|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет  имени Н. Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н. Э. Баумана)** |

|  |  |
| --- | --- |
| ФАКУЛЬТЕТ | «Информатика и системы управления» (ИУ) |

|  |  |
| --- | --- |
| КАФЕДРА | «Информационная безопасность» (ИУ8) |

Лабораторная работа № 3

ПО КУРСУ

«Алгоритмические языки»

на тему «**Изучение возможностей наследования классов**»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИУ8-23 |  |  |  | Д.С. Афанасьев |
|  | (Группа) |  |  |  | (И. О. Фамилия) |
|  |  |  |  |  |  |
| Преподаватель: |  |  |  |  | М. В. Малахов |
|  |  |  |  |  | (И.О. Фамилия) |

2024

Введение

Цели и задачи работы

Цель работы состоит в овладении навыками разработки программ на языке Си++, использующих возможности наследования классов для решения различных задач. Для достижения цели необходимо выполнить следующие задачи:

- изучить необходимые учебные материалы, посвященные наследованию классов в языке Си++;

- разработать программу на языке Си++ для решения заданного варианта задания;

- отладить программы;

- представить результаты работы программы;

- подготовить отчет по лабораторной работе.

Условия для 4 варианта

В работе первоначально необходимо создать базовый класс и на его основе создать производный класс. При этом необходимо использовать вызов конструктора базового класса внутри конструктора производного класса, а также возможности переопределения методов (функций) класса. Необходимо разобраться с понятием виртуального метода, а также статическим и динамическим полиморфизмом. Продемонстрировать данные возможности в выполняемом примере.

Создать базовый класс «прямоугольник». Элементы класса: поля, задающие длины сторон (статус доступа *protected*); конструктор для инициализации полей; функция для вычисления площади прямоугольника; функция для печати полей и значения площади. Создать производный класс «прямоугольный параллелепипед». Элементы класса: дополнительное поле, задающее высоту; конструктор для инициализации полей; переопределенная функция для вычисления объема (вместо площади) (внутри переопределенной функции должна вызываться функция из базового класса); переопределенная функция для печати полей и значения объема. Создать по 1 объекту каждого из классов. Показать вызов созданных функций. При переопределении функций обеспечить и продемонстрировать два варианта: статический полиморфизм и динамический полиморфизм.

Основная часть

Исходный текст программы:

Geometric\_figures.h:

#ifndef GEOMETRIC\_FIGURES\_H

#define GEOMETRIC\_FIGURES\_H

class Rectangle {

protected:

  double length;

  double width;

public:

  Rectangle();

  Rectangle(double length, double width);

  virtual double calculate\_area();

  virtual void print\_sidesValue();

  virtual void print\_areaValue();

};

class Cuboid : public Rectangle {

protected:

  double height;

public:

  Cuboid();

  Cuboid(double length, double width, double height);

  double calculate\_area() override;

  void print\_sidesValue() override;

  void print\_areaValue() override;

};

#endif

Geometric\_figures.cpp

#include "Geometric\_figures.h"

#include <iomanip>

#include <iostream>

Rectangle::Rectangle() : length(0), width(0) {}

Rectangle::Rectangle(double length, double width) : length(length), width(width) {}

double Rectangle::calculate\_area() {

  return width \* length;

}

void Rectangle::print\_sidesValue() {

  std::cout << std::fixed << std::setprecision(5)

            << "\tlength = " << length << std::endl

            << "\twidth = " << width << std::endl;

}

void Rectangle::print\_areaValue() {

  std::cout << std::fixed << std::setprecision(7)

            << "\tarea value: " << calculate\_area() << std::endl;

}

Cuboid::Cuboid() : Rectangle(), height(0) {}

Cuboid::Cuboid(double length, double width, double height) : Rectangle(length, width), height(height) {}

double Cuboid::calculate\_area() {

  return height \* Rectangle::calculate\_area();

}

void Cuboid::print\_sidesValue() {

  Rectangle::print\_sidesValue();

  std::cout << "\theight = " << height << std::endl;

}

void Cuboid::print\_areaValue() {

  std::cout << std::fixed << std::setprecision(7)

            << "\tarea value: " << Cuboid::calculate\_area() << std::endl;

}

lab3.cpp:

#include "Geometric\_figures.h"

*// g++ ./source/lab3/Geometric\_figures.cpp ./source/lab3/lab3.cpp -o ./build/lab3*

#include <fstream>

#include <iostream>

int main() {

  double length, width, height;

  std::cout << "Length = "; std::cin >> length;

  std::cout << "Width = "; std::cin >> width;

  std::cout << "Height = "; std::cin >> height;

  Rectangle\* rectangle = new Rectangle(length, width);

  Cuboid\* cuboid = new Cuboid(length, width, height);

  Cuboid cub(length, width, height);

  Rectangle\* rect1 = &cub;

  Rectangle rect2 = cub;

  std::cout << "\t\tRectangle parameters:\n";

*// rectangle->print\_sidesValue();*

*// rectangle->print\_areaValue();*

  rect2.print\_sidesValue();

  rect2.print\_areaValue();

  std::cout << "\t\tCuboid parameters:\n";

*// cuboid->print\_sidesValue();*

*// cuboid->print\_areaValue();*

  rect1->print\_sidesValue();

  rect1->print\_areaValue();

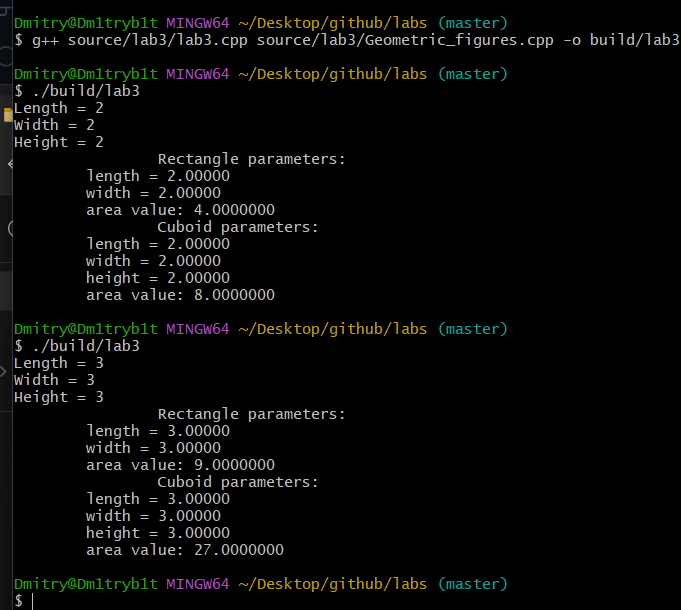
  delete rectangle;

  delete cuboid;

  return 0;

}

Снимки выполнения работы программы



Заключение

Задачи лабораторной работы были решены, результаты проверены. Приобретены навыки разработки программ на языке Си++, использующих возможности наследования классов для решения различных задач.