**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

Студент: Блистунова Валерия

Группа: 80-201

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2019

1. Постановка задачи

Вариант №12: Трапеция, Ромб, 5-угольник.

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;

1. Ссылка на github

<https://github.com/val-lera/oop_exercise_03>

1. Описание программы

Программа предназначена для работы с геометрическими фигурами. Печатается меню, где пользователь может выбрать нужное действие. Сначала необходимо добавить фигуру. На вход программа должна получить координаты вершин фигуры (трапеции, ромба или пятиугольника), затем происходит проверка, является ли заданная фигура действительно равнобедренной трапецией, ромбом. Далее можно вывести все фигуры, которые сохранялись в массив, их центр и площадь, также можно вывести общую площадь фигур в массиве и удалить фигуру по индексу. Для выхода из программы необходимо ввести 0.

1. Набор testcases

Тест 1.

|  |  |  |
| --- | --- | --- |
| Пункт меню | Действие | Предполагаемый результат |
| 2 | Добавить фигуру | Откроется дополнительное меню, в котором необходимо выбрать нужную фигуру |
| 1 | Ввести координаты трапеции  1 2 3 5 6 5 8 7 | Вывод сообщения, что трапеция не является равнобедренной, выход в основное меню |
| 2 | Добавить фигуру | Откроется дополнительное меню, в котором необходимо выбрать нужную фигуру |
| 1 | Ввести координаты трапеции  1 2 3 5 6 5 8 2 | В массив фигур добавится заданная трапеция |
| 2 | Добавить фигуру | Откроется дополнительное меню, в котором необходимо выбрать нужную фигуру |
| 2 | Ввести координаты ромба  -2 0 0 4 -4 2 -6 -2 | В массив фигур добавится заданный ромб |
| 2 | Добавить фигуру | Откроется дополнительное меню, в котором необходимо выбрать нужную фигуру |
| 3 | Ввести радиус описанной окружности и координаты центра пятиугольника  2 5 5 | В массив фигур добавится заданный пятиугольник |
| 2 | Добавить фигуру | Откроется дополнительное меню, в котором необходимо выбрать нужную фигуру |
| 1 | Ввести координаты трапеции  0 0 2 6 4 6 6 0 | В массив фигур добавится заданная трапеция |
| 3 | Вывести на экран информацию о всех фигурах в массиве | Вывод информации о всех фигурах (название фигуры, координаты всех вершин, координаты центра, площадь) |
| 4 | Вывести общую площадь всех фигур в массиве на экран | Вывод общей площади фигур в массиве 58.6085 |
| 5 | Удалить фигуру по индексу 4 | Из массива удалится последняя трапеция |
| 3 | Вывести на экран информацию о всех фигурах в массиве | Вывод информации о всех фигурах (название фигуры, координаты всех вершин, координаты центра, площадь) |
| 4 | Вывести общую площадь всех фигур в массиве на экран | Вывод общей площади фигур в массиве 34.6085 |
| 33 | - | Вывод сообщения о том, что такого пункта меню нет |
| 1 | Распечатать меню | Печать меню на экран |
| 0 | Выйти из программы | Завершение программы |

1. Результаты выполнения тестов

Фактические результаты выполнения тестов совпали с предполагаемыми.

1. Листинг программы

/\*Лабораторная работа №3

Выполнила: Блистунова Валерия

Группа М8О-201Б

Вариант №12: Трапеция, Ромб, 5-угольник

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\*/

#include<iostream>

#include<vector>

#include<cmath>

using namespace std;

#define pi 3.14159265358979323846

class Figure

{

public:

Figure(){}

virtual ~Figure(){}

double x, y, ar, all;

virtual void Center()=0;

virtual void Coord()=0;

virtual double Area()=0;

vector<double> X;

vector<double> Y;

void pCenter(){

cout << "Координаты центра: " << "(" << x << "; " << y << ")" << endl;

}

void pArea(){

cout << "Площадь: " << Area() << endl;

}

double length(double a, double b, double c, double d){

double l;

l= sqrt(pow((c-a),2)+pow((d-b),2));

return l;

}

};

class Trap : public Figure

{

public:

Trap(){};

Trap(double x1, double y1, double x2, double y2, double x3, double y3, double x4, double y4){

X.push\_back(x1); X.push\_back(x2); X.push\_back(x3); X.push\_back(x4);

Y.push\_back(y1); Y.push\_back(y2); Y.push\_back(y3); Y.push\_back(y4);

}

~Trap(){};

double Area() override{

double h;

h=length(X[1],Y[0],X[1],Y[1]);

ar = ((length(X[0],Y[0],X[3],Y[3])+length(X[2],Y[2],X[1],Y[1]))/2)\*h;

}

void Center()override{

x=0; y=0;

for(int i=0; i < X.size(); i++){

x+=X[i];

y+=Y[i];

}

x = x / X.size();

y = y / Y.size();

}

void Coord(){

cout << "Координаты вершин трапеции: ";

for(int i = 0; i < X.size(); i++)

cout << "(" << X[i] << "; " << Y[i] << ") ";

cout << endl;

}

};

class Romb : public Figure

{

public:

Romb(){};

Romb(double x1, double y1, double x2, double y2, double x3, double y3, double x4, double y4){

X.push\_back(x1); X.push\_back(x2); X.push\_back(x3); X.push\_back(x4);

Y.push\_back(y1); Y.push\_back(y2); Y.push\_back(y3); Y.push\_back(y4);

