**Лабораторна робота №1**

## Тема: Класи й об'єкти в С++ та C#.

***Мета роботи:*** Одержати практичні навички реалізації класів на С++ та C#.

**Варіант 24**

**Задача 1(8)**

**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.

**Умова:**

Зображення, що містить текст, Шрифт, знімок екрана, ряд

Вміст на основі ШІ може бути неправильним.

**Код:**

1)Опис класу, Product.h

#pragma once

#include <string>

using namespace std;

class Product

{

public:

string name;

int amount;

float value;

public:

Product();

Product(const string& name, int amount, float value);

Product(const Product& other);

~Product(void);

string getName();

int getAmount();

float getValue();

void setProduct(const string& n, int a, float v);

void setName(const string& n);

void setAmount(int a);

void setValue(float v);

void printProduct();

};

2)Product.cpp

#include "Product.h"

#include <iostream>

#include <iomanip>

#include <algorithm>

using namespace std;

Product::Product() : name(""), amount(0), value(0.0f) {}

Product::Product(const string& name, int amount, float value) {

setProduct(name, amount, value);

}

Product::Product(const Product& other)

: name(other.name), amount(other.amount), value(other.value) {

}

Product::~Product(void) {}

string Product::getName() { return name; }

int Product::getAmount() { return amount; }

float Product::getValue() { return value; }

void Product::setProduct(const string& n, int a, float v) {

setName(n);

setAmount(a);

setValue(v);

}

void Product::setName(const string& n) { name = n; }

void Product::setAmount(int a) {

amount = max(0, a);

}

void Product::setValue(float v) {

value = (v < 0.0f) ? 0.0f : v;

}

void Product::printProduct() {

cout << fixed << setprecision(2);

cout << "Product: " << name

<< ", amount: " << amount

<< ", value: " << value

<< ", total: " << (amount \* value)

<< endl;

}

3)main.cpp

#include "Product.h"

int main() {

Product p1;

p1.setProduct("USB Cable", 5, 12.50f);

p1.printProduct();

Product p2("Keyboard", 2, 399.99f);

p2.printProduct();

Product p3 = p2;

p3.setAmount(3);

p3.setValue(379.99f);

p3.printProduct();

Product bad("Test", -10, -5.0f);

bad.printProduct();

}

РЕЗУЛЬТАТ ПРОГРАМИ

Зображення, що містить текст, Шрифт, знімок екрана, чорний

Вміст на основі ШІ може бути неправильним.

**Задача 2(4)**

**Умова:**Створити клас Money, розробивши наступні елементи класу:

* + Поля:
    - int first;//номинал купюри
    - int second; //кількість купюр
  + Конструктор, що дозволяє створити екземпляр класу із заданими значенням полів.
  + Методи, що дозволяють:
    - вивести номінал і кількість купюр;
    - визначити, чи вистачить грошових коштів на покупку товару на суму N гривнів.
    - визначити, скільки шт товару вартості n гривнів можна купити на наявні грошові кошти.
  + Властивості:
    - які надають можливість одержати-встановити значення полів (доступне для читання і запису);
    - які надають можливість расчитатать суму грошей (доступне тільки для читання).

**Код:**1)Опис класу, Money.h  
#pragma once

#include <iostream>

using namespace std;

namespace AppConstants {

static const int CoinsArr[4] = { 1, 2, 5, 10 };

static const int BillsArr[6] = { 20, 50, 100, 200, 500, 1000 };

}

class Money {

private:

int nominal;

int count;

bool isCoin;

bool isValidCoin(int n);

bool isValidBill(int n);

void setNominal(int n);

void setCount(int c);

public:

Money();

void input();

void print();

int getNominal();

int getCount();

bool getIsCoin();

long long amount();

};

long long totalAmount(Money arr[], int n);

bool canBuy(Money arr[], int n, long long sum);

long long howManyItems(Money arr[], int n, long long price);

2) InputUtils.h

#pragma once

#include <string>

using namespace std;

bool isNumber(const string& s);

int readIntInRange(const string& prompt, int minVal, int maxVal);

long long readPositiveLL(const string& prompt);

#pragma once

3) InputUtils.cpp

#include "InputUtils.h"

