**Лабораторна робота №1. Класи й об'єкти в С++**

**Мета.** Одержати практичні навички реалізації класів на С++.

**Основний зміст роботи.**

Написати програму, в якій створюються і руйнуються об'єкти, визначеного користувачем класу. Виконати дослідження викликів конструкторів і деструкторів.

**Порядок виконання роботи.**

1.Визначити клас користувача відповідно до варіанта завдання (дивись додаток).

2.Визначити в класі наступні конструктори: без параметрів, з параметрами, копіювання.

3.Визначити в класі деструктор.

4.Визначити в класі компонента-функції для перегляду і встановлення полів даних.

5.Визначити покажчик на компонент-функцію.

6.Визначити покажчик на екземпляр класу.

7.Написати демонстраційну програму, в якій створюються і руйнуються об'єкти класу користувача і кожен виклик конструктора і деструктора супроводжується видачею відповідного повідомлення (який об'єкт, який конструктор чи деструктор викликав).

8.Показати в програмі використання покажчика на об'єкт і покажчика на компонент-функцію.

**Задача №1**

**Умова задачі:**

|  |
| --- |
| **13.КРАЇНА** |
| ім'я- char\* |
| форма правління – char\* |
| площа – float |

**Код програми:**

**Country.h**

#pragma once

#include <iostream>

using namespace std;

class Country {

private:

char\* name; // ім'я країни

char\* government\_form; // форма правління

float area; // площа

public:

// Конструктори

Country();

Country(const char\* n, const char\* g, float a);

Country(const Country& other);

// Деструктор

~Country();

// Методи доступу

char\* getName() const;

char\* getGovernmentForm() const;

float getArea() const;

void setCountry(const char\* n, const char\* g, float a);

void setName(const char\* n);

void setGovernmentForm(const char\* g);

void setArea(float a);

void printCountry() const;

};

**Country.cpp**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "Country.h"

#include <cstring>

// Конструктор без параметрів

Country::Country() : name(nullptr), government\_form(nullptr), area(0) {

cout << "Викликано конструктор без параметрiв: " << this << endl;

}

// Конструктор з параметрами

Country::Country(const char\* n, const char\* g, float a) : area(a) {

name = new char[strlen(n) + 1];

strcpy(name, n);

government\_form = new char[strlen(g) + 1];

strcpy(government\_form, g);

cout << "Викликано конструктор з параметрами: " << this << endl;

}

// Конструктор копіювання

Country::Country(const Country& other) : area(other.area) {

name = new char[strlen(other.name) + 1];

strcpy(name, other.name);

government\_form = new char[strlen(other.government\_form) + 1];

strcpy(government\_form, other.government\_form);

cout << "Викликано конструктор копiювання: " << this << endl;

}

// Деструктор

Country::~Country() {

delete[] name;

delete[] government\_form;

cout << "Викликано деструктор: " << this << endl;

}

// --- Методи доступу ---

char\* Country::getName() const {

return name;

}

char\* Country::getGovernmentForm() const {

return government\_form;

}

float Country::getArea() const {

return area;

}

void Country::setName(const char\* n) {

delete[] name;

name = new char[strlen(n) + 1];

strcpy(name, n);

}

void Country::setGovernmentForm(const char\* g) {

delete[] government\_form;

government\_form = new char[strlen(g) + 1];

strcpy(government\_form, g);

}

void Country::setArea(float a) {

area = a;

}

void Country::setCountry(const char\* n, const char\* g, float a) {

setName(n);

setGovernmentForm(g);

setArea(a);

cout << "Задано країну: " << name << ", " << government\_form << ", площа: " << area << endl;

}

void Country::printCountry() const {

cout << "Назва: " << name<< "\t Форма правлiння: " << government\_form<< "\t Площа: " << area << " км^2" << endl;

}

**Main.cpp**

#include "Country.h"

#include <conio.h>

#include <iostream>

using namespace std;

// Тип для покажчика на компонент-функцію

typedef void (Country::\* FuncPointer)() const;

int main() {

setlocale(0, "ukr");

