таблице 15.

Таблица 15 – Алгоритм конструктора класса cl_7

№	Предикат	Действия	№ перехода
1		Присваивание полю name значения суммы значения параметра name и строки "_7"	∅

3.15 Алгоритм метода print класса cl_7

Функционал: вывод наименования объекта.

Параметры: отсутствуют.

Возвращаемое значение: отсутствует (void).

Алгоритм метода представлен в таблице 16.

Таблица 16 – Алгоритм метода print класса cl_7

№	Предикат	Действия	№ перехода
1		Вывод значения поля name	∅

3.16 Алгоритм конструктора класса cl_8

Функционал: создание объекта с заданным наименованием.

Параметры: name - наименование объекта (строка).

Алгоритм конструктора представлен в таблице 17.

Таблица 17 – Алгоритм конструктора класса cl_8

№	Предикат	Действия	№ перехода
1		Присваивание полю name значения суммы значения параметра name и строки "_8"	∅

3.17 Алгоритм метода print класса cl_8

Функционал: вывод наименования объекта.

Параметры: отсутствуют.

Возвращаемое значение: отсутствует (void).

Алгоритм метода представлен в таблице 18.

Таблица 18 – Алгоритм метода print класса cl_8

№	Предикат	Действия	№ перехода
1		Вывод значения поля name	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-21.