#include <iostream>

using namespace std;

bool isNumber(const string& s) {

if (s.empty()) return false;

for (char c : s) {

if (!isdigit(c)) return false;

}

return true;

}

int readIntInRange(const string& prompt, int minVal, int maxVal) {

string input;

int value;

do {

cout << prompt;

cin >> input;

if (isNumber(input)) {

value = stoi(input);

}

else {

cout << "Введіть ціле число!\n";

value = minVal - 1;

}

if (value < minVal || value > maxVal) {

cout << "Значення має бути в діапазоні від "

<< minVal << " до " << maxVal << ".\n";

}

} while (value < minVal || value > maxVal);

return value;

}

long long readPositiveLL(const string& prompt) {

string input;

long long value;

do {

cout << prompt;

cin >> input;

if (isNumber(input)) {

value = stoll(input);

}

else {

cout << "Введіть число\n";

value = 0;

}

if (value <= 0) {

cout << "Значення має бути більше 0\n";

}

} while (value <= 0);

return value;

}

4)Money.cpp

#include "Money.h"

#include "InputUtils.h"

bool Money::isValidCoin(int n) {

for (int i = 0; i < 4; ++i) {

if (AppConstants::CoinsArr[i] == n) return true;

}

return false;

}

bool Money::isValidBill(int n) {

for (int i = 0; i < 6; ++i) {

if (AppConstants::BillsArr[i] == n) return true;

}

return false;

}

Money::Money() {

nominal = 0;

count = 0;

isCoin = true;

}

void Money::setNominal(int n) {

nominal = n;

}

void Money::setCount(int c) {

if (c < 0) c = 0;

count = c;

}

void Money::input() {

char t;

do {

cout << "Це монета (m) чи купюра (b)? ";

cin >> t;

} while (!(t == 'm' || t == 'M' || t == 'b' || t == 'B'));

isCoin = (t == 'm' || t == 'M');

cout << "Дозволені номінали: ";

if (isCoin) {

for (int v : AppConstants::CoinsArr) cout << v << " ";

}

else {

for (int v : AppConstants::BillsArr) cout << v << " ";

}

cout << "\n";

int n;

bool ok = false;

do {

n = (int)readPositiveLL("Введіть номінал: ");

ok = isCoin ? isValidCoin(n) : isValidBill(n);

if (!ok) {

cout << "Недопустимий номінал\n";

}

} while (!ok);

setNominal(n);

int c;

do {

c = (int)readPositiveLL("Введіть кількість: ");

if (c <= 0) {

cout << "Кількість має бути більше 0\n";

}

} while (c <= 0);

setCount(c);

}

void Money::print() {

cout << (isCoin ? "Монета " : "Купюра ")

<< nominal << " грн × " << count

<< " = " << amount() << " грн\n";

}

int Money::getNominal() { return nominal; }

int Money::getCount() { return count; }

bool Money::getIsCoin() { return isCoin; }

long long Money::amount() {

return 1LL \* nominal \* count;

}

long long totalAmount(Money arr[], int n) {

long long s = 0;

for (int i = 0; i < n; ++i) s += arr[i].amount();

return s;

}

bool canBuy(Money arr[], int n, long long sum) {

return totalAmount(arr, n) >= sum;

}

long long howManyItems(Money arr[], int n, long long price) {

if (price <= 0) return 0;

return totalAmount(arr, n) / price;

}

5) Main.cpp

#include "Money.h"

#include "InputUtils.h"

#include <iostream>

using namespace std;

int main() {

setlocale(0, "ukr");

int n = readIntInRange("Скільки елементів у гаманці? ", 1, 20);

Money wallet[20];

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

cout << "\nЕлемент #" << i + 1 << ":\n";

wallet[i].input();

}

cout << "\nВаш набір грошей:\n";

for (int i = 0; i < n; ++i) wallet[i].print();

long long N = readPositiveLL("\nСума покупки: ");

cout << (canBuy(wallet, n, N) ? "Грошей вистачить.\n" : "Грошей не вистачить.\n");

long long p = readPositiveLL("Ціна товару: ");

cout << "Максимум можна купити: " << howManyItems(wallet, n, p) << " шт\n";

cout << "Загальна сума: " << totalAmount(wallet, n) << " грн\n";

}

РЕЗУЛЬТАТ ПРОГРАМИ

Зображення, що містить текст, знімок екрана, Шрифт

Вміст на основі ШІ може бути неправильним.

