МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(московский ордена Ленина, ордена Октябрьской Революции

авиационный институт имени Серго Орджоникидзе)

(национальный исследовательский университет)

МАИ

Кафедра 304 «Вычислительные машины, системы и сети»

Отчёт по Лабораторной работе №1

По учебной дисциплине «Объектно-ориентированное программирование»

Вариант №3

Выполнил:

студент группы М3О-325Бк-21 Вельковский З.И.\_\_\_\_\_\_\_\_\_

Принял:

Доцент кафедры 304, к.т.н. Новиков П.В.

Старший преподаватель кафедры 304, Ивашенцев И.В.

Москва 2023

# Постановка задачи

ПРОСТЕЙШАЯ ИЕРАРХИЯ КЛАССОВ

**Тема работы**

Классы и объекты. Инкапсуляция.  
**Цель работы**

Изучение понятий «объект» и «класс», а также «поля данных» и «методы» класса. Реализация иерархии для раннего и позднего связывания, а также освоение технологии разработки классов и создания объектов на примере языков программирования C++ Visual Studio. Работа с открытыми и закрытыми полями данных и методами.

**Индивидуальное задание на выполнение**

Разработать классы Location, Point и Figure (Aster). Графическая фигура – Астероид.

# Проектирование классов

### Класс Location

Поля (защищённые):

x – координата1

y – координата2

Методы (открытые):

Конструктор: входные параметры: значения координат; в конструкторе происходит их первоначальная установка

### Класс Point : публичное наследование от класса Location

Методы (открытые):

Конструктор: входные параметры: значения координат; в конструкторе происходит их первоначальная установка

show – функция отрисовки; отрисовывает точку

move – функция перемещения; сменяет координаты на указанное значение

### Класс Aster : публичное наследование от класса Point

Поля (защищённые):

width – ширина линии, которой рисуется фигура

Методы (открытые):

Конструктор: входные параметры: значения координат; в конструкторе происходит их первоначальная установка

onKeyDown - функция обработки нажатия клавиш; сменяет координаты, при нажатии кнопок (W, A, S, D)

show - функция отрисовки; отрисовывает фигуру

# Текст программы с поздним связыванием

### oop\_1.cpp

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* КАФЕДРА № 304 2 КУРС \*

\*------------------------------------------------------------\*

\* Project Type : Win32 Console Application \*

\* Project Name : LR OOP1 \*

\* File Name : LR OOP1.cpp \*

\* Language : C++ \*

\* Programmers(s) : Вельковский З.И. \*

\* Created : 15/08/23 \*

\* Last Revision : 15/08/23 \*

\* Comments(s) : Простейшая иерархия классов \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdlib.h>

#include <string.h>

#include <tchar.h>

#include <windows.h>

#include "Figures.hpp"

static TCHAR szWindowClass[] = \_T("DesktopApp");

static TCHAR szTitle[] = \_T("LR 1");

HINSTANCE hInst;

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

int WINAPI WinMain(\_In\_ HINSTANCE hInstance, \_In\_opt\_ HINSTANCE hPrevInstance, \_In\_ LPSTR lpCmdLine, \_In\_ int nCmdShow) {

WNDCLASSEX wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(wcex.hInstance, IDI\_APPLICATION);

wcex.hCursor = LoadCursor(NULL, IDC\_ARROW);

wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);

wcex.lpszMenuName = NULL;

wcex.lpszClassName = szWindowClass;

wcex.hIconSm = LoadIcon(wcex.hInstance, IDI\_APPLICATION);

if (!RegisterClassEx(&wcex)) {

MessageBox(NULL, \_T("Call to RegisterClassEx failed!"), szTitle, NULL);

return 1;

}

hInst = hInstance;

HWND hWnd = CreateWindowEx(WS\_EX\_OVERLAPPEDWINDOW, szWindowClass, szTitle,

WS\_OVERLAPPEDWINDOW, CW\_USEDEFAULT, CW\_USEDEFAULT,

1024, 512, NULL, NULL, hInstance, NULL);

if (!hWnd) {

MessageBox(NULL, \_T("Call to CreateWindow failed!"), szTitle, NULL);

return 1;

}

ShowWindow(hWnd, nCmdShow);

MSG msg;

while (GetMessage(&msg, NULL, 0, 0)) {

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return (int)msg.wParam;