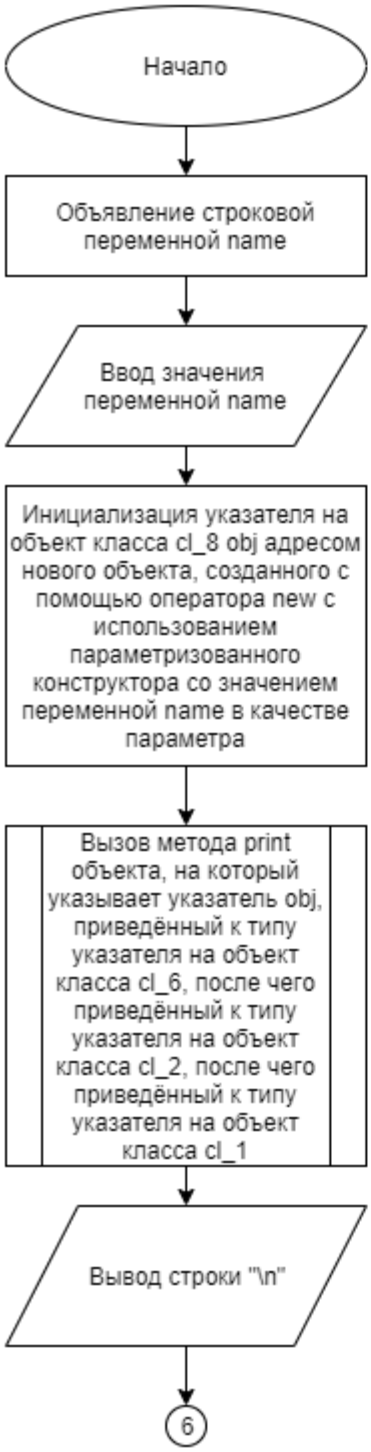

Рисунок 1 – Блок-схема алгоритма

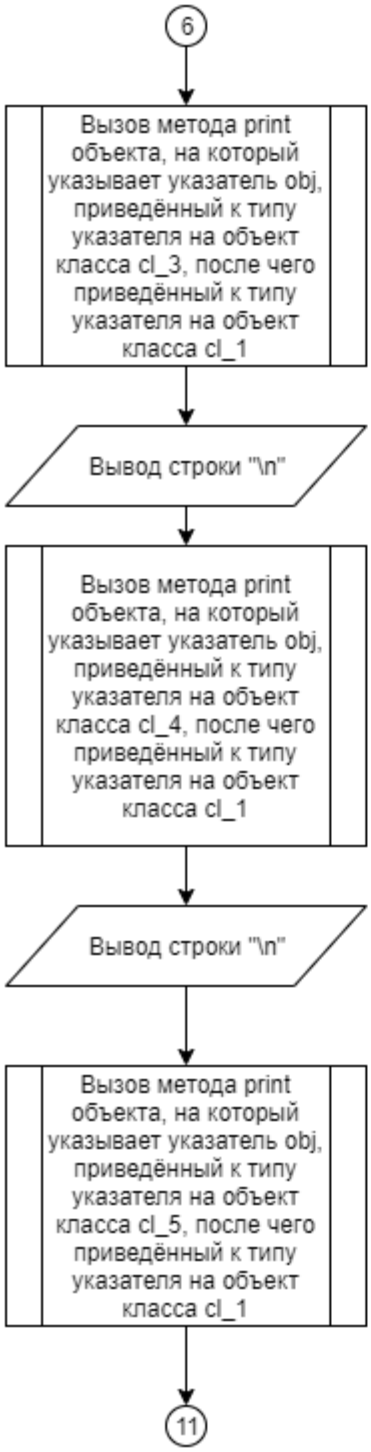

Рисунок 2 – Блок-схема алгоритма

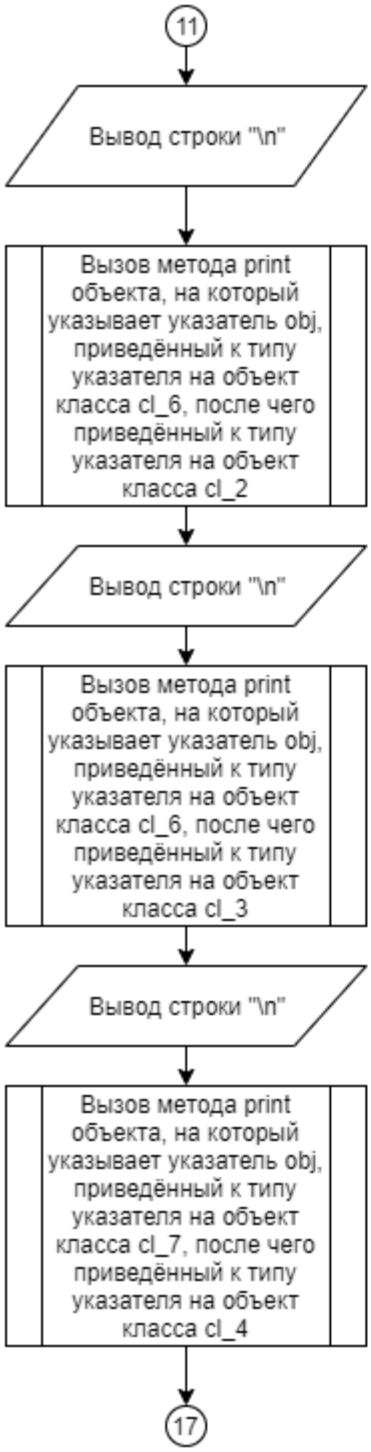

Рисунок 3 – Блок-схема алгоритма

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

Рисунок 10 – Блок-схема алгоритма

Рисунок 11 – Блок-схема алгоритма

Рисунок 12 – Блок-схема алгоритма

Рисунок 13 – Блок-схема алгоритма

Рисунок 14 – Блок-схема алгоритма

Рисунок 15 – Блок-схема алгоритма

Рисунок 16 – Блок-схема алгоритма

Рисунок 17 – Блок-схема алгоритма

Рисунок 18 – Блок-схема алгоритма

Рисунок 19 – Блок-схема алгоритма

Рисунок 20 – Блок-схема алгоритма

Рисунок 21 – 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(std::string name)
{
   this->name = name + "_1";
}

void cl_1::print()
{
   std::cout << name;
}
```

5.2 Файл cl_1.h

Листинг 2 – cl_1.h

```
#ifndef __CL_1__H
#define __CL_1__H

#include <iostream>
#include <string>

class cl_1
{
private:
   std::string name;
public:
   cl_1(std::string name);
   void print();
};

