МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра автоматизированных систем управления

Отчет по лабораторной работе №3.

По ОБЪЕКТО-ОРИЕНТИРОВАННОМУ ПРОГРАММИРОВАНИЮ

«Наследование и полиморфизм»

Выполнила:

студентка гр. ПМ-21-2 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бутусова В.М.

Проверил:

доц., к.п.н. кафедры АСУ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кургасов В. В.

Липецк 2022**Цель работы:**

Изучить механизмы наследования и полиморфизма с использованием виртуальных функций.

**Задание кафедры:**

Реализовать на языке С++ концепцию векторной функции скалярного аргумента (с привлечением механизма наследования и виртуальных функций) вида:

,

где – параметры элемента вектор-функции;

x – скалярный аргумент вектор-функции;

– функция вычисления одного элемента вектор-функции (выбирается пользователем из заданного множества функций);

N – размерность вектор-функции, заданная пользователем.

Количество элементов вектора результата и аргумент *х*, для которого производится вычисление, задаётся пользователем. Вид функции, по которому вычисляется значение каждого элемента вектора результата, указывается пользователем из заданного множества параметризованных функций (табл.). Параметры функций вводятся пользователем для каждого элемента вектора результата отдельно, при формировании вектора функций.

Элементы векторной функции скалярного аргумента:

|  |  |  |
| --- | --- | --- |
| Вариант | Функция 1 | Функция 2 |
| 3 |  |  |

**Текст программы:**

#include<iostream>

#include<vector>

using namespace std;

class VectorFuncia {// Базовый абстрактный класс, являющийся интерфейсом для производного класса Funkcia

protected:

int N;//размер вектор-функции

double x;

int n;

vector <double> funcElems;//вектор для хранения значений вектор-функции

double coefs[3];

public:

void Funcia() {//функция для определения размерности вектор-функции и значения х

cout << "Введите размерность вектор-функции: ";

cin >> N;

funcElems.resize(N);//переопределение размера массива значений вектор-функции

cout << endl;

}

void Cout() {//Вывод значений вектор-функции

for (int i = 0; i < N; i++)

cout << funcElems[i] << endl;

}

virtual void elems() = 0; //виртуальная функция для заполнения вектора элементами вектор-функции

virtual void calculate(double x) = 0;//Объявление виртуальной функции, которая делает расчёт элементов вектор-функции

};

class Funkcia : public VectorFuncia {//класс наследуемый методы и функции класса VectorFuncia

public:

void elems() override

{

cout << "Выберите функцию:" << endl << "1, если a\* sin(b \* x)" << endl << "2, если a\* lg(b \* x) + c" << endl;

cin >> n;

int i = 0;

if (n == 1)

{

cout << endl << "Введите a и b для " << i + 1 << " функции: ";

cin >> coefs[0] >> coefs[1];

cout << endl;

i++;

}

else

{

cout << endl << "Введите a, b и c для" << i + 1 << " функции: ";

cin >> coefs[0] >> coefs[1] >> coefs[2];

cout << endl;

i++;

}

}

void calculate(double x) override

{

for (int i = 0; i < N; i++)

{

elems();

//вычисление значения i-й строки вектор функции

if (n == 1)

{

funcElems[i] = coefs[0] \* sin(coefs[1] \* x);

}

else

{

funcElems[i] = coefs[0] \* log10(coefs[1] \* x) + coefs[2];

}

}

}

};

int main() {

setlocale(LC\_ALL, "rus");

Funkcia A;

A.Funcia();

cout << "Введите скалярный аргумент X: ";

int x;

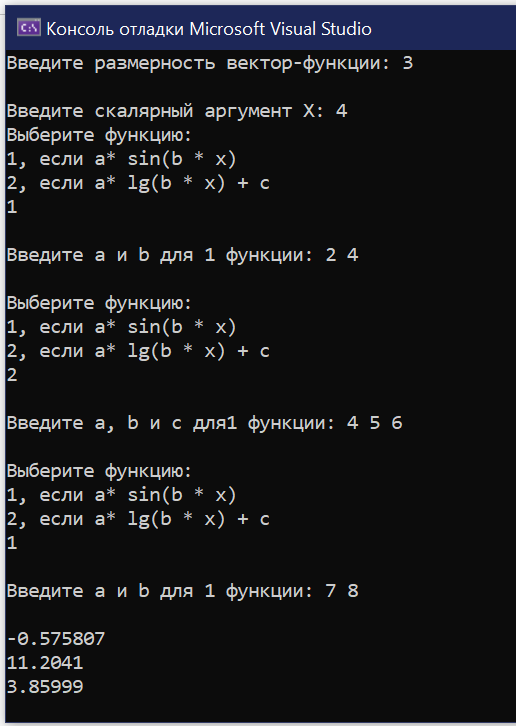
cin >> x;

A.calculate(x);

A.Cout();

}

**Пример работы программы:**

****

**Вывод:** изучила механизмы наследования и полиморфизма с использованием виртуальной функции и реализовала на языке C++ концепцию векторной функции скалярного аргумента.

**Контрольные вопросы:**

1. **Зачем нужен механизм наследования?**

Благодаря механизму наследования становится возможным создать новый класс на основе существующего (унаследовав свойства и методы этого класса), лишь переписав отличающийся функционал.

1. **Зачем используются модификаторы при наследовании классов? Какие это модификаторы?**

Модификаторы доступа private, protected, public.

При наследовании с модификатором public все данные родительского класса в уровнях доступа public и protected наследуются без изменения уровня доступа к ним;

При наследовании с модификатором protected все унаследованные данные становятся защищёнными (все заносятся в уровень доступа protected);

При наследовании с модификатором private все унаследованные данные становятся приватными (заносятся в уровень доступа private)

1. **Зачем нужен механизм полиморфизма?**

Полиморфизм **помогает снижать сложность программ, разрешая использование того же интерфейса для задания единого класса действий**. **Полиморфизм** позволяет писать более абстрактные программы и повысить коэффициент повторного использования кода.

1. **Что понимается под динамическим полиморфизмом?**

Динамический полиморфизм заключается в использовании виртуальных функций, которые определяются в базовом классе и могут быть переопределены в классе-наследнике. То есть в наследнике переопределённая виртуальная функция будет иметь реализацию, отличную от той, что определена в родительском классе.

1. **К какому из видов полиморфизма относится перегрузка оператора?**

Перегрузка оператора относится к статическому полиморфизму.

1. **Что такое интерфейс класса?**

**Интерфейс** — структура программы/синтаксиса, определяющая отношение с объектами, объединенными только некоторым поведением. При проектировании **классов**, разработка **интерфейса** тождественна разработке спецификации (множества методов, которые каждый **класс**, использующий **интерфейс**, должен реализовывать).

1. **Зачем нужен чисто виртуальный метод? Как он выглядит?**

Чисто виртуальный метод является функцией, которая объявляется в базовом классе, но не имеет в нем определения. Поскольку она не имеет определения, то есть  
тела в этом базовом классе, то всякий производный класс обязан иметь свою собственную версию определения.

Для объявления чисто виртуальной функции используется следующая общая форма:

**virtual тип\_возвращаемого\_значения имя\_функции(список параметров) = 0;**

1. **Какой класс называется абстрактным?**

Абстрактный класс в объектно-ориентированном программировании — **базовый класс, который не предполагает создания экземпляров**.

Абстрактные классы - это классы, которые содержат или наследуют без переопределения хотя бы одну чистую виртуальную функцию. Абстрактный класс определяет интерфейс для переопределения производными классами.