**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 6**

Тема: Проектирование структуры классов

Студент: Калугин Кирилл Алексеевич

Группа: 80-207

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

1. Постановка задачи

Спроектировать простейший «графический» векторный редактор.

Требование к функционалу редактора:

· создание нового документа

· импорт документа из файла

· экспорт документа в файл

· создание графического примитива (согласно варианту задания)

· удаление графического примитива

· отображение документа на экране (печать перечня графических объектов и их характеристик в std::cout)

· реализовать операцию undo, отменяющую последнее сделанное действие. Должно действовать для операций добавления/удаления фигур.

Требования к реализации:

· Создание графических примитивов необходимо вынести в отдельный класс – Factory.

· Сделать упор на использовании полиморфизма при работе с фигурами;

· Взаимодействие с пользователем (ввод команд) реализовать в функции main;

Вариант - пятиугольник, шестиугольник, восьмиугольник;

1. Описание программы

Программа позволяет пользователю создавать и записывать в рабочий файл фигуры из варианта. Фигуры могут быть вставлены на любое существующее место в списке. Удаление фигур также производится по номеру. В качестве альтернативного варианта имеется возможность загрузки данных из бинарного файла. Рабочий файл также может быть сохранен в виде бинарного. Взаимодействие с пользователем происходит при помощи интерактивного меню. Все фигуры являются правильными, и строго сориентированными относительно координатной сетки.

1. Набор тестов

PS C:\Users\Кирилл\Desktop\ООП\_7> .\a.exe

1. New file.

2. Create figure.

3. Delete figure.

4. Save file.

5. Load file.

6. Print figures.

7. Cancel last action.

0. Exit.

Choose option: 1

Choose option: 2

What figure do you want to create?

1. Pentagon.

2. Hexagon.

3. Octogon.

1

Enter position of new figure: 1

Enter center coordinates and radius: 1 1 1

Choose option: 2

What figure do you want to create?

1. Pentagon.

2. Hexagon.

3. Octogon.

2

Enter position of new figure: 2

Enter center coordinates and radius: 2 2 2

Choose option: 6

----------

A = (1,2)

B = (1.88295,1.46947)

C = (1.71934,0.305342)

D = (0.561629,0.101206)

E = (0.00973193,1.13917)

----------

----------

A = (2,4)

B = (3.73205,3)

C = (3.73205,1)

D = (2,0)

E = (0.267949,1)

F = (0.267949,3)

----------

Choose option: 2

What figure do you want to create?

1. Pentagon.

2. Hexagon.

3. Octogon.

3

Enter position of new figure: 2

Enter center coordinates and radius: 3 3 3

Choose option: 6

----------

A = (1,2)

B = (1.88295,1.46947)

C = (1.71934,0.305342)

D = (0.561629,0.101206)

E = (0.00973193,1.13917)

----------

----------

A = (3,6)

B = (5.12132,5.12132)

C = (6,3)

D = (5.12132,0.87868)

E = (3,0)

F = (0.87868,0.87868)

G = (0,3)

H = (0.87868,5.12132)

----------

----------

A = (2,4)

B = (3.73205,3)

C = (3.73205,1)

D = (2,0)

E = (0.267949,1)

F = (0.267949,3)

----------

Choose option: 4

Enter file name: aaa

Successfuly saved.

Choose option: 1

Choose option: 6

File is empty.

Choose option: 5

Enter file name: aaa

Successfuly loaded.

Choose option: 6

----------

A = (1,2)

B = (1.88295,1.46947)

C = (1.71934,0.305342)

D = (0.561629,0.101206)

E = (0.00973193,1.13917)

----------

----------

A = (3,6)

B = (5.12132,5.12132)

C = (6,3)

D = (5.12132,0.87868)

E = (3,0)

F = (0.87868,0.87868)

G = (0,3)

H = (0.87868,5.12132)

----------

----------

A = (2,4)

B = (3.73205,3)

C = (3.73205,1)

D = (2,0)

E = (0.267949,1)

F = (0.267949,3)

----------

Choose option: 2

What figure do you want to create?