// 1. Виклик конструктора без параметрів

Country c1;

c1.setCountry("Україна", "Республiка", 603700);

c1.printCountry();

// 2. Виклик конструктора з параметрами

Country c2("Францiя", "Республiка", 643801);

c2.printCountry();

// 3. Виклик конструктора копіювання

Country c3 = c2;

c3.printCountry();

// 4. Покажчик на об'єкт

Country\* pCountry = new Country("Нiмеччина", "Федеративна республiка", 357588);

pCountry->printCountry();

// 5. Покажчик на метод (компонент-функцію)

FuncPointer f = &Country::printCountry;

(pCountry->\*f)(); // виклик через покажчик на метод

// 6. Статичний масив об'єктів

Country countries[3]; // викличе 3 рази конструктор без параметрів

countries[0].setCountry("Iталiя", "Республiка", 301340);

countries[1].setCountry("Iспанiя", "Монархiя", 505990);

countries[2].setCountry("Польща", "Республiка", 312696);

cout << "\nВиведення масиву країн:\n";

for (int i = 0; i < 3; i++) {

countries[i].printCountry();

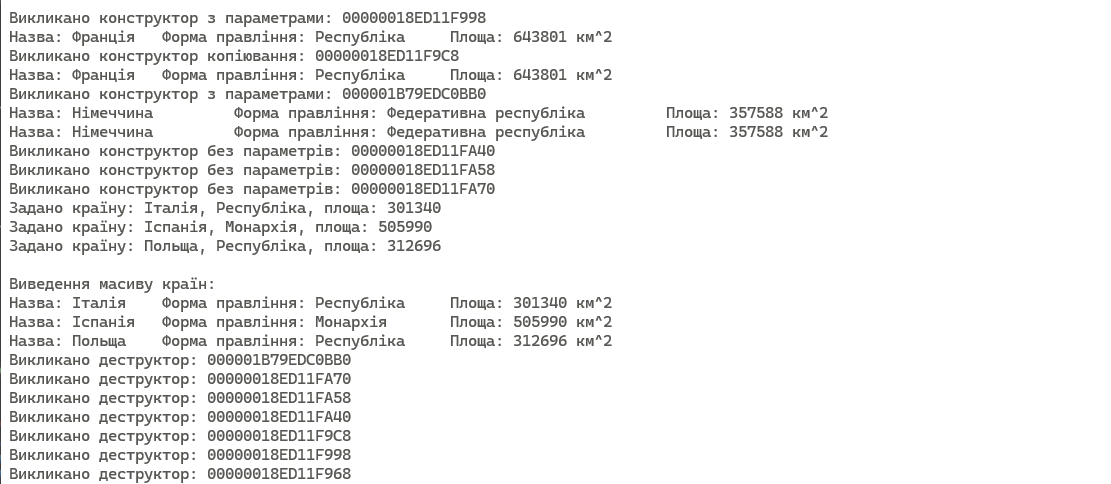
}

// прибирання

delete pCountry; // для динамічного одного об'єкта

return 0;

}**Результат роботи програми:**

****

**Задача №2**

**Умова задачі:**

1. Створити клас Rectangle, розробивши наступні елементи класу:
   * Поля:
     + int а, b;
   * Конструктор, що дозволяє створити екземпляр класу із заданими довжинами сторін.
   * Методи, що дозволяють:
     + вивести довжини сторін прямокутника на екран;
     + обчислити периметр прямокутника;
     + обчислити площу прямокутника.
   * Властивості:
     + одержати-встановити довжини сторін прямокутника (доступне для читання і запису);
     + які надають можливість встановити, чи є даний прямокутник квадратом (доступне тільки для читання).

**Постановка задачі :**

Розробити програмну реалізацію класу з тестовою програмою для перевірки функціонування всіх методів класу.

