**Лабораторна робота №1**

**Тема.** Класи і об’єкти в C# та C++

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

**Задача 1**

**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.

|  |
| --- |
| **3.КАДРИ** |
| ім'я – char\* |
| номер цеху-int |
| розряд- int |

Головна програма:

Cadr.h:

#pragma once

#include <iostream>

#include <string>

using namespace std;

class Cadr

{

private:

string name;

int number;

int rozr;

public:

Cadr();

Cadr(string n, int num, int r);

Cadr(const Cadr&);

~Cadr(void);

string getName();

void setName();

int getNum();

void setNum();

int getRozr();

void setRozr();

void SetCadr();

void PrintCadr();

};

Cadr.cpp:

#include "Cadr.h"

Cadr::Cadr() :name(""), number(0), rozr(0) { cout << "Використано конструктор за замовчуванням " << this<<endl; }

Cadr::Cadr(string n, int num, int r):

name(n),

number(num),

rozr(r)

{

cout << "Викликано звичайний конструктор "<< this << endl;

}

Cadr::Cadr(const Cadr& c) {

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

this->name = c.name;

this->number = c.number;

this->rozr = c.rozr;

}

Cadr::~Cadr(void) { cout << "Викликано деструктор" << this << endl; }

string Cadr::getName() {

return name;

}

void Cadr::setName() {

cout << "Введiть iм'я працівника: ";

cin >> name;

}

int Cadr::getNum() {

return number;

}

void Cadr::setNum() {

cout << "Введiть номер цеху: ";

cin >> number;

}

int Cadr::getRozr() {

return rozr;

}

void Cadr::setRozr() {

cout << "Введiть розряд: ";

cin >> rozr;

}

void Cadr::SetCadr() {

setName();

setNum();

setRozr();

}

void Cadr::PrintCadr() {

cout << "Iм'я: " << name << ", розряд: " << rozr << ", номер цеху: " << number << endl;

}

Program.cpp:

#include "Cadr.h"

int main() {

setlocale(0, "UKR");

Cadr cadr0; // Замовчування

Cadr cadr1("Name", 1, 1); // Класичний

Cadr cadr2 = cadr1; // Копіювання

cout << "Виведення\n";

cadr0.PrintCadr();

cadr1.PrintCadr();

cadr2.PrintCadr();

cout << "\nГетери / Сетери" << endl;

cadr0.setName();

cadr0.setNum();

cadr0.setRozr();

cout << "Заповнений:\nIм'я: " << cadr0.getName() << "\nНомер цеху: " << cadr0.getNum() << "\nРозряд: " << cadr0.getRozr() << endl;

cout << "\nМасив:" << endl;

Cadr\*array = new Cadr[5];

for (int i = 0; i < 5; i++)

array[i].SetCadr();

cout << "Заповнений масив:\n";

for (int i = 0; i < 5; i++)

array[i].PrintCadr();

delete[] array;

array = nullptr;

cout << "\nКомпонентний вказiвник" << endl;

void(Cadr:: \* pointer)();

pointer = &Cadr::PrintCadr;

(cadr1.\*pointer)();

cout << "\nВказiвник на екземпляр" << endl;

Cadr\* p = &cadr1;

p->PrintCadr();

}

Результат програми

|  |  |
| --- | --- |
| Програма | Тестовий набір |
|  | Cadr cadr0;  Cadr cadr1("Name", 1, 1),  Cadr cadr2 = cadr1;  cadr0.setName();  cadr0.setNum();  cadr0.setRozr();  Cadr\* array = new Cadr[5];  Cadr[0].SetCadr(),  Cadr[1].SetCadr(),  Cadr[2].SetCadr(),  Cadr[3].SetCadr(),  Cadr[4].SetCadr();  Cadr[0].PrintCadr(),  Cadr[1]. PrintCadr(),  Cadr[2]. PrintCadr(),  Cadr[3]. PrintCadr(),  Cadr[4]. PrintCadr();  void(Cadr:: \* pointer)();  pointer=&Cadr::PrintCadr;  (cadr1.\*pointer)();  Cadr\* p = &cadr1;  p->PrintCadr(); |

**Задача 2**

3. 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).

