**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: Хренов Геннадий

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:

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

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

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

3) Необходимо уметь вычислять общую площадь фигур в массиве.

4) Удалять из массива фигуру по индексу;

1. Описание программы

Программа состоит из базового класса Figure, где определены виртуальные методы для всех типов фигур. Далее создаются классы самих фигур, состоящие из координат и переопределённых методов базового класса. В функции main создаётся вектор указателей, куда мы можем записывать фигуры и удалять их по индексу. Методы классов вычисляют площадь фигуры, геометрический центр, выводят координаты фигур. Также по условию задания была создана функция, вычисляющая сумму всех площадей записанных фигур.

1. Набор testcases

1)push s 2 2 4 4 push r 1 1 7 4 push t 5 1 6 2 7 2 8 1

2)push t 1 1 4 4 7 4 10 1 push t 3 4 5 3 5 2 3 1

3)push s 1 1 1 1 push s 2 2 2 2 push s 3 3 3 3 push s 4 4 4 4

Для тестов 1-2 проверить вывод методов, для 3 проверить поиндексное удаление.

1. Результаты выполнения тестов

1)vector Figure\* was created. Press any bottom for istruction

push s 2 2 4 4 push r 1 1 7 4 push t 5 1 6 2 7 2 8 1

print

figure № 1

center : 3.000000, 3.000000

square : 4

A = (2.000000,2.000000) B = (2.000000,4.000000) C = (4.000000,4.000000) D = (4.000000,2.000000)

figure № 2

center : 4.000000, 2.500000

square : 18

A = (1.000000,1.000000) B = (1.000000,4.000000) C = (7.000000,4.000000) D = (7.000000,1.000000)

figure № 3

center : 6.500000, 1.500000

square : 2

A = (5.000000,1.000000) B = (6.000000,2.000000) C = (7.000000,2.000000) D = (8.000000,1.000000)

2)vector Figure\* was created. Press any bottom for istruction

push t 1 1 4 4 7 4 10 1 push t 3 4 5 3 5 2 3 1

print

figure № 1

center : 5.500000, 2.500000

square : 18

A = (1.000000,1.000000) B = (4.000000,4.000000) C = (7.000000,4.000000) D = (10.000000,1.000000)

figure № 2

center : 4.000000, 2.500000

square : 4

A = (3.000000,4.000000) B = (5.000000,3.000000) C = (5.000000,2.000000) D = (3.000000,1.000000)

squaresum

squaresum : 22

3)vector Figure\* was created. Press any bottom for istruction

del 0

invalid index

push s 1 1 1 1 push s 2 2 2 2 push s 3 3 3 3 push s 4 4 4 4

del 2

del 1

print

figure № 1

center : 1.000000, 1.000000

square : 0

A = (1.000000,1.000000) B = (1.000000,1.000000) C = (1.000000,1.000000) D = (1.000000,1.000000)

figure № 2

center : 4.000000, 4.000000

square : 0

A = (4.000000,4.000000) B = (4.000000,4.000000) C = (4.000000,4.000000) D = (4.000000,4.000000)

del 1

print

figure № 1

center : 1.000000, 1.000000

square : 0

A = (1.000000,1.000000) B = (1.000000,1.000000) C = (1.000000,1.000000) D = (1.000000,1.000000)

1. Листинг программы

//Хренов Геннадий М8О-207Б

// Созданны классы : квадрат, прямоугольник, трапеция

//и методы для подсчёта геометрического центра, площади и печать координат.

#include<iostream>

#include <math.h>

#include <vector>

struct point {

double x;

double y;

};

double dist(point a, point b){

return sqrt(pow(b.x - a.x,2) + pow(b.y - a.y,2));

}

class Figure {

public:

virtual void GCF() = 0;

virtual double SF() = 0;

virtual void print() = 0;

};

class Square : public Figure {

public:

Square() {}

Square(const point& z, const point& w) : Square(){

a.x = z.x;

c.x = w.x;

a.y = z.y;

c.y = w.y;

}

void GCF(){

printf("center : %lf, %lf\n", (a.x+c.x)/2, (a.y+c.y)/2);

}

double SF(){

point b;

b.x = a.x;

b.y = c.y;

return pow(dist(a,b), 2);

}

void print(){

printf("A = (%lf,%lf) B = (%lf,%lf) C = (%lf,%lf) D = (%lf,%lf)\n", a.x, a.y, a.x, c.y, c.x, c.y, c.x, a.y);

}

private:

point a;

point c;

};

class Rectan : public Figure {

public:

Rectan() {}

Rectan(const point& z, const point& w) : Rectan() {

a.x = z.x;

c.x = w.x;

a.y = z.y;

c.y = w.y;

}

void GCF(){

printf("center : %lf, %lf\n", (a.x+c.x)/2, (a.y+c.y)/2);;

}

double SF(){

point b;

b.x = a.x;

b.y = c.y;

return dist(a,b) \* dist(b,c);

}

void print(){

printf("A = (%lf,%lf) B = (%lf,%lf) C = (%lf,%lf) D = (%lf,%lf)\n", a.x, a.y, a.x, c.y, c.x, c.y, c.x, a.y);

}

private:

point a;

point c;

};

class Trape : public Figure {

public:

Trape() {}

Trape(const point& z, const point& e,const point& w, const point& q) : Trape() {

a.x = z.x;

c.x = w.x;

a.y = z.y;

c.y = w.y;

b.x = e.x;

b.y = e.y;

d.x = q.x;

d.y = q.y;

}

void GCF(){

printf("center : %lf, %lf\n", (a.x + c.x + b.x + d.x) / 4, (a.y + c.y + b.y + d.y) / 4);

}

double SF(){

double h = sqrt(pow(dist(a,b),2) - pow((dist(a,d) - dist(c,b))/2,2));

return (dist(a,d) + dist(b,c)) \* 0.5 \* h;

}

void print(){

printf("A = (%lf,%lf) B = (%lf,%lf) C = (%lf,%lf) D = (%lf,%lf)\n", a.x, a.y, b.x, b.y, c.x, c.y, d.x, d.y);

}

private:

point a;

point b;

point c;

point d;

};

void read(std::vector<Figure\*>& fig)

{

char shape;

Square \*s;

Rectan \*r = nullptr;

Trape \*t;

point a,b,c,d;

std::cin >> shape;

switch (shape) {

case 't':

std::cin >> a.x >> a.y >> b.x >> b.y >> c.x >> c.y >> d.x >> d.y;

t = new Trape(a,b,c,d);

fig.push\_back(t);

break;

case 's':

std::cin >> a.x >> a.y >> c.x >> c.y;

s = new Square(a,c);

fig.push\_back(s);

break;

case 'r':

std::cin >> a.x >> a.y >> c.x >> c.y;

r = new Rectan(a,c);

fig.push\_back(dynamic\_cast<Figure\*>(r));

break;

default:

std::cout << "doing something wrong\n";

return;

}

}

double allsq(std::vector<Figure\*>& fig) {

double sum = 0;

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

sum += fig[i]->SF();

}

return sum;

}

void del(std::vector<Figure\*>& fig, int n){

if (n < 0 || n >= fig.size()) {

std::cout << "invalid index\n";

return ;

} else{

delete fig[n];

for (int i = n; i < fig.size() - 1; ++i){

fig[i] = fig[i+1];

}

fig.pop\_back();

}

}

int main() {

std:: string str;

std::vector<Figure\*> figu;

std::cout << "vector Figure\* was created. Press any bottom for istruction\n";

while(std::cin >> str){

if (str == "push") {

read(figu);

}else if (str == "print") {

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

std::cout << "figure № " << i + 1 << std::endl;

figu[i]->GCF();

std::cout << "square : " << figu[i]->SF() << std::endl;

figu[i]->print();

std::cout << std::endl;

}

}else if (str == "squaresum") {

std::cout << "squaresum : " <<allsq(figu) << std::endl;

}else if (str == "del") {

int i;

std::cin >> i;

del(figu,i);

}else if ( str == "q"){

figu.clear();

return 0;

}else{

std::cout << "USE:" << std::endl

<< "'push (s,r,t) + coord' to add" << std::endl

<< "'print' to see all answers" << std::endl

<< "'squaresum' to see sum of all square"<< std:: endl

<< "'del' to delete" << std::endl

<< "'q' to quit" << std::endl;

}

}

}

6. Выводы: Наследование позволяет создавать производные классы (классы наследники), взяв за основу все методы и элементы базового класса (класса родителя). Таким образом экономится масса времени на написание и отладку кода новой программы. Объекты производного класса свободно могут использовать всё, что создано и отлажено в базовом классе. При этом, мы можем в производный класс дописать необходимый код для усовершенствования программы: добавить новые элементы, методы.

СПИСОК ЛИТЕРАТУРЫ

1. Наследование классов [Электронный ресурс]. URL:<http://cppstudio.com/post/10103/>

(дата обращения: 12.10.2019).

2.Наследование в С++ [Электронный ресурс]. URL: <https://habr.com/ru/post/445948/>

(дата обращения: 12.10.2019).