**Windows form Application  
Тема : «Рієлторська кантора»**Зображення, що містить текст, знімок екрана, програмне забезпечення, монітор

Вміст на основі ШІ може бути неправильним.

**Об`єкти на формі  
Зображення, що містить текст, знімок екрана, програмне забезпечення, монітор

Вміст на основі ШІ може бути неправильним.**

**Код:  
Опис класу**using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace рієлторська\_контора\_приклад\_гуменна

{

[Serializable]

public class PropertyItem

{

public Form1.PropertyType Type { get; set; }

public string Address { get; set; }

public decimal Area { get; set; }

public int Rooms { get; set; }

public int YearBuilt { get; set; }

public decimal Price { get; set; }

public Form1.DealStatus Status { get; set; }

public Form1.Currency Currency { get; set; }

public PropertyItem() { }

public PropertyItem(

Form1.PropertyType type,

string address,

decimal area,

int rooms,

int yearBuilt,

decimal price,

Form1.DealStatus status,

Form1.Currency currency)

{

Type = type;

Address = address;

Area = area;

Rooms = rooms;

YearBuilt = yearBuilt;

Price = price;

Status = status;

Currency = currency;

}

}

}

**Код форми**using System;

using System.Collections.Generic;

using System.Drawing;

using System.IO;

using System.Text;

using System.Text.RegularExpressions;

using System.Windows.Forms;

namespace рієлторська\_контора\_приклад\_гуменна

{

public partial class Form1 : Form

{

public enum PropertyType { Квартира, Будинок, Офіс, Ділянка }

public enum DealStatus { В\_продажу, Оренда, Продано, Резерв }

public enum Currency { UAH, USD, EUR }

public enum PropertyTypeFilter { Всі, Квартира, Будинок, Офіс, Ділянка }

public enum DealStatusFilter { Всі, В\_продажу, Оренда, Продано, Резерв }

private List<PropertyItem> properties = new List<PropertyItem>();

private string lastSortColumn = "";

private bool lastSortAsc = true;

public Form1()

{

InitializeComponent();

mnuEdit.Click += mnuEdit\_Click;

mnuDelete.Click += mnuDelete\_Click;

dataGridView1.SelectionChanged += (s, e) => UpdateMenuState();

this.Load += FormMain\_Load;

txtStreetName.Validating += (s, e) => ValidateStreetName();

mtxHouse.Validating += (s, e) => ValidateHouse();

txtSearch.TextChanged += (s, e) => ApplyView();

cmbFilterType.SelectedIndexChanged += (s, e) => ApplyView();

cmbFilterStatus.SelectedIndexChanged += (s, e) => ApplyView();

btnResetFilters.Click += (s, e) =>

{

txtSearch.Clear();

if (cmbFilterType.Items.Count > 0) cmbFilterType.SelectedIndex = 0;

if (cmbFilterStatus.Items.Count > 0) cmbFilterStatus.SelectedIndex = 0;

ApplyView();

};

dataGridView1.ColumnHeaderMouseClick += dataGridView1\_ColumnHeaderMouseClick;

}

private void FormMain\_Load(object sender, EventArgs e)

{

ConfigureGrid();

mtxHouse.Mask = "000>LL/000";

mtxHouse.PromptChar = ' ';

if (cmbStreetType.Items.Count == 0)

{

cmbStreetType.DropDownStyle = ComboBoxStyle.DropDownList;

cmbStreetType.Items.AddRange(new object[]

{ "вул.", "просп.", "пров.", "бульв.", "пл.", "шосе", "набережна", "узвіз", "тракт", "кв-л", "мікрорайон" });

cmbStreetType.SelectedIndex = 0;

}

cmbType.DropDownStyle = ComboBoxStyle.DropDownList;

cmbType.DataSource = Enum.GetValues(typeof(PropertyType));

cmbCurrency.DropDownStyle = ComboBoxStyle.DropDownList;

cmbCurrency.DataSource = Enum.GetValues(typeof(Currency));

cmbStatus.DropDownStyle = ComboBoxStyle.DropDownList;

cmbStatus.DataSource = new[]