Головна програма:

Rectangle.h:

#pragma once

#include <iostream>

using namespace std;

class Rectangle

{

private:

double a, b;

public:

Rectangle();

Rectangle(double a, double b );

double P();

double S();

void setSizes();

void setA();

void setB();

void getSizes();

double getA();

double getB();

bool IsSquare();

void WhileNotNormal(double& num);

};

Rectangle.cpp:

#include "Rectangle.h"

void Rectangle::WhileNotNormal(double& num){

while (num <= 0) {

cout << "Невiрно задано сторону введiть знову: ";

cin >> num;

}

}

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

WhileNotNormal(a);

WhileNotNormal(b);

this->a = a;

this->b = b;

}

Rectangle::Rectangle() { a = 0; b = 0;}

double Rectangle::P() {

if (a == 0 || b == 0) {

cout << "Недостатньо значень сторiн для периметру" << endl;

return 0;

}

return 2\*(a + b);

}

double Rectangle::S() {

if (a == 0 || b == 0) {

cout << "Недостатньо значень сторiн для площi." << endl;

return 0;

}

return a \* b;

}

void Rectangle::setA() {

cout << "Введiть сторону a: ";

cin >> this->a;

WhileNotNormal(a);

}

void Rectangle::setB() {

cout << "Введiть сторону b: ";

WhileNotNormal(b);

this->b = b;

}

void Rectangle::setSizes() {

this->setA();

this->setB();

}

double Rectangle::getA() {

return a;

}

double Rectangle::getB() {

return b;

}

void Rectangle::getSizes() {

if (IsSquare())

cout << "Квадрат\n";

else

cout << "Прямокуник\n";

cout << "Довжина сторони a: " << getA() << ", дожина сторни b:" << getB() << endl;

}

bool Rectangle::IsSquare() {

if (a == 0 || b == 0) {

cout << "Недостатньо значень сторiн для визначення того чи квадрат." << endl;

return false;

}

return a == b;

}

Program.cpp:

#include "Rectangle.h"

int main() {

setlocale(0, "UKR");

cout << "Виклик конструктора за замовчуванням\n";

Rectangle rectangle;

cout << "Значення площi: " << rectangle.S() << " та перимету: " << rectangle.P() << " в конструкторi за замовчуванням" << endl;

cout << endl;

cout << "Виклик парамеризованого конструктора з помилкою:\n";

Rectangle rectangle1(10, -10);

cout << "Площа: " << rectangle1.S() << ", периметр: " << rectangle1.P() << endl;

cout << endl;

cout << "Використання гетерiв та сетерiв:\n";

Rectangle rectangle3;

rectangle3.setA();

rectangle3.setB();

cout << "GetA: " << rectangle3.getA() << ",GetB: " << rectangle3.getB() << endl;

cout << "GetSizes: \n"; rectangle3.getSizes();

cout << endl;

cout << "Робота з масивом:\n";

Rectangle\* rectangles = new Rectangle[5];

for (int i = 0; i < 5; i++) {

cout << i + 1 << " прямокутник: " << endl;

rectangles[i].setSizes();

}

for (int i = 0; i < 5; i++) {

cout << i + 1 << " прямокутник: " << endl;

rectangles[i].getSizes();

}

delete[] rectangles;

rectangles = nullptr;

cout << endl;

}

Результат програми:

|  |  |
| --- | --- |
| Програма | Тестовй набір |
|  | Rectangle rectangle;  Rectangle rectangle1(10, -10);  Rectangle rectangle3;  rectangle3.setA();  rectangle3.setB();  Rectangle\* rectangles = new Rectangle[5];  rectangles[0].setSizes();  rectangles[1].setSizes();  rectangles[2].setSizes();  rectangles[3].setSizes();  rectangles[4].setSizes();  rectangles[0].getSizes();  rectangles[1].getSizes();  rectangles[2].getSizes();  rectangles[3].getSizes();  rectangles[4].getSizes(); |

**Задача 3**

Створити додаток-форму Windows. Описати в модулі форми під класом форми базовий клас згідно варіанту:

Магазин побуової техніки

Передбачивши в ньому не менше трьох полів, двох конструкторів різних видів, двох методів для виконання обчислень. Створити також у цьому класі метод Info для виводу інформації. Створити у формі дві кнопки для створення об´єктів описаного класу, викликаючи, відповідно, конструктор по замовчуванню і конструктор з параметрами та метод Info створених об’єктів. В нижній частині форми створіть кнопку для завершення роботи та забезпечте її функціонування.