**Код програми:**

**Rectangle.h**

#pragma once

#include <iostream>

using namespace std;

class Rectangle

{

private:

int a, b;

public:

// Конструктор за замовчуванням

Rectangle();

// Конструктор з параметрами

Rectangle(int s1, int s2);

// Конструктор копіювання

Rectangle(const Rectangle& c);

// Деструктор

~Rectangle();

// Методи

void printSides();

int calculatePerimeter();

int calculateArea();

// Властивості

int getA();

int getB();

void setA(int s);

void setB(int s);

bool isSquare();

};

**Rectangle.cpp**

#include "Rectangle.h"

// Конструктор за замовчуванням

Rectangle::Rectangle() : a(0), b(0) {

cout << "Створено прямокутник за замовчуванням: " << a << " x " << b << " ("<<this<<")" << endl;

}

// Конструктор з параметрами

Rectangle::Rectangle(int s1, int s2) : a(s1), b(s2) {

cout << "Створено прямокутник: " << a << " x " << b << " (" << this << ")" << endl;

}

// Конструктор копіювання

Rectangle::Rectangle(const Rectangle& c) : a(c.a), b(c.b) {

cout << "Викликано конструктор копiювання: " << a << " x " << b << " (" << this << ")" << endl;

}

// Деструктор

Rectangle::~Rectangle() {

cout << "Викликано деструктор для прямокутника: " << a << " x " << b<< " (" << this << ")" << endl;

}

// Методи

void Rectangle::printSides() {

cout << "Сторони прямокутника: a = " << a << ", b = " << b << " (" << this << ")" << endl;

}

int Rectangle::calculatePerimeter() {

return 2 \* (a + b);

}

int Rectangle::calculateArea() {

return a \* b;

}

// Властивості

int Rectangle::getA() { return a; }

int Rectangle::getB() { return b; }

void Rectangle::setA(int s) { a = s; }

void Rectangle::setB(int s) { b = s; }

bool Rectangle::isSquare() { return a == b; }

**Main.cpp**

#include "Rectangle.h"

#include <iostream>

using namespace std;

int main() {

setlocale(0, "ukr");

// 1. Конструктор за замовчуванням

Rectangle r1;

r1.printSides();

// 2. Конструктор з параметрами

Rectangle r2(10, 10);

r2.printSides();

// 3. Конструктор копіювання

Rectangle r3 = r2;

r3.printSides();

Rectangle rectArr[3] = {

Rectangle(),

Rectangle(8, 2),

Rectangle(r2)

};

for (int i = 0; i < 3; i++) {

cout << "\nПрямокутник " << i + 1 << ":\n";

rectArr[i].printSides();

cout << "Периметр = " << rectArr[i].calculatePerimeter() << endl;

cout << "Площа = " << rectArr[i].calculateArea() << endl;

cout << (rectArr[i].isSquare() ? "Це квадрат" : "Не квадрат") << endl;

}

// 5. Покажчик на компонент-функцію

typedef void (Rectangle::\* FuncPointer)();

FuncPointer fp = &Rectangle::printSides;

// 6. Покажчик на об'єкт

Rectangle\* pRect = &r2;

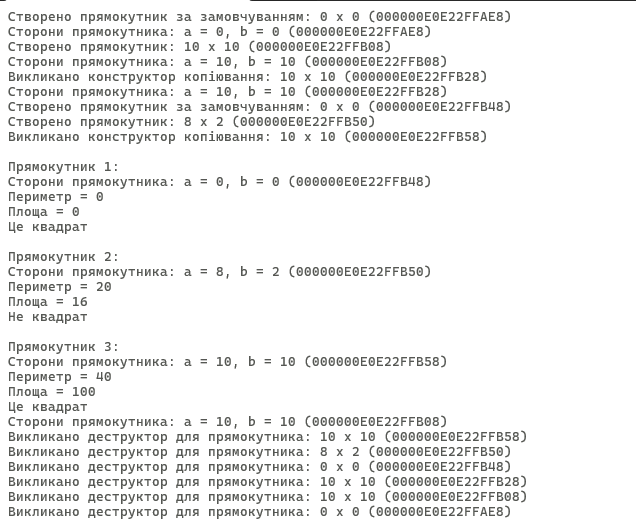
// 8. Виклик методу через обидва покажчики

(pRect->\*fp)();

return 0;

}

**Результат роботи програми:**

****

**Задача №3**

**Завдання**

1. Створіть додаток-форму Windows. Опишіть в модулі форми *під класом форми* базовий клас згідно варіанту, передбачивши в ньому не менше трьох полів, двох конструкторів різних видів, двох методів для виконання обчислень. Створіть також у цьому класі метод *Info* для виводу інформації.

