// virtual\_file.cpp: определяет точку входа для консольного приложения.

//

#include "stdafx.h"

// Abstract.cpp : Defines the entry point for the console application.

//

//#include "stdafx.h"

#include <iostream>

#include <fstream>

#include <conio.h>

#include <math.h>

#ifndef MAX

# define MAX 100

#endif

class Shape

{

protected:

double x, y;

public:

void set(double ix, double iy = 0)

{

x = ix;

y = iy;

}

Shape(double ix = 0, double iy = 0)

{

x = ix;

y = iy;

}

virtual ~Shape()

{

std::cout << "Deleting Shape\n";

}

virtual void print() = 0;

virtual unsigned type() const = 0;

virtual void store(std::ofstream& s)

{

s << x << std::endl << y << std::endl;

}

virtual void load(std::ifstream& s)

{

s >> x >> y;

}

};

class Triangle : public Shape

{

protected:

double z;

public:

Triangle(double ix, double iy, double iz = 0) : Shape(ix, iy)

{

z = iz;

}

~Triangle()

{

std::cout << "Deleting Triangle\n";

}

void print()

{

std::cout << " A triangle with sides " << x << " and " << y << " and " << z << "\n";

}

unsigned type() const

{

return 1;

}

virtual void store(std::ofstream& s)

{

Shape::store(s);

s << z << std::endl;

}

virtual void load(std::ifstream& s)

{

Shape::load(s);

s >> z;

}

};

class Rectangle : public Shape

{

public:

Rectangle(double ix, double iy) : Shape(ix, iy)

{

}

~Rectangle()

{

std::cout << "Deleting Rectangle\n";

}

void print()

{

std::cout << " A rectangle with sides " << x << " and " << y << "\n";

}

unsigned type() const

{

return 2;

}

};

class Circle : public Shape

{

public:

Circle(double ix) : Shape(ix, 0)

{

}

~Circle()

{

std::cout << "Deleting Circle\n";

}

void print()

{

std::cout << " Circle with radius " << x << "\n";

}

unsigned type() const

{

return 3;

}

};

/////////////////////////////////////////////////////////////////////

void clear(Shape \*arr[])

{

for (unsigned i = 0; i < MAX; i++)

{

if (arr[i]) delete arr[i];

arr[i] = NULL;

}

}

void store(std::ofstream& s, Shape \*arr[])

{

unsigned int i, count = 0;

for (i = 0; i < MAX; i++)

if (arr[i]) count++;

s << count << std::endl;

for (i = 0; i < MAX; i++)

{

if (!arr[i]) continue;

s << arr[i]->type() << std::endl;

arr[i]->store(s);

}

}

int load(std::ifstream& s, Shape \*arr[])

{ clear(arr);

int i, count;

s >> count;

for (i = 0; i < count && i < MAX; i++)

{ unsigned type = 0;

s >> type;

switch (type)

{ case 1: arr[i] = new Triangle(0, 0); break;

case 2: arr[i] = new Rectangle(0, 0); break;

case 3: arr[i] = new Circle(0); break;

default: throw "Unknown object";

}

arr[i]->load(s);

}

return count;

}

/////////////////////////////////////////////////////////////////////

int main()

{

Shape\* arr[MAX];

int count = 0;

for (unsigned i = 0; i < MAX; i++)

{

arr[i] = NULL;

}

int j = 0, ch;

do

{

std::cout

<< " <1> - Add triangle \n"

<< " <2> - Add Rectanglre \n"

<< " <3> - Add Circle \n"

<< " <4> - Print all element and your's metods \n"

<< " <5> - Save all element in file \n"

<< " <6> - Load all element of file \n"

<< " <7> - Clear all element \n"

<< " <8> - Exit \n";

std::cin >> ch; // menu

switch (ch)

{

case 1:

std::cout << "Enter the number of new triangle - \n" ;

std::cin >> j;

for (int i = count; i < count + j; i++)

{

arr[i] = new Triangle(1 + rand() % MAX, 1 + rand() % MAX, 1 + rand() % MAX);

arr[i]->print();

}

count += j;

break;

case 2:

std::cout << "Enter the number of new rectangle - \n";

std::cin >> j;

for (int i = count; i < count + j; i++)

{

arr[i] = new Rectangle(1 + rand() % MAX, 1 + rand() % MAX);

arr[i]->print();

}

count += j;

break;

case 3:

std::cout << "Enter the number of new circle - \n";

std::cin >> j;

for (int i = count; i < count + j; i++)

{

arr[i] = new Circle(1 + rand() % MAX);

arr[i]->print();

}

count += j;

break;

case 4:

for (int i = 0; i<count; i++)

{

arr[i]->print();

//arr[i]->pole();

//arr[i]->objetnosc();

}

break;

case 5:

{

std::ofstream f;

f.open("shapes.txt");

store(f, arr);

f.close();

}

break;

case 6:

{

std::ifstream f;

f.open("shapes.txt");

count = load(f, arr);

f.close();

}

break;

case 7:

clear(arr);

count = 0;

break;

}

} while (ch != 8);

clear(arr);

system("pause");

return 0;

}