1. Pentagon.

2. Hexagon.

3. Octogon.

1

Enter position of new figure: 1

Enter center coordinates and radius: 1 1 1

Choose option: 3

Enter position of element to delete: 1

Choose option: 3

Enter position of element to delete: 1

Choose option: 6

----------

A = (3,6)

B = (5.12132,5.12132)

C = (6,3)

D = (5.12132,0.87868)

E = (3,0)

F = (0.87868,0.87868)

G = (0,3)

H = (0.87868,5.12132)

----------

----------

A = (2,4)

B = (3.73205,3)

C = (3.73205,1)

D = (2,0)

E = (0.267949,1)

F = (0.267949,3)

----------

Choose option: 7

Choose option: 7

Choose option: 6

----------

A = (1,2)

B = (1.88295,1.46947)

C = (1.71934,0.305342)

D = (0.561629,0.101206)

E = (0.00973193,1.13917)

----------

----------

A = (1,2)

B = (1.88295,1.46947)

C = (1.71934,0.305342)

D = (0.561629,0.101206)

E = (0.00973193,1.13917)

----------

----------

A = (3,6)

B = (5.12132,5.12132)

C = (6,3)

D = (5.12132,0.87868)

E = (3,0)

F = (0.87868,0.87868)

G = (0,3)

H = (0.87868,5.12132)

----------

----------

A = (2,4)

B = (3.73205,3)

C = (3.73205,1)

D = (2,0)

E = (0.267949,1)

F = (0.267949,3)

----------

Choose option: 0

1. Листинг программы

//Калугин Кирилл Алексеевич М8О-207Б-19

/\*Спроектировать простейший «графический» векторный редактор.

Требование к функционалу редактора:

· создание нового документа

· импорт документа из файла

· экспорт документа в файл

· создание графического примитива (согласно варианту задания)

· удаление графического примитива

· отображение документа на экране (печать перечня графических объектов и их характеристик в std::cout)

· реализовать операцию undo, отменяющую последнее сделанное действие. Должно действовать для операций добавления/удаления фигур.

Требования к реализации:

· Создание графических примитивов необходимо вынести в отдельный класс – Factory.

· Сделать упор на использовании полиморфизма при работе с фигурами;

· Взаимодействие с пользователем (ввод команд) реализовать в функции main;

Вариант - пятиугольник, шестиугольник, восьмиугольник;

\*/

#include <iostream>

#include <memory>

#include <windows.h>

#include <algorithm>

#include <list>

#include <math.h>

#include <string>

#include <stack>

#include <iterator>

using namespace std;

const double PI = 3.141592653589793238462643383279502884197169399375105820974944;

const int Five\_Id = 1;

const int Six\_Id = 2;

const int Eight\_Id = 3;

class Figure {//создание класса фигур

public:

virtual void points () {}

virtual void write(FILE\* out) {}

};

template <typename T>

class Five: public Figure {//создание класса-наследника для пятиугольников

public:

using type = T;

pair<T, T> O;

T R;

virtual void points () {

cout << "----------\n"<< "A = (" << O.first << "," << O.second + R << ")\n" << "B = ("

<< O.first + R \* cos (28.0 / 180.0 \* PI) << "," << O.second + R \* sin (28.0 / 180.0 \* PI)

<< ")\n" << "C = (" << O.first + R \* cos (- 44.0 / 180.0 \* PI) << ","

<< O.second + R \* sin (- 44.0 / 180.0 \* PI) << ")\n" << "D = ("

<< O.first + R \* cos (- 116.0 / 180.0 \* PI) << "," << O.second + R \* sin (- 116.0 / 180.0 \* PI)

<< ")\n" << "E = (" << O.first + R \* cos (- 188.0 / 180.0 \* PI)

<< "," << O.second + R \* sin (- 188.0 / 180.0 \* PI) << ")\n" << "----------\n";

}

virtual void write (FILE\* out) {

fwrite(&Five\_Id, sizeof(int), 1, out);

fwrite(&O.first, sizeof(T), 1, out);

fwrite(&O.second, sizeof(T), 1, out);

fwrite(&R, sizeof(T), 1, out);

