Московский Авиационный Институт

(Национальный исследовательский Университет)

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

**Лабораторная работа**

**по курсу «ООП»**

**Тема:**

**Проектирование структуры классов.**

|  |  |
| --- | --- |
| Студент: | Суворова С. А. |
| Группа: | М80-206Б-18 |
| Преподаватель: | Журавлев А.А. |
| Вариант: | 22 |
| Оценка: |  |
| Дата: |  |

Москва

2019

1.**Код на C++:**

point.h:

#ifndef D\_POINT\_H\_

#define D\_POINT\_H\_

#include <iostream>

struct point{

double x,y;

};

std::istream& operator>>(std::istream& is, point& p);

std::ostream& operator<<(std::ostream& os,const point& p);

point operator+(point x1,point x2);

point& operator/= (point& x1, int number);

#endif

point.cpp:

#include <iostream>

#include "point.h"

std::istream& operator>> (std::istream& is, point& p){

is >> p.x >>p.y;

return is;

}

std::ostream& operator<< (std::ostream& os, const point& p){

os << p.x << " " << p.y;

return os;

}

point operator+(point x1,point x2){

point x3;

x3.x=x1.x+x2.x;

x3.y=x1.y+x2.y;

return x3;

}

point& operator/= (point& x1, int number){

x1.x=x1.x/number;

x1.y=x1.y/number;

return x1;

}

figure.h:

#ifndef D\_FIGURE\_H\_

#define D\_FIGURE\_H\_

#include <iostream>

#include "point.h"

struct figure{

virtual point center() const = 0;

virtual void print(std::ostream &os) const = 0;

virtual void help\_print(std::ostream &os) const = 0;

virtual double square() const = 0;

virtual ~figure() {};

};

#endif

five\_angles.h:

#ifndef D\_FIVE\_ANGLES\_H\_

#define D\_FIVE\_ANGLES\_H\_

#include <iostream>

#include "figure.h"

struct five\_angles : figure{

five\_angles(std::istream &is);

point center() const override;

void print(std::ostream &os) const override;

void help\_print(std::ostream &os) const override;

double square() const override;

private:

point one,two,three,four,five;

};

#endif

five\_angles.cpp:

#include <iostream>

#include "five\_angles.h"

five\_angles::five\_angles(std::istream &is){

is >> one >> two >> three >> four >> five;

}

point five\_angles::center() const {

point p;

p=one+two+three+four+five;

p/=5;

return p;

}

void five\_angles::print(std::ostream &os) const {

os << one << " " << two << " " << three << " " << four << " " << five <<"\n";

}

void five\_angles::help\_print(std::ostream &os) const {

os << "1 " << one << " " << two << " " << three << " " << four << " " << five <<"\n";

}

double five\_angles::square() const {

double s=0;

s=(one.x\*two.y+two.x\*three.y+three.x\*four.y+four.x\*five.y+five.x\*one.y-two.x\*one.y-

three.x\*two.y-four.x\*three.y-five.x\*four.y-one.x\*five.y)/2;

if(s<0){

return -s;

}else {

return s;

}

}

six\_angles.h:

#ifndef D\_SIX\_ANGLES\_H\_

#define D\_SIX\_ANGLES\_H\_

#include <iostream>

#include "figure.h"

struct six\_angles : figure{

six\_angles(std::istream &is);

point center() const override;

void print(std::ostream &os) const override;

void help\_print(std::ostream &os) const override;

double square() const override;

private:

point one,two,three,four,five,six;

};

#endif

six\_angles.cpp:

#include <iostream>

#include "six\_angles.h"

six\_angles::six\_angles(std::istream &is){

is >> one >> two >> three >> four >> five >>six;

}

point six\_angles::center() const {

point p;

p=one+two+three+four+five+six;

p/=6;

return p;

}

void six\_angles::print(std::ostream &os) const {

os << one << " " << two << " " << three << " " << four << " " << five << " " << six <<"\n";

}

void six\_angles::help\_print(std::ostream &os) const {

os <<"2 " << one << " " << two << " " << three << " " << four << " " << five << " " << six <<"\n";

}

double six\_angles::square() const {

double s=0;

s=(one.x\*two.y+two.x\*three.y+three.x\*four.y+four.x\*five.y+five.x\*six.y+six.x\*one.y-two.x\*one.y-

three.x\*two.y-four.x\*three.y-five.x\*four.y-six.x\*five.y-one.x\*six.y)/2;

if(s<0){

return -s;

}else {

return s;

}

}

eight\_angles.h:

#ifndef D\_EIGHT\_ANGLES\_H\_

#define D\_EIGHT\_ANGLES\_H\_

#include <iostream>

#include "figure.h"

struct eight\_angles : figure{

eight\_angles(std::istream &is);

point center() const override;

void print(std::ostream &os) const override;

void help\_print(std::ostream &os) const override;

double square() const override;

private:

point one,two,three,four,five,six,seven,eight;

};

#endif

eight\_angles.cpp:

#include <iostream>

#include "eight\_angles.h"

eight\_angles::eight\_angles(std::istream &is){

is >> one >> two >> three >> four >> five >>six >>seven >>eight;

}

point eight\_angles::center() const {

point p;

p=one+two+three+four+five+six+seven+eight;

p/=8;

return p;

}

void eight\_angles::print(std::ostream &os) const {

os << one << " " << two << " " << three << " " << four << " " << five << " " << six << " " << seven

<< " " << eight<<"\n";

}

void eight\_angles::help\_print(std::ostream &os) const {

os <<"3 " << one << " " << two << " " << three << " " << four << " " << five << " " << six << " " << seven

<< " " << eight<<"\n";

}

double eight\_angles::square() const {

double s=0;

s=(one.x\*two.y+two.x\*three.y+three.x\*four.y+four.x\*five.y+five.x\*six.y+six.x\*seven.y+seven.x\*eight.y+

eight.x\*one.y-two.x\*one.y-three.x\*two.y-four.x\*three.y-five.x\*four.y-six.x\*five.y-seven.x\*six.y

-eight.x\*seven.y-one.x\*eight.y)/2;

if(s<0){

return -s;

}else {

return s;

}

}

document.h:

#ifndef D\_DOCUMENT\_H\_

#define D\_DOCUMENT\_H\_

#include "figure.h"

#include "five\_angles.h"

#include "six\_angles.h"

#include "eight\_angles.h"

#include <vector>

#include <memory>

#include <iostream>

struct document{

document()= default;

void save(std::ostream& os) const;

void load(std::istream& is);

void add\_figure(std::unique\_ptr<figure>&& ptr, size\_t id);

void remove\_figure(size\_t id);

void show(std::ostream &os) const;

void undo();

struct command{

size\_t id;

std::unique\_ptr<figure> ptr\_;

virtual void undo(document &doc)=0;

};

struct add\_command:public command{

void undo(document &doc) override;

};

struct remove\_command:public command{

void undo(document &doc) override;

};

private:

std::vector<std::unique\_ptr<figure>> figures\_;

std::vector<std::unique\_ptr<command>> operations\_;

};

#endif

document.cpp:

#include <iostream>

#include "document.h"

void document::save(std::ostream& os) const {

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

figures\_[i]->help\_print(os);

}

}

void document::load(std::istream& is){

int help;

while(is >> help){

if(help ==1){

five\_angles fig(is);

std::unique\_ptr<figure> new\_figure;

new\_figure=std::make\_unique<five\_angles>(fig);

figures\_.push\_back(std::move(new\_figure));

}else if(help ==2){

six\_angles fig(is);

std::unique\_ptr<figure> new\_figure;

new\_figure=std::make\_unique<six\_angles>(fig);

figures\_.push\_back(std::move(new\_figure));

}else if(help ==3){

eight\_angles fig(is);

std::unique\_ptr<figure> new\_figure;

new\_figure=std::make\_unique<eight\_angles>(fig);

figures\_.push\_back(std::move(new\_figure));

}

}

}

void document::add\_figure(std::unique\_ptr<figure>&& ptr,size\_t id){

figures\_.insert(figures\_.begin() + id,std::move(ptr));

add\_command op1;

std::unique\_ptr<add\_command> op;

op=std::make\_unique<add\_command>(std::move(op1));

op->id=id;

op->ptr\_=nullptr;

operations\_.push\_back(std::move(op));

}

void document::remove\_figure(size\_t id){

remove\_command op1;

std::unique\_ptr<remove\_command> op;

op=std::make\_unique<remove\_command>(std::move(op1));

op->id=id;

op->ptr\_=std::move(figures\_[id]);

operations\_.push\_back(std::move(op));

figures\_.erase(figures\_.begin() + id);

}

void document::show(std::ostream &os) const {

if(figures\_.size()>0) {

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

os << "figure " << i << "\n";

os << "cordinats ";

figures\_[i]->print(os);

os << "center " << figures\_[i]->center() << "\n";

os << "square " << figures\_[i]->square() << "\n";

}

}

}

void document::undo() {

if(operations\_.size()>0) {

operations\_[operations\_.size()-1]->undo(\*this);

operations\_.erase(operations\_.begin()+operations\_.size()-1);

}

}

void document::add\_command::undo(document &doc) {

doc.figures\_.erase(doc.figures\_.begin() + id);

}

void document::remove\_command::undo(document &doc) {

doc.figures\_.insert(doc.figures\_.begin() + id,std::move(ptr\_));

}

factory.h:

#ifndef D\_FACTORY\_H\_

#define D\_FACTORY\_H\_

#include "document.h"

#include "figure.h"

#include "five\_angles.h"

#include "six\_angles.h"

#include "eight\_angles.h"

#include <vector>

#include <memory>

#include <iostream>

struct factory{

void construct(std::unique\_ptr<document>& vec);

};

#endif

factory.cpp:

#include <iostream>

#include "factory.h"

void factory::construct(std::unique\_ptr<document> &doc1) {

std::string figures;

std::cin >>figures;

size\_t id;

std::cin >> id;

if(figures == "five\_angles"){

std::unique\_ptr<figure> new\_figure;

new\_figure=std::make\_unique<five\_angles>( five\_angles(std::cin));

doc1->add\_figure(std::move(new\_figure),id);

}else if(figures == "six\_angles"){

std::unique\_ptr<figure> new\_figure;

new\_figure=std::make\_unique<six\_angles>( six\_angles(std::cin));

doc1->add\_figure(std::move(new\_figure),id);

}else if(figures == "eight\_angles"){

std::unique\_ptr<figure> new\_figure;

new\_figure=std::make\_unique<eight\_angles>( eight\_angles(std::cin));

doc1->add\_figure(std::move(new\_figure),id);

}

}

main.cpp:

#include <iostream>

#include <string>

#include <stdio.h>

#include <vector>

#include <memory>

#include <fstream>

#include "figure.h"

#include "five\_angles.h"

#include "six\_angles.h"

#include "eight\_angles.h"

#include "document.h"

#include "factory.h"

int main() {

std::string command;

factory fact;

std::unique\_ptr<document> doc1;

while(std::cin >> command){

if(command=="new"){

doc1=std::make\_unique<document>();

}else if(command=="save"){

std::string path;

std::cin >> path;

std::ofstream os(path);

doc1->save(os);

os.close();

}else if(command=="load"){

std::string path;

std::cin >> path;

std::ifstream is(path);

if(is) {

doc1->load(is);

}else {

std::cout << "No such file\n";

}

is.close();

}else if(command=="add"){

fact.construct(doc1);

}else if(command=="remove"){

size\_t id;

std::cin >> id;

doc1->remove\_figure(id);

}else if(command=="show"){

doc1->show(std::cout);

}else if(command == "undo"){

doc1->undo();

}

}

return 0;

}

2. **Ссылка на репозиторий в GitHub:**

https://github.com/Suvorova-Sofya/oop\_exercise\_07

3.**Набор testcases:**

test1:

new

add five\_angles 0

1 1 2 2 3 3 4 4 5 5

add six\_angles 1

1 1 2 2 3 3 4 4 5 5 6 6

add eight\_angles 2

1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

show

remove 1

show

remove 0

show

test2:

new

add five\_angles 0

1 1 2 2 3 3 4 4 5 5

add six\_angles 1

1 1 2 2 3 3 4 4 5 5 6 6

add eight\_angles 2

1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

save tfile.txt

show

remove 0

remove 0

remove 0

show

load tfile.txt

show

test3:

new

add five\_angles 0

1 1 2 2 3 3 4 4 5 5

add six\_angles 1

1 1 2 2 3 3 4 4 5 5 6 6

add eight\_angles 2

1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

show

undo

show

remove 0

show

undo

show

undo

show

4.**Результаты выполнения программы:**

test1:

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5

center 3 3

square 0

figure 1

cordinats 1 1 2 2 3 3 4 4 5 5 6 6

center 3.5 3.5

square 0

figure 2

cordinats 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

center 4.5 4.5

square 0

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5

center 3 3

square 0

figure 1

cordinats 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

center 4.5 4.5

square 0

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

center 4.5 4.5

square 0

test2:

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5

center 3 3

square 0

figure 1

cordinats 1 1 2 2 3 3 4 4 5 5 6 6

center 3.5 3.5

square 0

figure 2

cordinats 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

center 4.5 4.5

square 0

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5

center 3 3

square 0

figure 1

cordinats 1 1 2 2 3 3 4 4 5 5 6 6

center 3.5 3.5

square 0

figure 2

cordinats 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

center 4.5 4.5

square 0

test3:

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5

center 3 3

square 0

figure 1

cordinats 1 1 2 2 3 3 4 4 5 5 6 6

center 3.5 3.5

square 0

figure 2

cordinats 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

center 4.5 4.5

square 0

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5

center 3 3

square 0

figure 1

cordinats 1 1 2 2 3 3 4 4 5 5 6 6

center 3.5 3.5

square 0

remove 0

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5 6 6

center 3.5 3.5

square 0

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5

center 3 3

square 0

figure 1

cordinats 1 1 2 2 3 3 4 4 5 5 6 6

center 3.5 3.5

square 0

figure 0

cordinats 1 1 2 2 3 3 4 4 5 5

center 3 3

square 0

5**. Объяснение результатов работы программы:**

Пользователь вводит команду , и её дополнительные атрибуты (имя файла, имя фигуры, координаты, позицию). В зависимости от этого программы выполняет одно из семи команд: создание нового документа, загрузка документа в файл, выгрузка документа из файла, добавление фигуры, удаление фигуры, показ всех фигур с их характеристиками, отмена последнего действия.

6.**Вывод:**

В данной программе показывается ,каким образом можно создать собственный очень примитивный векторный графический редактор, чтобы наиболее простым образом показать, как происходит проектирование структуры классов.