**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 3**

Тема: Наследование, полиморфизм в С++

Студент: Савров Никита

Группа: 80-207

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2019

1. Постановка задачи

Разработать классы согласно варианту задания, классы должны наследоваться от базового класса Figure. Фигуры являются фигурами вращения. Все классы должны поддерживать набор общих методов:

1. Вычисление геометрического центра фигуры;

2. Вывод в стандартный поток вывода std::cout координат вершин фигуры; 3. Вычисление площади фигуры;

Создать программу, которая позволяет:

• Вводить из стандартного ввода std::cin фигуры, согласно варианту задания.

• Сохранять созданные фигуры в динамический массив std::vector<Figure\*>

• Вызывать для всего массива общие функции (1-3 см. выше).Т.е. распечатывать для каждой фигуры в массиве геометрический центр, координаты вершин и площадь.

• Необходимо уметь вычислять общую площадь фигур в массиве.

• Удалять из массива фигуру по индексу;

Фигуры, согласно варианту задания: Прямоугольник, трапеция, ромб.

1. Описание программы

Для реализации программы напишем класс Figure в котором объявим виртуальные функции которые реализуем в дочерних классах. Далее наследуем от класса Figure классы фигур согласно варианту задания. В этих классах реализуем общие функции, такие как нахождение площади, нахождение геометрического центра фигуры и вывод всех координат фигуры. Так-же реализуем функцию Set для ввода координат и сохранения их в поле класса. В функции main реализуем интерфейс с помощью которого будем выбирать нужное действие.

1. Набор testcases

test\_01

1

0 0

0 4

5 4

5 0

6

1

0 0

0 4

5 4

6 0

6

2

0 0

0 4

4 4

4 0

3

0 0

0 5

4 5

6 0

6

5

4

7

8

2

6

8

0

6

9

test\_02

1

0 0

0 5

6 5

6 0

1

0 0

0 7

3 7

3 0

2

0 0

0 6

6 6

6 0

2

0 0

0 8

8 8

8 0

3

0 0

0 5

4 5

7 0

6

4

5

7

8

2

8

2

6

9

test\_03

1

0 0

0 5

7 5

7 0

2

0 0

0 7

7 7

7 0

6

5

4

7

8

0

8

0

6

9

1. Результаты выполнения тестов.

