ОТЧЁТ

О ЛАБОРАТОРНОЙ РАБОТЕ №6  
по теме: Наследование, абстрактные классы, перегрузка функций

по дисциплине: Разработка программных модулей

09.02.07 Информационные системы и программирование

Проверил:

\_\_\_\_\_\_\_\_ Антонов М.Б.

Дата: «\_\_» \_\_\_\_\_ 20\_\_ г.

Оценка: \_\_\_\_\_\_\_\_\_\_

УНИВЕРСИТЕТ ИТМО

Факультет среднего профессионального образования

Санкт-Петербург 2019

Выполнил:

студент группы Y2333

\_\_\_\_\_\_\_\_\_\_\_Харченко Д.И.

1. ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Цель работы: написать программу, демонстрирующую работу с объектами двух типов. Классы: Triangle, Rectangle. Реализовать следующие функции (методы): move(x, y) – перемещает фигуру по двум координатам, согласно величинам, заданным с клавиатуры; compare(T1 \*obj1, T2 \*obj2) – сравнивает площади фигур.

Код main.cpp:

#include <iostream>

#include <math.h>

#include "Task.h"

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

int decision[2];

std::cout << "Выберите две фигуры: " << std::endl;

std::cout << "1 – Треугольник" << std::endl;

std::cout << "2 – Прямоугольник" << std::endl;

for (int i = 0; i < 2; i++) {

std::cin >> decision[i];

}

/// Описание 4-х возможных случаев: треугольник и прямоугольник, 2 треугольника, 2 прямоугольника, прямоугольник и треугольник

if (decision[0] == 1) {

std::cout << "Работаем с треугольником: " << std::endl;

float xs[3];

float ys[3];

for (int i = 0; i < 3; i++) {

std::cout << "Введите координаты " << i + 1 << " точки треугольника (x y)" << std::endl;

std::cin >> xs[i];

std::cin >> ys[i];

}

Triangle \*tr1 = new Triangle(xs[0], ys[0], xs[1], ys[1], xs[2], ys[2]); ///< создание фигуры

float deltaX1, deltaY1;

std::cout << "На сколько двигаем по X?" << std::endl; ///< направление смещения определяется знаком ("-" – влево/вниз, без знака – вправо/вверх)

std::cin >> deltaX1;

std::cout << "На сколько двигаем по Y?" << std::endl;

std::cin >> deltaY1;

tr1->move(deltaX1, deltaY1); ///< свдиг фигуры

tr1->toConsole(); ///< вывод фигуры в консоль

if (decision[1] == 2) {

std::cout << "Работаем с прямоугольником: " << std::endl;

float lx1, ly1;

float rx1, ry1;

std::cout << "Координаты левого верхнего угла: " << std::endl;

std::cin >> lx1;

std::cin >> ly1;

std::cout << "Координаты правого нижнего угла: " << std::endl;

std::cin >> rx1;

std::cin >> ry1;

Rectangle \*rt2 = new Rectangle(lx1, ly1, rx1, ry1); ///< создание фигуры

float deltaX2, deltaY2;

std::cout << "На сколько двигаем по X?" << std::endl;

std::cin >> deltaX2;

std::cout << "На сколько двигаем по Y?" << std::endl;

std::cin >> deltaY2;

rt2->move(deltaX2, deltaY2); ///< свдиг фигуры

rt2->toConsole(); ///< вывод фигуры в консоль

if (compare(tr1, rt2) > 0) { ///< сравнение площадей

std::cout << "Площадь треугольника больше площади прямоугольника" << std::endl;

} else {

if (compare(tr1, rt2) == 0) {

std::cout << "Площади равны" << std::endl;

} else {

std::cout << "Площадь треугольника меньше площади прямоугольника" << std::endl;

}

}

} else {

std::cout << "Работаем со вторым треугольником: " << std::endl;

float xs2[3];

float ys2[3];

for (int i = 0; i < 3; i++) {

std::cout << "Введите координаты " << i + 1 << " точки треугольника (x y)" << std::endl;

std::cin >> xs2[i];

std::cin >> ys2[i];

}

Triangle \*tr2 = new Triangle(xs2[0], ys2[0], xs2[1], ys2[1], xs2[2], ys2[2]); ///< создание фигуры

float deltaX2, deltaY2;

std::cout << "На сколько двигаем по X?" << std::endl;

std::cin >> deltaX2;

std::cout << "На сколько двигаем по Y?" << std::endl;

std::cin >> deltaY2;

tr2->move(deltaX2, deltaY2); ///< свдиг фигуры

tr2->toConsole(); ///< вывод фигуры в консоль

if (compare(tr1, tr2) > 0) { ///< сравнение площадей

std::cout << "Площадь первого треугольника больше площади второго треугольника" << std::endl;

} else {

if (compare(tr1, tr2) == 0) {

std::cout << "Площади равны" << std::endl;