#endif
```

5.3 Файл cl_2.cpp

Листинг 3 – cl_2.cpp

```
#include "cl_2.h"

cl_2::cl_2(std::string name) : cl_1(name + "_2")
{
   this->name = name + "_2";
}

void cl_2::print()
{
   std::cout << name;
}
```

5.4 Файл cl_2.h

Листинг 4 – cl_2.h

```
#ifndef __CL_2__H
#define __CL_2__H

#include <iostream>
#include <string>
#include "cl_1.h"

class cl_2 : public cl_1
{
private:
   std::string name;
public:
   cl_2(std::string name);
   void print();
};

#endif
```

5.5 Файл cl_3.cpp

Листинг 5 – cl_3.cpp

```
#include "cl_3.h"
```

```
cl_3::cl_3(std::string name) : cl_1(name + "_3")
{
   this->name = name + "_3";
}

void cl_3::print()
{
   std::cout << name;
}
```

5.6 Файл cl_3.h

Листинг 6 – cl_3.h

```
#ifndef __CL_3__H
#define __CL_3__H

#include <iostream>
#include <string>
#include "cl_1.h"

class cl_3 : public cl_1
{
private:
   std::string name;
public:
   cl_3(std::string name);
   void print();
};

#endif
```

5.7 Файл cl_4.cpp

Листинг 7 – cl_4.cpp

```
#include "cl_4.h"

cl_4::cl_4(std::string name) : cl_1(name + "_4")
{
   this->name = name + "_4";
}

void cl_4::print()
```

```
{
   std::cout << name;
}
```

5.8 Файл cl_4.h

Листинг 8 – cl_4.h

```
#ifndef __CL_4__H
#define __CL_4__H

#include <iostream>
#include <string>
#include "cl_1.h"

class cl_4 : virtual public cl_1
{
private:
   std::string name;
public:
   cl_4(std::string name);
   void print();
};

#endif
```

5.9 Файл cl_5.cpp

Листинг 9 – cl_5.cpp

```
#include "cl_5.h"

cl_5::cl_5(std::string name) : cl_1(name + "_5")
{
   this->name = name + "_5";
}

void cl_5::print()
{
   std::cout << name;
}
```

5.10 Файл cl_5.h

Листинг 10 – cl_5.h

```
#ifndef __CL_5__H
#define __CL_5__H

#include <iostream>
#include <string>
#include "cl_1.h"

class cl_5 : virtual public cl_1
{
private:
   std::string name;
public:
   cl_5(std::string name);
   void print();
};

#endif
```

5.11 Файл cl_6.cpp

Листинг 11 – cl_6.cpp

```
#include "cl_6.h"

cl_6::cl_6(std::string name)
   : cl_2(name + "_6"), cl_3(name + "_6")
{
   this->name = name + "_6";
}

void cl_6::print()
{
   std::cout << name;
}
```

5.12 Файл cl_6.h

Листинг 12 – cl_6.h

```
#ifndef __CL_6__H
```

```
#define __CL_6__H

#include <iostream>
#include <string>
#include "cl_2.h"
#include "cl_3.h"

class cl_6 : public cl_2, public cl_3
{
private:
   std::string name;
public:
   cl_6(std::string name);
   void print();
};

#endif
```

5.13 Файл cl_7.cpp

Листинг 13 – cl_7.cpp

```
#include "cl_7.h"

cl_7::cl_7(std::string name)
   : cl_4(name + "_7"), cl_5(name + "_7"), cl_1(name + "_7")
{
   this->name = name + "_7";
}

void cl_7::print()
{
   std::cout << name;
}
```

5.14 Файл cl_7.h

Листинг 14 – cl_7.h

```
#ifndef __CL_7__H
#define __CL_7__H

#include <iostream>
#include <string>
```

```
#include "cl_4.h"
#include "cl_5.h"

class cl_7 : public cl_4, public cl_5
{
private:
   std::string name;
public:
   cl_7(std::string name);
   void print();
};

