

# СТЕКОВІ ОБ'ЄКТИ. КОНСТРУКТОР, ДЕКТРУКТОР, ВІДОБРАЖЕННЯ, ПЕРЕДАЧА

## Лабораторна робота №1

Мета:

- навчитися створювати об'єкти
- отримати розуміння створення об'єкта на стеку, а також передачу об'єкта по значенню.

## 1 ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

Варіант1. Створити клас даних <Вікно> та клас відображення даних <View>. Об'єкт відображення конструюється на стеку функції main() об'єктом даних, що заздалегідь створений на стеку. Передавати <Вікно> як значення. <Вікно> має всі публічні поля та методи. <View> лише виконує відображення даних у форматі <Назва поля>=<Значення>.

## 2 РОЗРОБКА ПРОГРАМИ

Для реалізації програми було створено:

- клас <Вікно> з полями ID вікна, x1, y1, x2, y2;
- клас <View> з методом відображення даних;

### 2.1 Засоби

У розробленій програмі використані наступні засоби ООП:

- розділення програми на ієрархію класів (інкапсуляція);

### 2.2 Ієрархія та структура класів

На рис.2.2 наведена структура розроблених класів



Рисунок 2.2 – Ієрархія класів

## 2.3 Опис програми

На рис.2.3 наведена структура розробленого проекту

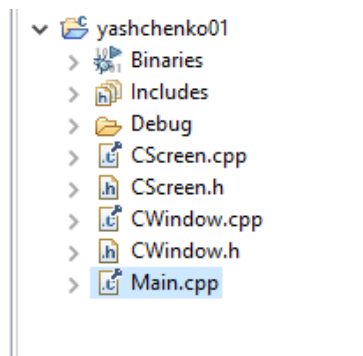


Рисунок 2.3 – Структура проекту

Призначення спроектованих класів наведено на рис.2.4



Класи	
Класи, структури, об'єднання та інтерфейси з коротким описом.	
 <b>CScreen</b>	Клас відображення даних
 <b>CWindow</b>	Клас даних предметної області Вікно

Рисунок 2.4 – Призначення класів

## 2.4 Важливі фрагменти програми

### 2.4.1 Файл CWindow.h

```
/*
 * CWindow.h
 * Created on: 10 сент. 2017 г.
 * Author: yashchenko
 * Description: CScreen declaration
 */

#ifndef CWINDOW_H_
#define CWINDOW_H_

//CLASS DECLARATION
class CWindow {
public:
    /**
     * Конструктор без параметра
     */
    CWindow();
    /**
     * Конструктор з параметрами
     * @param _ID Задає ID вікна
     * @param _x1 Задає параметр вікна x1
     */
}
```

```

    * @param _y1 Задає параметр вікна y1
    * @param _x2 Задає параметр вікна x2
    * @param _y2 Задає параметр вікна y2
    */
CWindow(int _ID, int _x1, int _y1, int _x2, int _y2);
/**
 * Конструктор копіювання
 * @param _cWindow Вихідний об'єкт із даними для копіювання
 */
CWindow(const CWindow& _cWindow);
/**
 * Деструктор
 */
~CWindow();
public:
    //ID вікна
    int ID;
    //параметр вікна x1
    int x1;
    //параметр вікна y1
    int y1;
    //параметр вікна x2
    int x2;
    //параметр вікна y2
    int y2;
};
#endif /* CWINDOW_H_ */

```

## 2.4.2 Файл CWindow.cpp

```

/*
 * CWindow.cpp
 * Created on: 10 сент. 2017 г.
 * Author: yashchenko
 * Description: CScreen implementation
 */

#include <CWindow.h>
#include <Windows.h>

//Конструктор без параметра
CWindow::CWindow() {
    ID = 0;
    x1 = 0;
    y1 = 0;
    x2 = 0;
    y2 = 0;
}

//Конструктор із параметрами
CWindow::CWindow (int _ID, int _x1, int _y1, int _x2, int _y2) {
    ID = _ID;
    x1 = _x1;
    y1 = _y1;
    x2 = _x2;
    y2 = _y2;
}

