**ЛАБОРАТОРНАЯ РАБОТА 2**

**ИССЛЕДОВАНИЕ МЕХАНИЗМА ОДИНОЧНОГО НАСЛЕДОВАНИЯ КЛАССОВ**

**Цель лабораторной работы -** исследовать механизм одиночного наследования классов

**Цели.**

1. Исследовать механизм одиночного наследования классов.
2. Написать и отладить программы на языке С++, которые демонстрируют применение механизма одиночного наследования классов (табл.9).

Таблица 8

**Варианты заданий**

| *№* | *Базовый класс* | *Производный класс* | *Задание* | |
| --- | --- | --- | --- | --- |
| *Наследование с атрибутом* private | *Наследование с атрибутом* public |
|  | Линия: координаты начала і конца, конструктор с параметрами, метод вычисления длины линии | Отрезок:  конструктор с параметрами, метод вычисления угла между отрезком и осью ОY, метод вывода отрезка | Описание объекта,  вычисление и вывод его длины и угла с осью ОY,  вывод объекта | |
|  | Строка:  строка, конструктор с параметром, метод вычисления длины строки | Цифровая строка:  конструктор с параметром, метод обращения цифровой строки, метод вывода строки | Описание объекта,  вычисление и вывод его длины,  вывод объекта | |
|  | Фигура:  координаты вершин,  конструктор с параметрами,  метод вычисления длины стороны | Треугольник:  конструктор с параметрами, методы вычисления площади и периметра, метод вывода треугольника | Описание объекта,  вычисление и вывод его площади и периметра,  вывод объекта | |
|  | Линия:  координаты начала и конца, конструктор с параметрами, метод вычисления длины линии | Отрезок:  конструктор с параметрами, метод вычисления угла между отрезком и осью ОХ, метод вывода отрезка | Описание объекта,  вычисление и вывод его длины и угла с осью ОX,  вывод объекта | |
|  | Строка:  строка, конструктор с параметром, метод вычисления длины строки | Буквенная строка:  конструктор с параметром, метод сортирования буквенной строки по возрастанию, метод вывода строки | Описание объекта,  вычисление и вывод его длины,  вывод объекта | |
|  | Фигура:  координаты вершин, конструктор с параметрами,  метод вычисления длины стороны | Прямоугольник:  конструктор с параметрами, методы вычисления площади и периметра, метод вывода прямоугольника | Описание объекта,  вычисление и вывод его площади и периметра,  вывод объекта | |

Окончание табл. 8

|  | Линия:  координаты начала и конца, конструктор с параметрами, метод вычисления длины линии | Отрезок:  конструктор с параметрами, метод увеличения отрезка в 2 раза, метод вывода отрезка | Описание объекта,  вычисление и вывод его длины,  вывод объекта |
| --- | --- | --- | --- |
|  | Строка:  строка, конструктор с параметром, метод вычисления длины строки | Символьная строка:  конструктор с параметром, метод обмена символа '? ' на символы '\*\*', метод вывода строки | Описание объекта,  вычисление и вывод его длины,  вывод объекта |
|  | Фигура:  координаты вершин, конструктор с параметрами, метод вычисления длины стороны | Квадрат:  конструктор с параметрами, методы вычисления площади и периметра, метод вывода квадрата | Описание объекта,  вычисление и вывод его площади и периметра,  вывод объекта |
|  | Линия:  координаты начала и конца, конструктор с параметрами, метод вычисления длины линии | Отрезок:  конструктор с параметрами, метод уменьшения отрезка на 5 единиц, метод вывода отрезка | Описание объекта,  вычисление и вывод его длины,  вывод объекта |
|  | Строка:  строка, конструктор с параметром, метод вычисления длины строки | Буквенная строка:  конструктор с параметром, метод циклического сдвига строки в правую сторону на один символ, то есть последний символ на первое место, метод вывода строки | Описание объекта,  вычисление и вывод его длины,  вывод объекта |
|  | Фигура:  координаты вершин, конструктор с параметрами, метод вычисления длины стороны | Трапеция:  конструктор с параметрами, методы вычисления площади и периметра, метод вывода трапеции | Описание объекта,  вычисление и вывод его площади и периметра,  вывод объекта |
|  | Строка:  строка, конструктор с параметром, метод вычисления длины строки | Цифровая строка:  конструктор с параметром, метод удаления с строки символа ’2', метод вывода строки | Описание объекта,  вычисление и вывод его длины,  вывод объекта |
|  | Фигура:  координаты вершин, конструктор с параметрами, метод вычисления длины стороны | Ромб:  конструктор с параметрами, методы вычисления площади и периметра, метод вывода ромба | Описание объекта,  вычисление и вывод его площади и периметра,  вывод объекта |
|  | Строка:  строка, конструктор с параметром, метод вычисления длины строки | Символьная строка:  конструктор с параметром, метод сортирования строки по убыванию, метод вывода строки | Описание объекта,  вычисление и вывод его длины,  вывод объекта |