}

Five () {};//конструктор класса

Five (T x, T y, T r): O(x, y), R(r) {};//конструктор класса

};

template <typename T>

class Six: public Figure {//создание класса-наследника для шестиугольников

public:

using type = T;

pair<T, T> O;

T R;

virtual void points () {//метод вывода вершин

cout << "----------\n" << "A = (" << O.first << "," << O.second + R << ")\n" << "B = ("

<< O.first + R \* cos (PI / 6.0) << "," << O.second + R \* sin (PI / 6.0)

<< ")\n" << "C = (" << O.first + R \* cos (- PI / 6.0) << ","

<< O.second + R \* sin (- PI / 6.0) << ")\n" << "D = (" << O.first << ","

<< O.second - R << ")\n" << "E = (" << O.first + R \* cos (- 5.0 \* PI / 6.0)

<< "," << O.second + R \* sin (- 5.0 \* PI / 6.0) << ")\n" << "F = ("

<< O.first + R \* cos (- 7.0 \* PI / 6.0) << ","

<< O.second + R \* sin (- 7.0 \* PI / 6.0) << ")\n" << "----------\n";

}

virtual void write (FILE\* out) {

fwrite(&Six\_Id, sizeof(int), 1, out);

fwrite(&O.first, sizeof(T), 1, out);

fwrite(&O.second, sizeof(T), 1, out);

fwrite(&R, sizeof(T), 1, out);

}

Six () {};//конструктор класса

Six (T x, T y, T r): O(x, y), R(r) {};//конструктор класса

};

template <typename T>

class Eight: public Figure {//создание класса-наследника для восьмиугольник

public:

using type = T;

pair<T, T> O;

T R;

virtual void points () {//метод вывода вершин

cout << "----------\n" << "A = (" << O.first << "," << O.second + R << ")\n" << "B = ("

<< O.first + R \* cos (PI / 4) << "," << O.second + R \* sin (PI / 4)

<< ")\n" << "C = (" << O.first + R << "," << O.second << ")\n"

<< "D = (" << O.first + R \* cos (- PI / 4) << "," << O.second + R \* sin (- PI / 4)

<< ")\n" << "E = (" << O.first << "," << O.second - R << ")\n"

<< "F = (" << O.first + R \* cos (- 3 \* PI / 4) << "," << O.second + R \* sin (- 3 \* PI / 4)

<< ")\n" << "G = (" << O.first - R << "," << O.second << ")\n" << "H = ("

<< O.first + R \* cos (- 5 \* PI / 4) << "," << O.second + R \* sin (- 5 \* PI / 4) << ") \n" << "----------\n";

}

virtual void write (FILE\* out) {

fwrite(&Eight\_Id, sizeof(int), 1, out);

fwrite(&O.first, sizeof(T), 1, out);

fwrite(&O.second, sizeof(T), 1, out);

fwrite(&R, sizeof(T), 1, out);

}

Eight () {};//конструктор класса

Eight (T x, T y, T r): O(x, y), R(r) {};//конструктор класса

};

template<class T, class FIGURE>

class Factory;

template <class T>

class Factory <T, Five <T>> {//класс Factory для создания пятиугольников

public:

static shared\_ptr <Figure> Create () {

T x;

T y;

T r;

cout << "Enter center coordinates and radius: ";

cin >> x;

cin >> y;

cin >> r;

Five <T>\* ret = new Five <T> (x, y, r);

return shared\_ptr <Figure> (ret);

}

static shared\_ptr <Figure> Read(FILE\* in) {

T x;

T y;

T r;

fread(&x, sizeof(T), 1, in);

fread(&y, sizeof(T), 1, in);

fread(&r, sizeof(T), 1, in);

Five <T>\* ret = new Five <T> (x, y, r);

return shared\_ptr <Figure> (ret);

}

};

template <class T>

class Factory <T, Six <T>> {//класс Factory для создания шестиугольников

public:

static shared\_ptr <Figure> Create () {

T x;