#endif
```

5.15 Файл cl_8.cpp

Листинг 15 – cl_8.cpp

```
#include "cl_8.h"

cl_8::cl_8(std::string name)
   : cl_6(name + "_8"), cl_7(name + "_8"), cl_1(name + "_8")
{
   this->name = name + "_8";
}

void cl_8::print()
{
   std::cout << name;
}
```

5.16 Файл cl_8.h

Листинг 16 – cl_8.h

```
#ifndef __CL_8__H
#define __CL_8__H

#include <iostream>
#include <string>
#include "cl_6.h"
#include "cl_7.h"

class cl_8 : public cl_6, public cl_7
{
```

```
private:
   std::string name;
public:
   cl_8(std::string name);
   void print();
};

#endif
```

5.17 Файл main.cpp

Листинг 17 – main.cpp

```
#include <iostream>
#include <string>
#include "cl_1.h"
#include "cl_6.h"
#include "cl_7.h"
#include "cl_8.h"

int main()
{
   // Объявление строковой переменной nmae
   std::string name;
   // Ввод значения переменной name
   std::cin >> name;

   // Инициализация указателя на объект
   // класса cl_8
   cl_8* obj = new cl_8(name);

   // Вызов метода print объекта, на который
   // указывает указатель obj.
   // приведённый последовательно к типам
   // cl_6, cl_2, cl_1
   ((cl_1*)(cl_2*)(cl_6*)obj)->print();
   // Вывод "\n" (переход на новую строку)
   std::cout << '\n';

   // Вызов метода print объекта, на который
   // указывает указатель obj.
   // приведённый последовательно к типам
   // cl_3, cl_1
   ((cl_1*)(cl_3*)obj)->print();
   // Вывод "\n" (переход на новую строку)
   std::cout << '\n';

   // Вызов метода print объекта, на который
   // указывает указатель obj.
   // приведённый последовательно к типам
```

```
        // cl_4, cl_1
        ((cl_1*)(cl_4*)obj)->print();
        // Вывод "\n" (переход на новую строку)
        std::cout << '\n';

        // Вызов метода print объекта, на который
        // указывает указатель obj.
        // приведённый последовательно к типам
        // cl_5, cl_1
        ((cl_1*)(cl_5*)obj)->print();
        // Вывод "\n" (переход на новую строку)
        std::cout << '\n';

        // Вызов метода print объекта, на который
        // указывает указатель obj.
        // приведённый последовательно к типам
        // cl_6, cl_2
        ((cl_2*)(cl_6*)obj)->print();
        // Вывод "\n" (переход на новую строку)
        std::cout << '\n';

        // Вызов метода print объекта, на который
        // указывает указатель obj.
        // приведённый последовательно к типам
        // cl_6, cl_3
        ((cl_3*)(cl_6*)obj)->print();
        // Вывод "\n" (переход на новую строку)
        std::cout << '\n';

        // Вызов метода print объекта, на который
        // указывает указатель obj.
        // приведённый последовательно к типам
        // cl_7, cl_4
        ((cl_4*)(cl_7*)obj)->print();
        // Вывод "\n" (переход на новую строку)
        std::cout << '\n';
        // Вызов метода print объекта, на который
        // указывает указатель obj.
        // приведённый последовательно к типам
        // cl_7, cl_5
        ((cl_5*)(cl_7*)obj)->print();
        // Вывод "\n" (переход на новую строку)
        std::cout << '\n';

        // Вызов метода print объекта, на который
        // указывает указатель obj.
        // приведённый последовательно к типу
        // cl_6
        ((cl_6*)obj)->print();
        // Вывод "\n" (переход на новую строку)
        std::cout << '\n';

        // Вызов метода print объекта, на который
        // указывает указатель obj.
        // приведённый последовательно к типу
```

```
        // cl_7
        ((cl_7*)obj)->print();
        // Вывод "\n" (переход на новую строку)
        std::cout << '\n';

        // Вызов метода print объекта, на который
        // указывает указатель obj
        obj->print();

        // Удаление объекта, на который указывает
        // указтель obj
        delete obj;
        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).