```

```

//Конструктор копіювання
CWindow::CWindow(const CWindow& _cWindow) {
    ID = _cWindow.ID;
    x1 = _cWindow.x1;
    y1 = _cWindow.y1;
    x2 = _cWindow.x2;
    y2 = _cWindow.y2;
}

//Деструктор
CWindow::~CWindow() {
    OutputDebugString("Destructor is called\n");
}

```

### 2.4.3 Файл CScreen.h

```

/*
 * Name: CScreen.h
 * Created on: 10 сент. 2017 г.
 * Author: yashchenko
 * Description: CScreen declaration
 */

#ifndef CSCREEN_H_
#define CSCREEN_H_

#include <CWindow.h>

//CLASS DECLARATION
class CScreen {
public:
    /**
     * Конструктор без параметра
     */
    CScreen();
    /**
     * Деструктор
     */
    ~CScreen();
public:
    /**
     * Функція відображення даних об'єкта класу CWindow
     */
    void view(CWindow& cWindow);
};

#endif /* CSCREEN_H_ */

```

### 2.4.4 Файл CScreen.cpp

```

/*
 * CScreen.cpp
 * Created on: 10 сент. 2017 г.
 * Author: yashchenko
 * Description: CScreen implementation
 */

```

```

#include <iostream>
#include <stdio.h>
#include <Windows.h>
#include <CScreen.h>

//конструктор без параметра
CScreen::CScreen() {

}

//Деструктор
CScreen::~CScreen() {

}

//Функція відображення даних об'єкта класу CWindow
void CScreen::view(CWindow& CWindow) {
    std::cout << "ID = " << CWindow.ID << std::endl;
    std::cout << "x1 = " << CWindow.x1 << std::endl;
    std::cout << "y1 = " << CWindow.y1 << std::endl;
    std::cout << "x2 = " << CWindow.x2 << std::endl;
    std::cout << "y2 = " << CWindow.y2 << std::endl;
}

```

#### 2.4.5 Файл Main.cpp

```

/*
 * Main.cpp
 * Created on: 10 сент. 2017 г.
 * Author: yashchenko
 * Description: point of entry
 */
#include <CScreen.h>
#include <iostream>
#include <stdio.h>
#include <Windows.h>

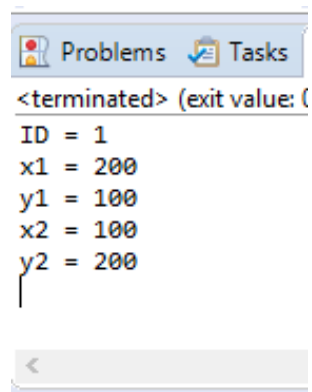
//point of entry
int main() {

    CWindow cWindow1(1, 200, 100, 100, 200);
    CScreen view;
    view.view(cWindow1);

}

```

### 3 РЕЗУЛЬТАТИ РОБОТИ

A screenshot of a debugger's 'Problems' or 'Tasks' window. The window has a title bar with 'Problems' and 'Tasks' tabs. The main area displays the following text: '<terminated> (exit value: (' on the first line, followed by 'ID = 1', 'x1 = 200', 'y1 = 100', 'x2 = 100', and 'y2 = 200' on subsequent lines. A vertical cursor is positioned at the end of the last line. At the bottom of the window, there is a horizontal scrollbar with a left-pointing arrow.

```
<terminated> (exit value: (  
ID = 1  
x1 = 200  
y1 = 100  
x2 = 100  
y2 = 200  
|
```

Рисунок 3.1 – Приклад роботи програми

## ВИСНОВКИ

В розробленій програмі я навчився створювати об'єкти, отримав розуміння створення об'єктів на стеку, а також передачі об'єкта по значенню.