{

new { Key = DealStatus.В\_продажу, Value = "В продажу" },

new { Key = DealStatus.Оренда, Value = "Оренда" },

new { Key = DealStatus.Продано, Value = "Продано" },

new { Key = DealStatus.Резерв, Value = "Резерв" },

};

cmbStatus.DisplayMember = "Value";

cmbStatus.ValueMember = "Key";

cmbFilterType.DropDownStyle = ComboBoxStyle.DropDownList;

cmbFilterType.DataSource = Enum.GetValues(typeof(PropertyTypeFilter));

cmbFilterStatus.DropDownStyle = ComboBoxStyle.DropDownList;

cmbFilterStatus.DataSource = Enum.GetValues(typeof(DealStatusFilter));

ApplyView();

}

private void ConfigureGrid()

{

var dgv = dataGridView1;

dgv.AllowUserToAddRows = false;

dgv.ReadOnly = true;

dgv.RowHeadersVisible = false;

dgv.AutoGenerateColumns = false;

dgv.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.DisplayedCells;

dgv.BackgroundColor = Color.White;

dgv.Columns.Clear();

dgv.Columns.Add(new DataGridViewTextBoxColumn { Name = "colType", HeaderText = "Тип", Width = 100 });

dgv.Columns.Add(new DataGridViewTextBoxColumn { Name = "colAddress", HeaderText = "Адреса", Width = 260, DefaultCellStyle = { WrapMode = DataGridViewTriState.True } });

dgv.Columns.Add(new DataGridViewTextBoxColumn { Name = "colArea", HeaderText = "Площа", Width = 80, DefaultCellStyle = { Alignment = DataGridViewContentAlignment.MiddleRight, Format = "N2" } });

dgv.Columns.Add(new DataGridViewTextBoxColumn { Name = "colRooms", HeaderText = "Кімнат", Width = 70, DefaultCellStyle = { Alignment = DataGridViewContentAlignment.MiddleCenter, Format = "N0" } });

dgv.Columns.Add(new DataGridViewTextBoxColumn { Name = "colYear", HeaderText = "Рік", Width = 70, DefaultCellStyle = { Alignment = DataGridViewContentAlignment.MiddleCenter, Format = "N0" } });

dgv.Columns.Add(new DataGridViewTextBoxColumn { Name = "colPrice", HeaderText = "Ціна", Width = 100, DefaultCellStyle = { Alignment = DataGridViewContentAlignment.MiddleRight, Format = "N2" } });

dgv.Columns.Add(new DataGridViewTextBoxColumn { Name = "colCurrency", HeaderText = "Валюта", Width = 70, DefaultCellStyle = { Alignment = DataGridViewContentAlignment.MiddleCenter } });

dgv.Columns.Add(new DataGridViewTextBoxColumn { Name = "colStatus", HeaderText = "Статус", Width = 110 });

}

private bool ValidateStreetName()

{

string text = txtStreetName.Text.Trim();

bool ok = Regex.IsMatch(text, @"^[\p{L}][\p{L}\-’'\s]{0,}$", RegexOptions.CultureInvariant);

if (!ok)

errorProvider1.SetError(txtStreetName, "Тільки літери, пробіли, дефіси, апостроф.");

else

errorProvider1.SetError(txtStreetName, "");

return ok;

}

private bool ValidateHouse()

{

string raw = mtxHouse.Text.Trim();

bool ok = !string.IsNullOrWhiteSpace(raw);

if (!ok)

errorProvider1.SetError(mtxHouse, "Вкажіть номер (напр. 12, 12А, 12/3).");

else

errorProvider1.SetError(mtxHouse, "");

return ok;

}

private string BuildAddress()

{

string type = (cmbStreetType.Text ?? "").Trim();

string name = Regex.Replace((txtStreetName.Text ?? "").Trim(), @"\s+", " ");

string house = (mtxHouse.Text ?? "").Trim();

return type + " " + name + ", " + house;

}

private void ClearInputs()