**“Подорожуємо Україною”**

**Код програми:**

Class1.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LR1\_3

{

public class Podorozh

{

public string Misto { get; set; }

public double Vidstan { get; set; }

public int Dni { get; set; }

public string Transport { get; set; } // нове поле

public Podorozh()

{

Misto = "Львів";

Vidstan = 150;

Dni = 1;

Transport = "Автобус"; // за замовчуванням

}

public Podorozh(string misto, double vidstan, int dni, string transport)

{

Misto = misto;

Vidstan = vidstan;

Dni = dni;

Transport = transport;

}

public double RozrakhVartist()

{

double tsinaZaKm = 5;

double tsinaZaDen = 200;

return (Vidstan \* tsinaZaKm) + (Dni \* tsinaZaDen);

}

public double SerednyaTsinaKm()

{

return RozrakhVartist() / Vidstan;

}

public string Info()

{

return $"Місто: {Misto}, Відстань: {Vidstan} км, Днів: {Dni}, Транспорт: {Transport}, Вартість: {RozrakhVartist()} грн";

}

}

}

Form1.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LR1\_3

{

public partial class Form1 : Form

{

private List<Podorozh> podorozhi = new List<Podorozh>();

public Form1()

{

InitializeComponent();

dataGridView1.AutoGenerateColumns = false;

// Додаємо колонки

dataGridView1.Columns.Add("Misto", "Місто");

dataGridView1.Columns.Add("Vidstan", "Відстань (км)");

dataGridView1.Columns.Add("Dni", "Днів");

dataGridView1.Columns.Add("Transport", "Транспорт"); // нова колонка

dataGridView1.Columns.Add("Vartist", "Вартість");

dataGridView1.Columns.Add("Serednya", "Середня ціна/км");

dataGridView2.AutoGenerateColumns = false;

dataGridView2.Columns.Add("Misto", "Місто");

dataGridView2.Columns.Add("Vidstan", "Відстань (км)");

dataGridView2.Columns.Add("Dni", "Днів");

dataGridView2.Columns.Add("Transport", "Транспорт");

dataGridView2.Columns.Add("Vartist", "Вартість");

dataGridView2.Columns.Add("Serednya", "Середня ціна/км");

// Заповнюємо ComboBox транспортом

comboBox1.Items.AddRange(new string[] { "Автобус", "Поїзд", "Літак", "Авто" });

comboBox1.SelectedIndex = 0; // за замовчуванням "Автобус"

textBox1.KeyPress += TextBox1\_KeyPress;

textBox2.KeyPress += TextBox2\_KeyPress;

textBox3.KeyPress += TextBox3\_KeyPress;

}

// Місто — лише літери та пробіл

private void TextBox1\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!char.IsControl(e.KeyChar) && !char.IsLetter(e.KeyChar) && e.KeyChar != ' ')

{

e.Handled = true; // забороняємо введення

MessageBox.Show("Тільки літери та пробіли дозволені!");

}

}

// Відстань — лише цифри і крапка

private void TextBox2\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && e.KeyChar != '.')

{

e.Handled = true;

MessageBox.Show("Тільки цифри та крапка дозволені!");

}

// Не більше однієї крапки

if (e.KeyChar == '.' && (sender as TextBox).Text.Contains("."))