test\_01

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

1

input cordinats

0 0

0 4

5 4

5 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

6

cordinats figure number - 0

0 0

0 4

5 4

5 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

1

input cordinats

0 0

0 4

5 4

6 0

wrong input

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

6

cordinats figure number - 0

0 0

0 4

5 4

5 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

2

input cordinats

0 0

0 4

4 4

4 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

3

input cordinats

0 0

0 5

4 5

6 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

6

cordinats figure number - 0

0 0

0 4

5 4

5 0

cordinats figure number - 1

0 0

0 4

4 4

4 0

cordinats figure number - 2

0 0

0 5

4 5

6 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

5

geom-center figure number - 0

2.5 2

geom-center figure number - 1

2 2

geom-center figure number - 2

2.46667 2.66667

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

4

square figure number - 0

20

square figure number - 1

16

square figure number - 2

25

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

7

all square = 61

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

8

input index

2

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

6

cordinats figure number - 0

0 0

0 4

5 4

5 0

cordinats figure number - 1

0 0

0 4

4 4

4 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

8

input index

0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

6

cordinats figure number - 0

0 0

0 4

4 4

4 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

9

test\_02

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

1

input cordinats

0 0

0 5

6 5

6 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

1

input cordinats

0 0

0 7

3 7

3 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

2

input cordinats

0 0

0 6

6 6

6 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

2

input cordinats

0 0

0 8

8 8

8 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

3

input cordinats

0 0

0 5

4 5

7 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

6

cordinats figure number - 0

0 0

0 5

6 5

6 0

cordinats figure number - 1

0 0

0 7

3 7

3 0

cordinats figure number - 2

0 0

0 6

6 6

6 0

cordinats figure number - 3

0 0

0 8

8 8

8 0

cordinats figure number - 4

0 0

0 5

4 5

7 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

4

square figure number - 0

30

square figure number - 1

21

square figure number - 2

36

square figure number - 3

64

square figure number - 4

27.5

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

5

geom-center figure number - 0

3 2.5

geom-center figure number - 1

1.5 3.5

geom-center figure number - 2

3 3

geom-center figure number - 3

4 4

geom-center figure number - 4

2.68182 2.72727

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

7

all square = 178.5

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

8

input index

2

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

8

input index

2

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

6

cordinats figure number - 0

0 0

0 5

6 5

6 0

cordinats figure number - 1

0 0

0 7

3 7

3 0

cordinats figure number - 2

0 0

0 5

4 5

7 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

9

test\_03

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

1

input cordinats

0 0

0 5

7 5

7 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

2

input cordinats

0 0

0 7

7 7

7 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

6

cordinats figure number - 0

0 0

0 5

7 5

7 0

cordinats figure number - 1

0 0

0 7

7 7

7 0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

5

geom-center figure number - 0

3.5 2.5

geom-center figure number - 1

3.5 3.5

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

4

square figure number - 0

35

square figure number - 1

49

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

7

all square = 84

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

8

input index

0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

8

input index

0

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

6

1 - enter rectangle

2 - enter rhomb

3 - enter trapeze

4 - square for all figure

5 - geomcenter for all figure

6 - print all figure

7 - all square

8 - erase figure

9 - exit

9

5.Листинг программы

// Лабораторная работа №3 Савров Никита М80-207Б-18

// Разработать классы согласно варианту задания, классы должны наследоваться от базового класса Figure. Фигуры являются фигурами вращения. Все классы должны поддерживать набор общих методов:

// 1. Вычисление геометрического центра фигуры;

// 2. Вывод в стандартный поток вывода std::cout координат вершин фигуры; 3. Вычисление площади фигуры;

// Создать программу, которая позволяет:

// • Вводить из стандартного ввода std::cin фигуры, согласно варианту задания.

// • Сохранять созданные фигуры в динамический массив std::vector<Figure\*>

// • Вызывать для всего массива общие функции (1-3 см. выше).Т.е. распечатывать для каждой фигуры в массиве геометрический центр, координаты вершин и площадь.

// • Необходимо уметь вычислять общую площадь фигур в массиве.

// • Удалять из массива фигуру по индексу;

// Фигуры, согласно варианту задания: Прямоугольник, трапеция, ромб.

#include <iostream>

#include <vector>

#include <cmath>

using namespace std;

class Figure{

public:

virtual double Square () = 0;

virtual void Print\_figure () = 0;

virtual pair<double, double> Geom\_center () = 0;

virtual void Set (vector <pair <double, double>> cord) = 0;

virtual bool IsFigure () = 0;

};

class Rectangle : public Figure {

private:

int count\_point = 4;

vector <pair <double, double>> cordinats;

public:

Rectangle(){

cordinats.resize(count\_point);

for (int i = 0; i < count\_point; i++){

cordinats[i].first = 0;

cordinats[i].second = 0;

}

}

void Set (vector <pair <double, double>> cord){

for (int i = 0; i < count\_point; i++){

this->cordinats[i].first = cord[i].first;

this->cordinats[i].second = cord[i].second;

}

}

double Square (){

double x\_a = abs(cordinats[0].first - cordinats[1].first);

double y\_a = abs(cordinats[0].second - cordinats[1].second);

double size\_a = sqrt(x\_a \* x\_a + y\_a \* y\_a);

double x\_b = abs(cordinats[0].first - cordinats[3].first);

double y\_b = abs(cordinats[0].second - cordinats[3].second);

double size\_b = sqrt(x\_b \* x\_b + y\_b \* y\_b);

return size\_a \* size\_b;

}

void Print\_figure (){

for (int i = 0; i < count\_point; i++){

cout << cordinats[i].first << " " << cordinats[i].second << '\n';

}

return;

}

pair<double, double> Geom\_center (){

return make\_pair((cordinats[2].first - cordinats[0].first) / 2, (cordinats[2].second - cordinats[0].second) / 2);