} else {

std::cout << "Площадь первого треугольника меньше площади второго треугольника" << std::endl;

}

}

}

}

if (decision[0] == 2) {

std::cout << "Работаем с прямоугольником: " << std::endl;

float lx1, ly1;

float rx1, ry1;

std::cout << "Координаты левого верхнего угла: " << std::endl;

std::cin >> lx1;

std::cin >> ly1;

std::cout << "Координаты правого нижнего угла: " << std::endl;

std::cin >> rx1;

std::cin >> ry1;

Rectangle \*rt1 = new Rectangle(lx1, ly1, rx1, ry1); ///< создание фигуры

float deltaX1, deltaY1;

std::cout << "На сколько двигаем по X?" << std::endl;

std::cin >> deltaX1;

std::cout << "На сколько двигаем по Y?" << std::endl;

std::cin >> deltaY1;

rt1->move(deltaX1, deltaY1); ///< свдиг фигуры

rt1->toConsole(); ///< вывод фигуры в консоль

if (decision[1] == 2) {

std::cout << "Работаем со вторым прямоугольником: " << std::endl;

float lx2, ly2;

float rx2, ry2;

std::cout << "Координаты левого верхнего угла: " << std::endl;

std::cin >> lx2;

std::cin >> ly2;

std::cout << "Координаты правого нижнего угла: " << std::endl;

std::cin >> rx2;

std::cin >> ry2;

Rectangle \*rt2 = new Rectangle(lx2, ly2, rx2, ry2); ///< создание фигуры

float deltaX2, deltaY2;

std::cout << "На сколько двигаем по X?" << std::endl;

std::cin >> deltaX2;

std::cout << "На сколько двигаем по Y?" << std::endl;

std::cin >> deltaY2;

rt2->move(deltaX2, deltaY2); ///< свдиг фигуры

rt2->toConsole(); ///< вывод фигуры в консоль

if (compare(rt1, rt2) > 0) { ///< сравнение площадей

std::cout << "Площадь первого прямоугольника больше площади второго прямоугольника" << std::endl;

} else {

if (compare(rt1, rt2) == 0) {

std::cout << "Площади равны" << std::endl;

} else {

std::cout << "Площадь первого прямоугольника меньше площади второго прямоугольника" << std::endl;

}

}

} else {

std::cout << "Работаем с треугольником: " << std::endl;

float xs2[3];

float ys2[3];

for (int i = 0; i < 3; i++) {

std::cout << "Введите координаты " << i + 1 << " точки треугольника (x y)" << std::endl;

std::cin >> xs2[i];

std::cin >> ys2[i];

}

Triangle \*tr2 = new Triangle(xs2[0], ys2[0], xs2[1], ys2[1], xs2[2], ys2[2]); ///< создание фигуры

float deltaX2, deltaY2;

std::cout << "На сколько двигаем по X?" << std::endl;

std::cin >> deltaX2;

std::cout << "На сколько двигаем по Y?" << std::endl;

std::cin >> deltaY2;

tr2->move(deltaX2, deltaY2); ///< сдвиг фигуры

tr2->toConsole(); ///< вывод фигуры в консоль

if (compare(rt1, tr2) > 0) { ///< сравнение площадей

std::cout << "Площадь прямоугльника больше площади треугольника" << std::endl;

} else {

if (compare(rt1, tr2) == 0) {

std::cout << "Площади равны" << std::endl;

} else {

std::cout << "Площадь прямоугльникаа меньше площади треугольника" << std::endl;

}

}

}

}

return 0;

}

Код Task.h

#ifndef Task\_h

#define Task\_h

#include <math.h>

class Figure {

public:

/\*\*

Функция для перемещения объекта по координатной плоскости

@param deltaX значение по оси X

@param deltaY значение по оси Y

\*/

virtual void move(float deltaX, float deltaY) = 0;

/\*\*

Функция, выводящая координаты объекта в консоль

\*/

virtual void toConsole() = 0;

/\*\*

Функция, возвращающая площадь объекта

@return площадь

\*/

virtual float getArea() = 0;

};

class Triangle : public Figure {

public:

float x1, y1;

float x2, y2;

float x3, y3;

/\*\*

Создание объекта "Треугольник" по трем координатам

@param x1 координата x первой точки

@param y1 координата y первой точки

@param x2 координата x второй точки

@param y2 координата y второй точки

@param x3 координата x третьей точки

@param y3 координата y третьй точки

\*/

explicit Triangle(float x1, float y1, float x2, float y2, float x3, float y3) {

this -> x1 = x1;

this -> y1 = y1;

this -> x2 = x2;

this -> y2 = y2;

this -> x3 = x3;

this -> y3 = y3;

}

void move(float deltaX, float deltaY) override {

this -> x1 = x1 + deltaX;

this -> y1 = y1 + deltaY;

this -> x2 = x2 + deltaX;

this -> y2 = y2 + deltaY;

this -> x3 = x3 + deltaX;

this -> y3 = y3 + deltaY;