}

~Romb(){};

double Area() override{

ar = (length(X[0],Y[0],X[2],Y[2])\*length(X[1],Y[1],X[3],Y[3]))/2;

}

void Center()override{

x=0; y=0;

for(int i=0; i < X.size(); i++){

x+=X[i];

y+=Y[i];

}

x = x / X.size();

y = y / Y.size();

}

void Coord(){

cout << "Координаты вершин ромба: ";

for(int i = 0; i < X.size(); i++)

cout << "(" << X[i] << "; " << Y[i] << ") ";

cout << endl;

}

};

class Pent : public Figure

{

public:

Pent(){};

Pent(double r, double xc, double yc){

for(int i=0; i<5; i++){

X.push\_back(xc+r\*cos(2\*pi\*i/5));

Y.push\_back(yc+r\*sin(2\*pi\*i/5));

}

}

~Pent(){};

double Area()override{

double a=length(X[0],Y[0],X[1],Y[1]);

ar= (5/2)\*a\*sqrt((pow((a/(2\*sin(pi/5))),2))-(pow(a,2)/4));

}

void Center()override{

x=0; y=0;

for(int i=0; i < X.size(); i++){

x+=X[i];

y+=Y[i];

}

x = x / X.size();

y = y / Y.size();

}

void Coord(){

cout << "Координаты вершин пятиугольника: ";

for(int i = 0; i < X.size(); i++)

cout << "(" << X[i] << "; " << Y[i] << ") ";

cout << endl;

}

};

void Menu(){

cout << "Введите число от 1 до 5 или 0 для действий: " << endl <<

"1. Распечатать меню" << endl <<

"2. Добавить фигуру" << endl <<

"3. Общие функции для всего массива" << endl <<

"4. Вычислить общую площадь фигур в массиве" << endl <<

"5. Удалить фигуру из массива по индексу" << endl <<

"0. Выход из программы" << endl;

}

double l(double a, double b, double c, double d){

double l;

l= sqrt(pow((c-a),2)+pow((d-b),2));

return l;

}

int main(){

vector<Figure\*> f;

int user\_command =1;

double x1, y1, x2, y2, x3, y3, x4, y4, xc, yc, r;

for(;;){

switch (user\_command){

case 1:

Menu();

break;

case 2:

int k;

cout << "Введите номер фигуры, которую нужно добавить "<< endl <<

"1. Трапеция" << endl <<

"2. Ромб"<< endl <<

"3. 5-угольник"<< endl <<

"0. Выход в основное меню"<< endl;

cin >> k;

switch (k){

case 1:

{cout << "Введите координаты вершин трапеции ";

cin >> x1 >> y1 >> x2 >> y2 >> x3 >> y3 >> x4 >> y4;

if(l(x1,y1,x2,y2)!=l(x3,y3,x4,y4)){

cout << "Трапеция не равнобедренная" << endl;

break;

}

else{

Figure \*t;

t= new Trap(x1,y1,x2,y2,x3,y3,x4,y4);

f.push\_back(t);}

break;

}

case 2:

cout << "Введите координаты вершин ромба ";

cin >> x1 >> y1 >> x2 >> y2 >> x3 >> y3 >> x4 >> y4;

if(l(x1,y1,x2,y2)!=l(x2,y2,x3,y3) || l(x3,y3,x2,y2)!=l(x3,y3,x4,y4)){

cout << "Введены неверные координаты ромба" << endl;

break;

}

else {

Figure \*t;

t= new Romb(x1,y1,x2,y2,x3,y3,x4,y4);

f.push\_back(t);

}

break;

case 3:

cout << "Введите радиус окружности, в которую вписан пятиугольник, координаты центра пятиугольника ";

cin >> r >> xc >> yc;

Figure \*t;

t=new Pent(r,xc,yc);

f.push\_back(t);

break;

case 0:

Menu();

break;

default:

cerr << "Нет такого пункта" << endl;

break;

}

break;

case 3:

if (f.size()==0) {

cout << "Массив фигур пуст" << endl;

break;

}

else{

for(int i = 0; i < f.size(); i++)

(\*f[i]).Area();

for(int i = 0; i < f.size(); i++)

(\*f[i]).Center();

for(int i = 0; i < f.size(); i++){

(\*f[i]).Coord();

(\*f[i]).pCenter();

(\*f[i]).pArea();

}

break;

}

case 4:

double j;

j=0;

for(int i = 0; i < f.size(); i++)

j+=(\*f[i]).Area();

cout << "Общая площадь фигур в массиве: " << j << endl;

break;

case 5:

if (f.size()==0){

cout << "Массив пуст" << endl;

break;

}

else{

cout << "Введите индекс: ";

cin >> k;

if(k <= 0 || k > f.size()){

cout << "Неверный индекс" << endl;

break;

}

else{

f.erase(f.begin()+(k-1));

break;

}

break;

}

case 0:

{for(size\_t i = 0; i < f.size(); i++){

delete f[i];

}

exit(0);}

default:

cerr << "Нет такого пункта" << endl;

break;

}

cout << endl << "Введите пункт меню: ";

cin >> user\_command;

}

return 0;

}

1. Вывод

Создана программа на языке С++, позволяющая работать с геометрическими фигурами: находить координаты центра фигур, считать площадь как отдельной фигуры, так и всех фигур в массиве, удалять фигуру из массива по индексу, выводить на экран нужную информацию.

1. Список литературы

1. Наследование в С++ <https://habr.com/ru/post/445948/>

2.Наследование классов в С++ [https://code-live.ru/post/cpp-class-inheritance](https://code-live.ru/post/cpp-class-inheritance/)