}

bool IsFigure (){

if (cordinats[0].first != cordinats[1].first || cordinats[2].first != cordinats[3].first)

return false;

if (cordinats[0].second != cordinats[3].second || cordinats[1].second != cordinats[2].second)

return false;

return true;

}

};

class Rhomb : public Figure {

int count\_point = 4;

vector <pair <double, double>> cordinats;

public:

Rhomb(){

cordinats.resize(count\_point);

for (int i = 0; i < count\_point; i++){

cordinats[i].first = 0;

cordinats[i].second = 0;

}

}

void Set (vector <pair <double, double>> cord){

for (int i = 0; i < count\_point; i++){

this->cordinats[i].first = cord[i].first;

this->cordinats[i].second = cord[i].second;

}

}

public:

double Square (){

double x\_d1 = abs(cordinats[0].first - cordinats[2].first);

double y\_d1 = abs(cordinats[0].second - cordinats[2].second);

double size\_d1 = sqrt(x\_d1 \* x\_d1 + y\_d1 \* y\_d1);

double x\_d2 = abs(cordinats[1].first - cordinats[3].first);

double y\_d2 = abs(cordinats[1].second - cordinats[3].second);

double size\_d2 = sqrt(x\_d2 \* x\_d2 + y\_d2 \* y\_d2);

return size\_d1 \* size\_d2 / 2;

}

void Print\_figure (){

for (int i = 0; i < count\_point; i++){

cout << cordinats[i].first << " " << cordinats[i].second << '\n';

}

return;

}

pair<double, double> Geom\_center (){

return make\_pair((cordinats[2].first - cordinats[0].first) / 2, (cordinats[2].second - cordinats[0].second) / 2);

}

bool IsFigure (){

double x\_a = abs(cordinats[0].first - cordinats[1].first);

double y\_a = abs(cordinats[0].second - cordinats[1].second);

double size\_a = sqrt(x\_a \* x\_a + y\_a \* y\_a);

double x\_b = abs(cordinats[0].first - cordinats[3].first);

double y\_b = abs(cordinats[0].second - cordinats[3].second);

double size\_b = sqrt(x\_b \* x\_b + y\_b \* y\_b);

double x\_c = abs(cordinats[1].first - cordinats[2].first);

double y\_c = abs(cordinats[1].second - cordinats[2].second);

double size\_c = sqrt(x\_c \* x\_c + y\_c \* y\_c);

double x\_d = abs(cordinats[2].first - cordinats[3].first);

double y\_d = abs(cordinats[2].second - cordinats[3].second);

double size\_d = sqrt(x\_d \* x\_d + y\_d \* y\_d);

if (size\_a != size\_b || size\_a != size\_c || size\_a != size\_d)

return false;

return true;

}

};

class Trapeze : public Figure {

int count\_point = 4;

vector <pair <double, double>> cordinats;

public:

Trapeze(){

cordinats.resize(count\_point);

for (int i = 0; i < count\_point; i++){

cordinats[i].first = 0;

cordinats[i].second = 0;

}

}

void Set(vector <pair <double, double>> cord){

for (int i = 0; i < count\_point; i++){

this->cordinats[i].first = cord[i].first;

this->cordinats[i].second = cord[i].second;

}

}

double Square(){

double h = cordinats[1].second - cordinats[0].second;

double a = cordinats[2].first - cordinats[1].first;

double b = cordinats[3].first - cordinats[0].first;

return ((a + b) / 2) \* h;

}

void Print\_figure (){

for (int i = 0; i < count\_point; i++){

cout << cordinats[i].first << " " << cordinats[i].second << '\n';

}

}

pair <double, double> Geom\_center (){

double h = cordinats[1].second - cordinats[0].second;

double a = cordinats[2].first - cordinats[1].first;

double b = cordinats[3].first - cordinats[0].first;

double mid\_a\_x = (cordinats[2].first + cordinats[1].first) / 2;

double mid\_a\_y = cordinats[1].second;

double mid\_b\_x = (cordinats[3].first + cordinats[0].first) / 2;

double mid\_b\_y = cordinats[0].second;

double k = (mid\_b\_y - mid\_a\_y) / (mid\_b\_x - mid\_a\_x);

double c = mid\_a\_y - k \* mid\_a\_x;

double y = (h / 3) \* (2 \* b + a) / (a + b);

double x = (y - c) / k;

return make\_pair(x,y);