Головна програма:

Appliaces.cs:

using Microsoft.SqlServer.Server;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LabRob1

{

public class Appliaces

{

static int id = 1;

public string Id { get; private set; }

public string Name { get; set; }

public string Brand { get; set; }

public double Price { get; set; }

public short Year { get; set; }

public string EnergyClass { get; set; }

public short Power { get; set; }

public double EnergyCost { get; private set; }

public Appliaces(): this(id.ToString(),"Ім'я " + id, "Бренд " + id, 0, 0, "No",0) { }

public Appliaces(string Id, string Name, string Brand, double Price, short Year, string EnergyClass, short Power)

{

this.Id = Id;

this.Name = Name;

this.Brand = Brand;

this.Price = Price;

this.Year = Year;

this.EnergyClass = EnergyClass;

this.Power = Power;

id++;

CalculateCost();

}

// Вартість за годину роботи

private void CalculateCost()

{

double priceKW = 3.25;

EnergyCost = (double)Power \* priceKW /1000;

}

bool disc = false;

public void Discount()

{

double discount = 0;

switch (EnergyClass)

{

case "A+++": discount = 0.2;break;

case "A++": discount = 0.1;break;

case "A+": discount = 0.1;break;

}

if(disc == false)

{

disc = true;

Price -= Price \* discount;

}

}

}

}

Form1.cs:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LabRob1

{

public partial class Form1 : Form

{

DataTable dt = new DataTable();

DataTable dt1;

List<Appliaces> list;

string filter = "Binary file (\*.BIN)|\*.BIN";

CheckBox[] chk;

Control[] textBoxes;

public Form1()

{

InitializeComponent();

dt.Columns.Add("ID", typeof(string));

dt.Columns.Add("Ім'я", typeof(string));

dt.Columns.Add("Бренд", typeof(string));

dt.Columns.Add("Ціна", typeof(double));

dt.Columns.Add("Рік випуску", typeof(int));

dt.Columns.Add("Тип спожвання", typeof(string));

dt.Columns.Add("Потужність", typeof(int));

dt.Columns.Add("Витратність (на год.)", typeof(double));

dt1 = dt.Copy();

dataGridView1.DataSource = dt;

list = new List<Appliaces>();

saveFileDialog1.Filter = filter;

saveFileDialog1.Title = "Збереження в бінарний файл";

openFileDialog1.Filter = filter;

openFileDialog1.Title = "Відкриття бінарного файлу";

chk = new CheckBox[] { checkBox9, checkBox10, checkBox11, checkBox12, checkBox13, checkBox14, checkBox15 };

textBoxes = new Control[] { textBox19, textBox25, textBox24, textBox22, textBox21, comboBox13, textBox20 };

}

// Створення

private void toolStripButton5\_Click(object sender, EventArgs e)

{

if (IsEmpty(textBox10) || IsEmpty(textBox8) || IsEmpty(textBox9) || IsEmpty(textBox11) || IsEmpty(textBox12) || IsEmpty(textBox23) || IsEmpty(comboBox11))

return;

list.Add(new Appliaces(textBox10.Text, textBox8.Text, textBox9.Text, double.Parse(textBox11.Text), short.Parse(textBox12.Text), comboBox11.Text, short.Parse(textBox23.Text)));

}

// Введення в поле

private void comboBox11\_KeyPress(object sender, KeyPressEventArgs e)

{

e.Handled = true;

}

private void textBox11\_KeyPress(object sender, KeyPressEventArgs e)

{

if (char.IsDigit(e.KeyChar) || char.IsControl(e.KeyChar))

return;

if (e.KeyChar == ',' && !textBox11.Text.Contains(',') && textBox11.SelectionStart != 0)

return;

e.Handled = true;

}

private void textBox23\_KeyPress(object sender, KeyPressEventArgs e)

{

if (char.IsDigit(e.KeyChar) || char.IsControl(e.KeyChar))

return;

e.Handled = true;

}

private bool IsEmpty(Control t)

{

if (string.IsNullOrEmpty(t.Text))

{

t.Focus();

return true;

}

return false;

}

private void AddToTable(Appliaces a)

{

dt.Rows.Add(a.Id, a.Name, a.Brand, a.Price, a.Year, a.EnergyClass, a.Power, a.EnergyCost);

}

private void UpdateDataGridView()

{

dt.Clear();

foreach (Appliaces a in list)

AddToTable(a);

dataGridView1.DataSource = null;

dataGridView1.DataSource = dt;

}

private void toolStripButton6\_Click(object sender, EventArgs e)

{

foreach (DataGridViewRow row in dataGridView1.SelectedRows)

if (row.Index < list.Count)

list[row.Index].Discount();

UpdateDataGridView();

}

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

{

UpdateDataGridView();

}

private void Clear(params Control[] controls)

{

foreach (Control c in controls)

c.Text = "";

}

// Збереженн - читання

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

{

if (list.Count > 0 && saveFileDialog1.ShowDialog() == DialogResult.OK)

{

try

{

string path = saveFileDialog1.FileName;

using (FileStream fs = new FileStream(path, FileMode.Create))

using (BinaryWriter br = new BinaryWriter(fs))

{

foreach (Appliaces a in list)

{

br.Write(a.Id);

br.Write(a.Name);

br.Write(a.Brand);

br.Write(a.Price);

br.Write(a.Year);

br.Write(a.EnergyClass);

br.Write(a.Power);