}

Aster aster(0, 0);

Point\* facePtr = &aster;

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam) {

PAINTSTRUCT ps;

HDC hdc;

TCHAR greeting[] = \_T("Hello, Windows desktop!");

switch (message) {

case WM\_KEYDOWN:

aster.onKeyDown(wParam);

InvalidateRect(hWnd, NULL, TRUE);

break;

case WM\_PAINT:

hdc = BeginPaint(hWnd, &ps);

facePtr->show(hdc);

//aster.show(hdc);

EndPaint(hWnd, &ps);

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

break;

}

return 0;

}

### Figures.hpp

#ifndef LR\_OOP1\_FIGURES

#define LR\_OOP1\_FIGURES

#include <windows.h>

class Location;

class Point;

class Aster;

class Location

{

protected:

int x;

int y;

public:

Location(int x, int y);

};

class Point : public Location

{

public:

Point(int x, int y);

virtual void show(HDC hdc);

void move(int offsetX, int offsetY);

virtual void onKeyDown(WPARAM wParam) {};

};

class Aster : public Point

{

protected:

int width = 4;

public:

Aster(int x, int y);

void onKeyDown(WPARAM wParam);

void show(HDC hdc);

};

#endif

### Figures.cpp

#include "Figures.hpp"

Location::Location(int x, int y)

{

this->x = x;

this->y = y;

}

Point::Point(int x, int y) : Location(x, y) {};

void Point::show(HDC hdc)

{

SetPixel(hdc, x, y, RGB(0, 0, 0));

}

void Point::move(int offsetX, int offsetY)

{

this->x += offsetX;

this->y += offsetY;

}

Aster::Aster(int x, int y) : Point(x, y) {};

void Aster::onKeyDown(WPARAM wParam) {

int step = 5;

switch (wParam) {

case 'W':

move(0, -step);

break;

case 'S':

move(0, step);

break;

case 'A':

move(-step, 0);

break;

case 'D':

move(step, 0);

break;

}

}

void Aster::show(HDC hdc)

{

HPEN pen1= CreatePen(PS\_SOLID, width, RGB(0, 0, 0));

HPEN pen2 = CreatePen(PS\_SOLID, width, RGB(255, 255, 255));

SelectObject(hdc, pen1);

Ellipse(hdc, 100 + x + width, 100 + y + width, 200 + x + width, 200 + y + width);

Ellipse(hdc, 120 + x + width, 120 + y + width, 130 + x + width, 130 + y + width);

Ellipse(hdc, 130 + x + width, 140 + y + width, 142 + x + width, 152 + y + width);

Ellipse(hdc, 140 + x + width, 175 + y + width, 160 + x + width, 190 + y + width);

SelectObject(hdc, pen2);

Pie(hdc, 90 + x + width, 170 + y + width, 158 + x + width, 190 + y + width, 100 + x + width, 100 + y + width, 200 + x + width, 200 + y + width);

Pie(hdc, 180 + x + width, 100 + y + width, 220 + x + width, 160 + y + width, 100 + x + width, 100 + y + width, 200 + x + width, 200 + y + width);

SelectObject(hdc, pen1);

Ellipse(hdc, 90 + x + width, 170 + y + width, 130 + x + width, 190 + y + width);

Ellipse(hdc, 170 + x + width, 122 + y + width, 200 + x + width, 170 + y + width);

SelectObject(hdc, pen2);

Pie(hdc, 80 + x + width, 155 + y + width, 125 + x + width, 195 + y + width, 90 + x + width, 170 + y + width, 130 + x + width, 190 + y + width);

SelectObject(hdc, GetStockObject(BLACK\_PEN));

DeleteObject(pen1);

DeleteObject(pen2);

}

# Текст программы с ранним связыванием

### oop1

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* КАФЕДРА № 304 2 КУРС \*

\*------------------------------------------------------------\*

\* Project Type : Win32 Console Application \*

\* Project Name : LR OOP1 \*

\* File Name : LR OOP1.cpp \*

\* Language : C++ \*

\* Programmers(s) : Вельковский З.И. \*

\* Created : 15/08/23 \*

\* Last Revision : 15/08/23 \*

\* Comments(s) : Простейшая иерархия классов \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#pragma once

//#include <string>

//#include <iostream>

#include <conio.h>

#include <windows.h>

#include <iostream>

#include "Figure\_.hpp"

using namespace std;