T y;

T r;

cout << "Enter center coordinates and radius: ";

cin >> x;

cin >> y;

cin >> r;

Six <T>\* ret = new Six <T> (x, y, r);

return shared\_ptr <Figure> (ret);

}

static shared\_ptr <Figure> Read(FILE\* in) {

T x;

T y;

T r;

fread(&x, sizeof(T), 1, in);

fread(&y, sizeof(T), 1, in);

fread(&r, sizeof(T), 1, in);

Six <T>\* ret = new Six <T> (x, y, r);

return shared\_ptr <Figure> (ret);

}

};

template <class T>

class Factory <T, Eight <T>> {//класс Factory для создания восьмиугольников

public:

static shared\_ptr <Figure> Create () {

T x;

T y;

T r;

cout << "Enter center coordinates and radius: ";

cin >> x;

cin >> y;

cin >> r;

Eight <T>\* ret = new Eight <T> (x, y, r);

return shared\_ptr <Figure> (ret);

}

static shared\_ptr <Figure> Read(FILE\* in) {

T x;

T y;

T r;

fread(&x, sizeof(T), 1, in);

fread(&y, sizeof(T), 1, in);

fread(&r, sizeof(T), 1, in);

Eight <T>\* ret = new Eight <T> (x, y, r);

return shared\_ptr <Figure> (ret);

}

};

template <class T>

class File {//класс рабочего файла

public:

int size = 0;

struct Action {

int action;

int place;

shared\_ptr <Figure> figure;

};

stack <Action> uvault;

list <shared\_ptr <Figure>> vault;

void NFile () {

vault.erase (vault.begin (), vault.end ());

while (!uvault.empty ()) {

uvault.pop ();

}

}

void Add () {//функция добавления новой фигуры

int id;

int pos;

Action act;

list <shared\_ptr <Figure>> :: iterator it = vault.begin ();

cout << "What figure do you want to create?\n\t1. Pentagon.\n\t2. Hexagon.\n\t3. Octogon." << endl;

cin >> id;

if ((id != 1) && (id != 2) && (id != 3)) {

throw "ERROR\nIncorrect number.";

}

cout << "Enter position of new figure: ";

cin >> pos;

shared\_ptr <Figure> nf;

if ((pos > size + 1) || (pos < 1)) {

throw "ERROR\nThere is no such position in file.";

}

if (id == Five\_Id) {

nf = Factory<T, Five <T>>::Create ();

} else if (id == Six\_Id) {

nf = Factory<T, Six <T>>::Create ();

} else if (id == Eight\_Id) {

nf = Factory<T, Eight <T>>::Create ();

}

act.action = 1;

act.place = pos;

for (int i = 1; i < pos; i ++) {

it ++;

}

vault.insert (it, nf);

uvault.push (act);

size ++;

}

void Delete () {//функция удаления новой фигуры

int pos;

Action act;

list <shared\_ptr <Figure>> :: iterator it = vault.begin ();

cout << "Enter position of element to delete: ";

cin >> pos;

if (vault.empty()) {

throw ("ERROR\nFile is empty.");

} else {

for (int i = 1; i < pos; i ++) {

it++;

}

act.action = 2;

act.place = pos;

act.figure = \*it;

vault.erase (it);

uvault.push(act);

}

}

void Save () {//функция сохранения файла

string s;

int size = vault.size();

list< shared\_ptr <Figure> >::iterator it = vault.begin();

cout << "Enter file name: ";

cin >> s;

FILE \*file = fopen(s.c\_str(),"wb");

if (file == NULL) {

throw "ERROR\nNo such file.";

} else {

fwrite(&size, sizeof(int), 1, file);

for (it; it != vault.end (); it++) {

(\*it)->write (file);

}

fclose(file);

cout << "Successfuly saved." << endl;

}

}

void Load () {//функция загрузки файла

string s;

int size = vault.size();

int id;

cout << "Enter file name: ";

cin >> s;

FILE \*file = fopen(s.c\_str(),"rb");

if (file == NULL) {

throw "ERROR\nNo such file.";