}

bool IsFigure (){

if (cordinats[0].second == cordinats[3].second && cordinats[1].second == cordinats[2].second)

return true;

if (cordinats[0].first == cordinats[1].first && cordinats[3].first == cordinats[2].first)

return true;

return false;

}

};

int main(){

vector <Figure\*> f(10);

int f\_size = 0;

pair <double, double> p;

vector <pair <double, double>> v1(4);

double sum;

char menu = '0';

while (menu != '9'){

cout << "1 - enter rectangle\n2 - enter rhomb\n3 - enter trapeze\n4 - square for all figure\n5 - geomcenter for all figure\n6 - print all figure\n7 - all square\n8 - erase figure\n9 - exit\n";

cin >> menu;

switch (menu)

{

case '1':

if (f.size() <= f\_size)

f.resize(2 \* f\_size);

cout << "input cordinats\n";

for (int i = 0; i < 4; i++){

cin >> v1[i].first >> v1[i].second;

}

f[f\_size] = new Rectangle();

f[f\_size]->Set(v1);

if (f[f\_size]->IsFigure()){

f\_size++;

}else{

cout << "wrong input\n";

delete(f[f\_size]);

f.pop\_back();

}

break;

case '2':

cout << "input cordinats\n";

if (f.size() <= f\_size){

f.resize(2 \* f\_size);

}

for (int i = 0; i < 4; i++){

cin >> v1[i].first >> v1[i].second;

}

f[f\_size] = new Rhomb();

f[f\_size]->Set(v1);

if (f[f\_size]->IsFigure()){

f\_size++;

}else{

cout << "wrong input\n";

delete(f[f\_size]);

f.pop\_back();

}

break;

case '3':

cout << "input cordinats\n";

if (f.size() <= f\_size){

f.resize(2 \* f\_size);

}

for (int i = 0; i < 4; i++){

cin >> v1[i].first >> v1[i].second;

}

f[f\_size] = new Trapeze();

f[f\_size]->Set(v1);

if (f[f\_size]->IsFigure()){

f\_size++;

}else{

cout << "wrong input\n";

delete(f[f\_size]);

f.pop\_back();

}

break;

case '4':

for (int i = 0; i < f\_size; i++){

cout << "square figure number - " << i << '\n' << f[i]->Square() << '\n';

}

break;

case '5':

for (int i = 0; i < f\_size; i++){

p = f[i]->Geom\_center();

cout << "geom-center figure number - " << i << '\n' << p.first << " " << p.second << '\n';

}

break;

case '6':

for (int i = 0; i < f\_size; i++){

cout << "cordinats figure number - " << i << '\n';

f[i]->Print\_figure();

}

break;

case '7':

sum = 0;

for (int i = 0; i < f\_size; i++){

sum += f[i]->Square();

}

cout << "all square = " << sum << '\n';

break;

case '8':

cout << "input index\n";

int j;

cin >> j;

if (j >= f\_size){

cout << "not exist\n";

break;

}

for (int i = j; i < f\_size - 1; i++){

swap(f[i], f[i + 1]);

}

delete(f[f\_size]);

f.pop\_back();

f\_size--;

break;

default:

break;

}

}

}

https://github.com/trol53/oop\_exercise\_01/tree/master/oop\_exercise\_03

6.Вывод

Данная программа может быть полезна для работы с геометрическими фигурами. Так-же ее можно улучшить добавив дополнительные функции вроде нахождения периметра фигуры и прочие. Благодаря парадигме ООП мы можем представлять пользовательские типы данных в виде объектов что значительно упрощает программирование. Так-же эта работа показывает практическое применение наследования классов.

Литература

1.Информационный портал хабр[электронный ресурс] URL:https://habr.com/ru/post/445948/

(дата обращения 13.10.2019)

2.Информационный портал хабр [электронный ресурс] URL: https://ru.cppreference.com/w/cpp/container/vector

(дата обращения 14.10.2019)