}

}

}

catch { MessageBox.Show("Помилка запису в файл", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error); return; }

}

}

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

{

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

try

{

string path = openFileDialog1.FileName;

using (FileStream fs = new FileStream(path, FileMode.Open))

using (BinaryReader br = new BinaryReader(fs))

{

list.Clear();

while (br.BaseStream.Position < br.BaseStream.Length)

{

string id = br.ReadString();

string name = br.ReadString();

string brand = br.ReadString();

double price = br.ReadDouble();

short year = br.ReadInt16();

string type = br.ReadString();

short power = br.ReadInt16();

list.Add(new Appliaces(id, name, brand, price, year, type, power));

}

}

}

catch { MessageBox.Show("Помилка читання файлу.", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error); return; }

UpdateDataGridView();

}

}

// Виделення

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

int index = e.RowIndex;

if (index >= list.Count)

return;

textBox13.Text = list[index].Id;

textBox18.Text = list[index].Name;

textBox17.Text = list[index].Brand;

textBox16.Text = list[index].Price.ToString();

textBox15.Text = list[index].Year.ToString();

comboBox12.Text = list[index].EnergyClass;

textBox14.Text = list[index].Power.ToString();

}

private void toolStripButton1\_Click(object sender, EventArgs e)

{

List<Appliaces> delete = new List<Appliaces>();

foreach (DataGridViewRow row in dataGridView1.SelectedRows)

{

int index = row.Index;

if (index >= 0 && index < list.Count)

if (MessageBox.Show($"Видаляти {list[index].Name} {list[index].Brand} {list[index].Year}?", "Питання", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

delete.Add(list[index]);

}

foreach (Appliaces a in delete)

list.Remove(a);

Clear(textBox18, textBox17, textBox16, textBox15, textBox14, textBox13, comboBox12);

UpdateDataGridView();

}

private void tabControl2\_TabIndexChanged(object sender, EventArgs e)

{

TabControl tabControl = (TabControl)sender;

string text = tabControl.SelectedTab.Text;

if (text == "Пошук")

{

dataGridView2.Visible = true;

label39.Visible = true;

dataGridView1.Height = 230;

}

else

{

dataGridView1.Height = 470;

dataGridView2.Visible = false;

label39.Visible = false;

}

}

private void toolStripButton2\_Click(object sender, EventArgs e)

{

bool any = false;

foreach (CheckBox c in chk)

any |= c.Checked;

if (!any)

return;

dt1.Rows.Clear();

foreach (DataRow row in dt.Rows)

{

bool find = true;

for (int i = 0; i < chk.Length; i++)

{

if (chk[i].Checked)

if (textBoxes[i].Text != row[i].ToString())

{

find = false;

break;

}

}

if (find)

dt1.Rows.Add(row.ItemArray);

}

if(dt1.Rows.Count == 0)

{

MessageBox.Show("Елементів не знайдено.", "Повідомлення", MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

dataGridView2.DataSource = dt1;

}

private void toolStripButton7\_Click(object sender, EventArgs e)

{

list.Add(new Appliaces());

}

}

}

Тествовй набір:

|  |  |  |
| --- | --- | --- |
| № | Програма | Тестовий набір |
| 1 |  | Створено п’ять об’єків та виведено їх  На екран  list.Add(new Appliaces(textBox10.Text, textBox8.Text, textBox9.Text, double.Parse(textBox11.Text), short.Parse(textBox12.Text), comboBox11.Text, short.Parse(textBox23.Text)));  Додано ще п’яь об’єктів та виведено їх  list.Add(new Appliaces(textBox10.Text, textBox8.Text, textBox9.Text, double.Parse(textBox11.Text), short.Parse(textBox12.Text), comboBox11.Text, short.Parse(textBox23.Text)));  Додано об’єкти за замовчуванням.  list.Add(new Appliaces());  Визначено знижку для товарів.  list[i].Discout();  Підтвердження видалення об’єкта.  DialogResult = Yes  Видалено об’єкт  List.Remove(delete[i])  Відкрито бінарний файл для перенесення в DataGridView  OpenFileDialog.ShowDialog() == DialogResult.OK  Елемент не знайдено.  Dt1.Rows.Count = 0  Пошук за декількома критеріями.  Dt1.Rows.Count = 2  Пошук за одним критерієм.  Dt1.Rows.Count = 5 |

Висновок: засвоєно основи роботи із властивостями, конструктором за замовчуванням, конструктором копіювання, параметризованим конструктором, деструктором, компонентним вказівником, ключовим словом this та засвоєно роботу із модифікаторами доступу в проектах із графічним інтерфейсом C# Windows Form та консольними проектами C++.