//макрос для определения кода нажатой клавиши

#define KEY\_DOWN(vk\_code) ((GetAsyncKeyState(vk\_code) & 0x8000) ? 1 : 0)

HDC hdc;// Объявим контекст устройства

HWND GetConcolWindow(); //указатель на консольное окно

int main()

{

//получим дескриптор консольного окна

HWND hwnd = GetConcolWindow();

if (hwnd == nullptr)

{

exit(1);

}

hdc = GetWindowDC(hwnd);

if (hdc == 0)

{

exit(2);

}

Aster aster(0, 0);

Point\* facePtr = &aster;

char key;

bool run = true;

while (run)

{

key = \_getch();

aster.onKeyDown(key);

}

return 0;

}

HWND GetConcolWindow()

{

char str[128];

// char title[128]="xxxxxxxxxxxxxxxxxx";

LPWSTR title = (LPWSTR)"xxxxxxxxxxxxxxxxxx"; //новая версия Windows

GetConsoleTitle((LPWSTR)str, sizeof((LPWSTR)str)); // получить заголовок окна

SetConsoleTitle(title); // установить новый заголовок окна

Sleep(100); // ждем смены заголовка окна (100 мс);

HWND hwnd = FindWindow(NULL, (LPWSTR)title);// определяем дескриптор окна

SetConsoleTitle((LPWSTR)str); //возвращаем прежний заголовок

return hwnd;//вернуть дескриптор окна

}

### Figures.hpp

#ifndef LR\_OOP1\_FIGURES

#define LR\_OOP1\_FIGURES

#include <windows.h>

class Location;

class Point;

class Aster;

class Location

{

protected:

int x;

int y;

public:

Location(int x, int y);

};

class Point : public Location

{

public:

Point(int x, int y);

virtual void show(HDC hdc);

void move(int offsetX, int offsetY);

virtual void onKeyDown(WPARAM wParam) {};

};

class Aster : public Point

{

protected:

int width = 4;

public:

Aster(int x, int y);

void onKeyDown(WPARAM wParam);

void show(HDC hdc);

};

#endif

### Figures.cpp

#include "Figures.hpp"

Location::Location(int x, int y)

{

this->x = x;

this->y = y;

}

Point::Point(int x, int y) : Location(x, y) {};

void Point::show(HDC hdc)

{

SetPixel(hdc, x, y, RGB(0, 0, 0));

}

void Point::move(int offsetX, int offsetY)

{

this->x += offsetX;

this->y += offsetY;

}

Aster::Aster(int x, int y) : Point(x, y) {};

void Aster::onKeyDown(WPARAM wParam) {

int step = 5;

switch (wParam) {

case 'W':

move(0, -step);

break;

case 'S':

move(0, step);

break;

case 'A':

move(-step, 0);

break;

case 'D':

move(step, 0);

break;

}

}

void Aster::show(HDC hdc)

{

HPEN pen1= CreatePen(PS\_SOLID, width, RGB(0, 0, 0));

HPEN pen2 = CreatePen(PS\_SOLID, width, RGB(255, 255, 255));

SelectObject(hdc, pen1);

Ellipse(hdc, 100 + x + width, 100 + y + width, 200 + x + width, 200 + y + width);

Ellipse(hdc, 120 + x + width, 120 + y + width, 130 + x + width, 130 + y + width);

Ellipse(hdc, 130 + x + width, 140 + y + width, 142 + x + width, 152 + y + width);

Ellipse(hdc, 140 + x + width, 175 + y + width, 160 + x + width, 190 + y + width);

SelectObject(hdc, pen2);

Pie(hdc, 90 + x + width, 170 + y + width, 158 + x + width, 190 + y + width, 100 + x + width, 100 + y + width, 200 + x + width, 200 + y + width);

Pie(hdc, 180 + x + width, 100 + y + width, 220 + x + width, 160 + y + width, 100 + x + width, 100 + y + width, 200 + x + width, 200 + y + width);

SelectObject(hdc, pen1);

Ellipse(hdc, 90 + x + width, 170 + y + width, 130 + x + width, 190 + y + width);

Ellipse(hdc, 170 + x + width, 122 + y + width, 200 + x + width, 170 + y + width);

SelectObject(hdc, pen2);