{

e.Handled = true;

}

}

// Дні — тільки цифри

private void TextBox3\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))

{

e.Handled = true;

MessageBox.Show("Тільки цифри дозволені!");

}

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

string misto = textBox1.Text;

double vidstan = double.Parse(textBox2.Text);

int dni = int.Parse(textBox3.Text);

string transport = comboBox1.SelectedItem.ToString(); // додаємо транспорт

Podorozh p = new Podorozh(misto, vidstan, dni, transport); // 4 аргументи

podorozhi.Add(p);

dataGridView1.Rows.Add(

p.Misto,

p.Vidstan,

p.Dni,

p.Transport, // нова колонка

p.RozrakhVartist(),

p.SerednyaTsinaKm().ToString("F2")

);

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

}

catch

{

MessageBox.Show("Будь ласка, введіть коректні дані!");

}

}

private void створитиToolStripMenuItem\_Click(object sender, EventArgs e)

{

podorozhi.Clear();

dataGridView1.Rows.Clear();

MessageBox.Show("Створено новий список подорожей!");

}

private void зберегтиToolStripMenuItem\_Click(object sender, EventArgs e)

{

SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.Filter = "Текстові файли|\*.txt";

saveFileDialog.Title = "Збереження даних";

if (saveFileDialog.ShowDialog() == DialogResult.OK)

{

using (System.IO.StreamWriter sw = new System.IO.StreamWriter(saveFileDialog.FileName))

{

foreach (var p in podorozhi)

{

sw.WriteLine($"{p.Misto};{p.Vidstan};{p.Dni};{p.Transport};{p.RozrakhVartist()};{p.SerednyaTsinaKm():F2}");

}

}

MessageBox.Show("Дані збережено!");

}

}

private void вивестиToolStripMenuItem\_Click(object sender, EventArgs e)

{

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "Текстові файли|\*.txt";

openFileDialog.Title = "Відкрити файл з даними";

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

podorozhi.Clear();

dataGridView1.Rows.Clear();

using (System.IO.StreamReader sr = new System.IO.StreamReader(openFileDialog.FileName))

{

string line;

while ((line = sr.ReadLine()) != null)

{

string[] parts = line.Split(';');

// Перевіряємо, що в рядку є хоча б 3 колонки (misto, vidstan, dni)

if (parts.Length >= 3)

{

string misto = parts[0];

double vidstan = double.Parse(parts[1]);

int dni = int.Parse(parts[2]);

// Якщо transport є у файлі, беремо його, інакше дефолт

string transport = parts.Length > 3 ? parts[3] : "Не вказано";

Podorozh p = new Podorozh(misto, vidstan, dni, transport);

podorozhi.Add(p);

// Додаємо в DataGridView, не забуваємо колонку Transport

dataGridView1.Rows.Add(

p.Misto,

p.Vidstan,

p.Dni,

p.Transport,

p.RozrakhVartist(),

p.SerednyaTsinaKm().ToString("F2")

);

}

}

}

MessageBox.Show("Дані завантажено!");

}

}

private void пошукToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form2 formSearch = new Form2(podorozhi);

if (formSearch.ShowDialog() == DialogResult.OK)

{

dataGridView2.Rows.Clear();

foreach (var p in formSearch.SearchResults)

{

dataGridView2.Rows.Add(

p.Misto,

p.Vidstan,

p.Dni,

p.Transport,

p.RozrakhVartist(),

p.SerednyaTsinaKm().ToString("F2")

);

}

}

}

}

}

Form2 searchForm

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;

namespace LR1\_3

{

public partial class Form2 : Form

{

private List<Podorozh> \_list;

public List<Podorozh> SearchResults { get; private set; }

public Form2(List<Podorozh> list)

{

InitializeComponent();

\_list = list;

// Заповнення combobox1 критеріями пошуку

comboBox1.Items.Clear();

comboBox1.Items.Add("Місто");

comboBox1.Items.Add("Транспорт");

comboBox1.Items.Add("Дні");

comboBox1.Items.Add("Місто + Транспорт + Дні"); // для комбінованого пошуку

comboBox1.SelectedIndex = 0; // за замовчуванням перший елемент

// Заповнення combobox2 транспортом

comboBox2.Items.Clear();

comboBox2.Items.Add("Будь-який"); // для всіх

foreach (var t in \_list.Select(p => p.Transport).Distinct())

{

comboBox2.Items.Add(t);