**Теоретические сведения**

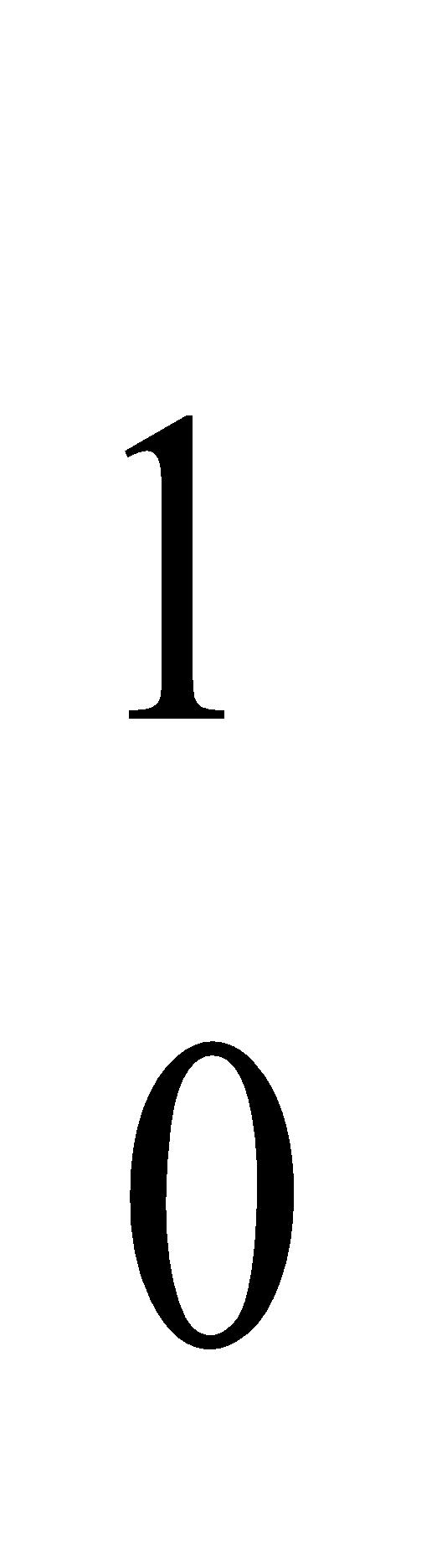
**Описание базового класса:**

<описание базового класса> ::= <описание класса>

**Описание производного класса:**

<описание производного класса> ::= class <пометка класса> : <атрибут доступа>

<пометка базового класса> {<тело производного класса>};

<атрибут доступа> ::= {private} | public | protected

<пометка базового класса> ::= <идентификатор>

Если базовый класс наследуется с атрибутом доступа public, то элементы*public* базового класса становятся элементами *public* производного класса. Элементы *protected* базового класса становятся элементами *protected*производного класса. Элементы *private* базового класса не наследуются.