} else {

NFile ();

fread(&size, sizeof(int), 1, file);

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

fread(&id, sizeof(int), 1, file);

if (id == Five\_Id) {

vault.push\_back(Factory<T, Five<T>>:: Read(file));

} else if (id == Six\_Id){

vault.push\_back(Factory<T, Six<T>>:: Read(file));

} else if (id == Eight\_Id){

vault.push\_back(Factory<T, Eight<T>>:: Read(file));

}

}

fclose(file);

cout << "Successfuly loaded." << endl;

}

}

void Undo () {//функция отмены последнего действия

if (!uvault.empty ()) {

Action last = uvault.top ();

int pos = last.place;

list <shared\_ptr <Figure>> :: iterator it = vault.begin ();

if (last.action == 1) {

if (vault.empty()) {

throw ("ERROR\nFile is empty.");

} else {

for (int i = 1; i < pos; i ++) {

it ++;

}

vault.erase (it);

}

} else if (last.action == 2) {

shared\_ptr <Figure> nf;

if ((pos > size + 1) || (pos < 1)) {

throw "ERROR\nThere is no such position in file.";

}

nf = last.figure;

for (int i = 1; i < pos; i ++) {

it ++;

}

vault.insert (it, nf);

size ++;

}

uvault.pop ();

} else

throw "ERROR\nNo actions to cancel.";

}

};

int main () {

int option = 99;

File <double> WF;

cout << "1. New file.\n";

cout << "2. Create figure.\n";

cout << "3. Delete figure.\n";

cout << "4. Save file.\n";

cout << "5. Load file.\n";

cout << "6. Print figures.\n";

cout << "7. Cancel last action.\n";

cout << "0. Exit.\n";

while (option != 0) {

cout << "Choose option: ";

cin >> option;

switch (option) {

case 1: {

try {

WF.NFile ();

}

catch (const char\* exc) {

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 4));

cerr << exc << endl;

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 7));

}

break;

}

case 2: {

try {

WF.Add ();

}

catch (const char\* exc) {

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 4));

cerr << exc << endl;

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 7));

}

break;

}

case 3: {

try {

WF.Delete ();

}

catch (const char\* exc) {

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 4));

cerr << exc << endl;

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 7));

}

break;

}

case 4: {

try {

WF.Save();

}

catch (const char\* exc) {

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 4));

cerr << exc << endl;

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 7));

}

break;

}

case 5: {

try {

WF.Load();

}

catch (const char\* exc) {

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 4));

cerr << exc << endl;

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 7));

}

break;

}

case 6: {

try {

if (WF.vault.size() == 0) {

cout << "File is empty." << endl;

} else {

list< shared\_ptr <Figure> >::iterator curr = WF.vault.begin ();

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

(\*curr)->points();

++curr;

}

}

}

catch (const char\* exc) {

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 4));

cerr << exc << endl;

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 7));

}

break;

}

case 7: {

try {

WF.Undo ();

}

catch (const char\* exc) {

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 4));

cerr << exc << endl;

SetConsoleTextAttribute(hConsole, (WORD) ((0 << 4) | 7));

}

break;

}

case 0: {

break;

}

default: {

cout << "Wrong command!\n";

break;

}

}

}

}

1. Выводы

Литература

# Список list в С++: полный материал[Электронный ресурс]

URL:https://codelessons.ru/cplusplus/spisok-list-v-s-polnyj-material.htmlдата обращения (15.12.2020)

# List[Электронный ресурс]

URL:https://metanit.com/cpp/tutorial/7.6.phpдата обращения (15.12.2020)

# Стек (stack) в C++: легко и просто![Электронный ресурс]

URL:https://codelessons.ru/cplusplus/realizaciya-steka-stack-v-c.htmlдата обращения (15.12.2020)

# Класс stack[Электронный ресурс]

URL:https://docs.microsoft.com/ru-ru/cpp/standard-library/stack-class?view=msvc-160&viewFallbackFrom=vs-2019дата обращения (15.12.2020)