}

void toConsole() override {

std::cout << "Triangle: " << std::endl;

std::cout << "v1: " << this -> x1 << ", " << this -> y1 << std::endl;

std::cout << "v2: " << this -> x2 << ", " << this -> y2 << std::endl;

std::cout << "v3: " << this -> x3 << ", " << this -> y3 << std::endl;

}

float getArea() override {

return abs( ( (x1 - x3) \* (y2 - y3) - (y1 - y3) \* (x2 - x3) ) / 2);

}

};

class Rectangle : public Figure {

public:

float x1, y1; ///< Левый верхний угол

float x2, y2;

float x3, y3;

float x4, y4; ///< Правый нижний угол

/\*\*

Создание объекта "Прямоугольник" по двум координатам

@param x1 Левый верхний угол Y

@param y1 Левый верхний угол Y

@param x4 Правый нижний угол X

@param y4 Правый нижний угол Y

\*/

explicit Rectangle(float x1, float y1, float x4, float y4) {

this -> x1 = x1;

this -> y1 = y1;

this -> x2 = x1;

this -> y2 = y4;

this -> x3 = x4;

this -> y3 = y1;

this -> x4 = x4;

this -> y4 = y4;

}

void move(float deltaX, float deltaY) override {

this -> x1 = x1 + deltaX;

this -> y1 = y1 + deltaY;

this -> x2 = x2 + deltaX;

this -> y2 = y2 + deltaY;

this -> x3 = x3 + deltaX;

this -> y3 = y3 + deltaY;

this -> x4 = x4 + deltaX;

this -> y4 = y4 + deltaY;

};

void toConsole() override {

std::cout << "Rectangle: " << std::endl;

std::cout << "v1: " << this -> x1 << ", " << this -> y1 << std::endl;

std::cout << "v2: " << this -> x2 << ", " << this -> y2 << std::endl;

std::cout << "v3: " << this -> x3 << ", " << this -> y3 << std::endl;

std::cout << "v4: " << this -> x4 << ", " << this -> y4 << std::endl;

}

float getArea() override {

float height;

float width;

height = abs(this -> y1 - this -> y2);

width = abs(this -> x1 - this -> x3);

return height \* width;

}

};

/\*\*

Функция для сранвения площадей фигур двух классов

@param obj1 первый объект для сравнения

@param obj2 второй объект для сравнения

@return разность площадей фигур

\*/

float compare(Triangle \*obj1, Triangle \*obj2) {

return obj1->getArea() - obj2->getArea();

}

/// перегрузка вышеописанной функции

float compare(Triangle \*obj1, Rectangle \*obj2) {

return obj1->getArea() - obj2->getArea();

}

/// перегрузка вышеописанной функции

float compare(Rectangle \*obj1, Rectangle \*obj2) {

return obj1->getArea() - obj2->getArea();

}

/// перегрузка вышеописанной функции

float compare(Rectangle \*obj1, Triangle \*obj2) {

return obj1->getArea() - obj2->getArea();

}

#endif /\* Task\_h \*/

Вывод программы:

|  |  |
| --- | --- |
| Выберите две фигуры:  1 – Треугольник  2 – Прямоугольник  2  2  Работаем с прямоугольником:  Координаты левого верхнего угла:  0 5  Координаты правого нижнего угла:  5 0  На сколько двигаем по X?  1  На сколько двигаем по Y?  -1  Rectangle:  v1: 1, 4  v2: 1, -1  v3: 6, 4  v4: 6, -1  Работаем со вторым прямоугольником:  Координаты левого верхнего угла:  1 1  Координаты правого нижнего угла:  2 0  На сколько двигаем по X?  0  На сколько двигаем по Y?  0  Rectangle:  v1: 1, 1  v2: 1, 0  v3: 2, 1  v4: 2, 0  Площадь первого прямоугольника больше площади второго прямоугольника  Program ended with exit code: 0 | Выберите две фигуры:  1 – Треугольничек  2 – Прямоугольник  1  2  Работаем с треугольником:  Введите координаты 1 точки треугольника (x y)  3 4  Введите координаты 2 точки треугольника (x y)  0 0  Введите координаты 3 точки треугольника (x y)  2 -2  На сколько двигаем по X?  1  На сколько двигаем по Y?  -2  Triangle:  v1: 4, 2  v2: 1, -2  v3: 3, -4  Работаем с прямоугольником:  Координаты левого верхнего угла:  0 5  Координаты правого нижнего угла:  2 2  На сколько двигаем по X?  -2  На сколько двигаем по Y?  -3  Rectangle:  v1: -2, 2  v2: -2, -1  v3: 0, 2  v4: 0, -1  Площадь треугольника больше площади прямоугольника  Program ended with exit code: 0 |