Если базовый класс наследуется с атрибутом доступа private, то элементы *public* и *protected* базового класса становятся элементами *private* производного класса. Элементы *private* базового класса не наследуются.

Если базовый класс наследуется с атрибутом доступа protected, то элементы*public* и *protected* базового класса становятся элементами *protected* производного класса. Элементы *private* базового класса не наследуются.

**Явный вызов <конструктора> базового класса:**

<вызов конструктора базового класса> ::= <пометка производного класса>

(<список формальных параметров базовый и производного классов>) :

<пометка базового класса>(<список фактических параметров базового класса>)

{<тело конструктора производного класса>}

**Методические указания**

Необходимо помнить, что <конструктор> и <деструктор> базового класса не наследуются.

При создании объекта производного класса всегда сначала вызывается <конструктор> базового класса, а потом <конструктор> производного класса. Поэтому для инициирования закрытых элементов базового класса необходимо явным образом обращаться к <конструктору> базового класса (через список инициализации см. пример), если этого не сделать, то будет вызван конструктор по умолчанию для базового класса.

Таблица 9

| Наследование | | | | | |
| --- | --- | --- | --- | --- | --- |
| Атрибут доступа private | | Атрибут доступа public | | Атрибут доступа protected | |
| Использование  конструкторов | Использование  Методов базового класса | Использование  Конструкторов | Использование  методов базового класса | Использование  конструкторов | Использование  методов базового класса |

Пример:

// Секция описания класса обычная строка

//////////////////////////////////////////////////////////////////////

class SimpleString

{

public:

SimpleString(char\* data);

virtual ~SimpleString();

int getLength();//метод определения длины

protected:

char\* m\_data;//член данных хранит указатель на строку

};

SimpleString::SimpleString(char\* data):m\_data(NULL)

{

//Если задана не пустая строка то копируем ее

// и сохраняем указатель на вновь полученную

//строку-копию в члене класса m\_data

if (data)

m\_data=strdup(data);

}

//Деструктор освобождает память занимаемую строкой

SimpleString::~SimpleString()

{

free(m\_data);

}

//Метод определения длины строки

int SimpleString::getLength()

{

//считаем количество символов в строке

//для перемещения по строке используем указатель

//pos перемещаемся до тех пор пока не достигнем

//значения в ячейке памяти равное 0 т.е. конец строки

//Реализация опущена!!!

}

//класс буквенная строка

class LetterString : public SimpleString

{

public:

LetterString(char\*);//конструктор

void DirectSort();//сортировка строки

void PrintString();//метод вывода строки

};

//конструктор в списке инициализации вызывает конструктор базового

//класса

LetterString::LetterString(char\* data):SimpleString(data)

{

char\* buf=m\_data;

//удаляем все символы кроме букв A-Za-zА-Яа-я и пробела

//Реализация опущена!!!

}

/\*

\*сортировка с изменением текущей строки

\*/

void LetterString::DirectSort()

{

………………………………………………………….

}

//метод вывода

void LetterString::PrintString()

{

printf(m\_data);

}

// Главная программа.

//

int main(int argc, char\* argv[])

{

//Создаем строку передаем в конструктор как буквы

//так и цифры чтобы показать что цифры будут удалены

//в теле конструктора

LetterString\* str=new LetterString("BcaCfd4j");

printf("Initial string: ");

str->PrintString();

printf("\nString length: %d",str->getLength());

printf("\nSortedString: ");

str->DirectSort();

str->PrintString();

delete str;//удаляем строку т.к. мы создали ее динамически

return 0;//код возврата из программы

}

Примечание:

В примере показана частичная реализация примера наследования с атрибутом доступа public, вам необходимо также разработать аналогичные реализации с атрибутами private и protected.

**Контрольные вопросы**

1. В чем состоит сущность механизма наследования?
2. Расскажите какие бывают типы наследования.
3. Объясните роль атрибута доступа в наследовании.
4. Приведите примеры использования методов базового класса с атрибутом доступа public.
5. Приведите примеры использования методов базового класса с атрибутом доступа private.