}

comboBox2.SelectedIndex = 0; // за замовчуванням “Будь-який”

}

private void buttonSearch\_Click(object sender, EventArgs e)

{

string crit = comboBox1.SelectedItem?.ToString(); // критерій пошуку

string mistoText = textBox1.Text.Trim();

string transportText = comboBox2.SelectedItem?.ToString();

string dniText = textBox2.Text.Trim();

var results = \_list.AsEnumerable();

if (!string.IsNullOrEmpty(crit))

{

switch (crit)

{

case "Місто":

if (string.IsNullOrEmpty(mistoText))

{

MessageBox.Show("Будь ласка, введіть місто для пошуку!");

return;

}

results = results.Where(p => p.Misto.IndexOf(mistoText, StringComparison.OrdinalIgnoreCase) >= 0);

break;

case "Транспорт":

if (string.IsNullOrEmpty(mistoText))

{

MessageBox.Show("Будь ласка, введіть транспорт для пошуку!");

return;

}

results = results.Where(p => p.Transport.IndexOf(mistoText, StringComparison.OrdinalIgnoreCase) >= 0);

break;

case "Дні":

if (string.IsNullOrEmpty(dniText))

{

MessageBox.Show("Будь ласка, введіть кількість днів для пошуку!");

return;

}

if (!int.TryParse(dniText, out int dni))

{

MessageBox.Show("Кількість днів повинна бути числом!");

return;

}

results = results.Where(p => p.Dni == dni);

break;

case "Місто + Транспорт + Дні":

if (string.IsNullOrEmpty(mistoText) || string.IsNullOrEmpty(transportText) || string.IsNullOrEmpty(dniText))

{

MessageBox.Show("Будь ласка, заповніть усі поля для комбінованого пошуку!");

return;

}

if (!int.TryParse(dniText, out int dniCombined))

{

MessageBox.Show("Кількість днів повинна бути числом!");

return;

}

results = results.Where(p =>

p.Misto.IndexOf(mistoText, StringComparison.OrdinalIgnoreCase) >= 0 &&

p.Transport.Equals(transportText, StringComparison.OrdinalIgnoreCase) &&

p.Dni == dniCombined);

break;

}

}

var finalResults = results.ToList();

if (finalResults.Count == 0)

{

MessageBox.Show("Нічого не знайдено!");

return;

}

double totalCost = finalResults.Sum(p => p.RozrakhVartist());

double avgDays = finalResults.Average(p => p.Dni);

double minDistance = finalResults.Min(p => p.Vidstan);

double maxDistance = finalResults.Max(p => p.Vidstan);

MessageBox.Show(

$"Знайдено {finalResults.Count} подорожей\n" +

$"Сумарна вартість: {totalCost}\n" +

$"Середня кількість днів: {avgDays:F1}\n" +

$"Відстань: від {minDistance} до {maxDistance} км"

);

SearchResults = finalResults;

this.DialogResult = DialogResult.OK;

this.Close();

}

}

}

**Результат програми:**

|  |  |
| --- | --- |
|  | Дані завантажено через вкладку “Вивести” з файлу. |
|  | Програма робить перевірки на введення даних в текст бокси  А також коли не усі дані записано виводить повідомлення що введіть коректні дані. |
|  | При натисканні на кнопку  “Створити” видається повідомлення і створюється нова таблиця |

|  |  |
| --- | --- |
|  | При натисканні на кнопку “Зберегти”, програма відкриває savefiledialog і Ви обираєте куда вам зберегти. Програма видає повідомлення що файл успішно збережено. |
|  | А це коли ми завантажуємо з файлу, програма видає повідомлення про успішність. |
|  | Спробуємо зробити пошук. Для цього натиснемо на вкладку “Пошук” і в нас відкриється інша форма для пошуку. Оберемо за чим шукати. Я візьму за містом. |
|  | Після знайденої інформації програма також робить додаткові розрахунки. |
|  | Програма працює досконало. |

**Висновок:**Під час виконання лабораторної роботи №1, я навчився працювати з класами та об’єктами на С++, а також у Windows Forms на С#.