Pie(hdc, 80 + x + width, 155 + y + width, 125 + x + width, 195 + y + width, 90 + x + width, 170 + y + width, 130 + x + width, 190 + y + width);

SelectObject(hdc, GetStockObject(BLACK\_PEN));

DeleteObject(pen1);

DeleteObject(pen2);

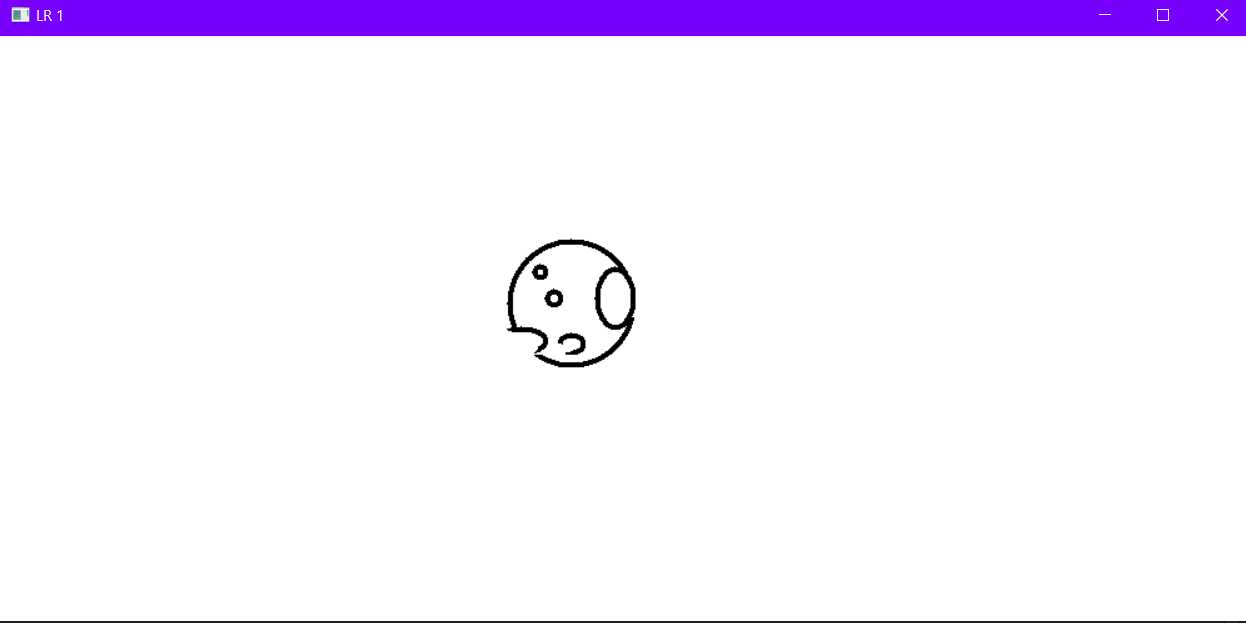
}

# Результат работы программы

При запуске программы появляется окно, с нарисованной фигурой (астеройдом):



При нажатии клавиш W, A, S, D фигура будет перемещаться вверх, влево, вниз, вправо соответственно (на рисунке пример фигуры после некоторых перемещений):



# Вывод

В результате выполнения лабораторной работы были изучены понятия «экземпляр» и «класс», «поля данных» и «методы» класса. Была основана разработка классов и создание их экземпляров в языке программирования C++ на базе редактора кода Visual Studio. Были рассмотрены случаи раннего и позднего связывания. В случае раннего связывания вызов функции сопоставляются с вызовом на этапе компиляции программы. При позднем связывании это происходит по ходу выполнения программы.

## Список источников:

1. Лекционные материалы по дисциплине «Объектно-ориентированное программирование» (2023г.)
2. Гайд по фигурам (КАК рисовать в Win32 API):  
   <http://radiofront.narod.ru/htm/prog/htm/winda/api/paint.html>
3. Объектно-ориентированное программирование МАИ Новиков П.В.: <https://docs.yandex.ru/docs/view?url=ya-mail%3A%2F%2F182677259885232628%2F1.2&name=%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%9C%D0%90%D0%98%20%D0%9D%D0%BE%D0%B2%D0%B8%D0%BA%D0%BE%D0%B2%20%D0%9F.%D0%92..pdf&uid=276902236>