{

txtStreetName.Clear();

mtxHouse.Clear();

numArea.Value = 0;

numRooms.Value = 0;

numYear.Value = DateTime.Now.Year;

numPrice.Value = 0;

errorProvider1.SetError(txtStreetName, "");

errorProvider1.SetError(mtxHouse, "");

txtStreetName.Focus();

}

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

{

if (!ValidateStreetName() | !ValidateHouse())

{

MessageBox.Show("Перевірте коректність адреси.", "Увага");

return;

}

var item = new PropertyItem

{

Type = (PropertyType)cmbType.SelectedItem,

Currency = (Currency)cmbCurrency.SelectedItem,

Status = (DealStatus)cmbStatus.SelectedValue,

Address = BuildAddress(),

Area = numArea.Value,

Rooms = (int)numRooms.Value,

YearBuilt = (int)numYear.Value,

Price = numPrice.Value

};

properties.Add(item);

ApplyView();

ClearInputs();

}

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

{

ClearInputs();

}

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

{

ClearInputs();

if (cmbStreetType.Items.Count > 0) cmbStreetType.SelectedIndex = 0;

if (cmbType.Items.Count > 0) cmbType.SelectedIndex = 0;

if (cmbStatus.Items.Count > 0) cmbStatus.SelectedIndex = 0;

if (cmbCurrency.Items.Count > 0) cmbCurrency.SelectedIndex = 0;

if (cmbFilterType.Items.Count > 0) cmbFilterType.SelectedIndex = 0;

if (cmbFilterStatus.Items.Count > 0) cmbFilterStatus.SelectedIndex = 0;

}

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

{

if (properties.Count == 0)

{

MessageBox.Show("Немає даних для збереження.");

return;

}

SaveFileDialog sfd = new SaveFileDialog();

sfd.Filter = "Binary (\*.bin)|\*.bin";

sfd.FileName = "properties.bin";

if (sfd.ShowDialog() != DialogResult.OK) return;

SaveToBin(sfd.FileName);

MessageBox.Show("Дані збережено.", "OK");

}

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

{

OpenFileDialog ofd = new OpenFileDialog();

ofd.Filter = "Binary (\*.bin)|\*.bin";

if (ofd.ShowDialog() != DialogResult.OK) return;

LoadFromBin(ofd.FileName);

MessageBox.Show("Дані завантажено.", "OK");

ApplyView();

}

private void SaveToBin(string path)

{

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

using (BinaryWriter bw = new BinaryWriter(fs, Encoding.UTF8))

{

bw.Write(properties.Count);

foreach (PropertyItem it in properties)

{

bw.Write((int)it.Type);

bw.Write(it.Address ?? "");

bw.Write(it.Area);

bw.Write(it.Rooms);

bw.Write(it.YearBuilt);

bw.Write(it.Price);

bw.Write((int)it.Currency);

bw.Write((int)it.Status);

}

}

}

private void LoadFromBin(string path)

{

properties.Clear();

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

using (BinaryReader br = new BinaryReader(fs, Encoding.UTF8))

{

int count = br.ReadInt32();

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

{

PropertyItem it = new PropertyItem();

it.Type = (PropertyType)br.ReadInt32();

it.Address = br.ReadString();

it.Area = br.ReadDecimal();

it.Rooms = br.ReadInt32();

it.YearBuilt = br.ReadInt32();

it.Price = br.ReadDecimal();

it.Currency = (Currency)br.ReadInt32();

it.Status = (DealStatus)br.ReadInt32();

properties.Add(it);

}

}

}

private void ApplyView()

{

string searchText = (txtSearch.Text == null) ? "" : txtSearch.Text.Trim().ToLower();

PropertyTypeFilter typeFilter = PropertyTypeFilter.Всі;

DealStatusFilter statusFilter = DealStatusFilter.Всі;

if (cmbFilterType.SelectedItem != null)

typeFilter = (PropertyTypeFilter)cmbFilterType.SelectedItem;

if (cmbFilterStatus.SelectedItem != null)

statusFilter = (DealStatusFilter)cmbFilterStatus.SelectedItem;

dataGridView1.Rows.Clear();

foreach (PropertyItem it in properties)

{

bool ok = true;

if (searchText.Length > 0)

{

string addr = (it.Address ?? "").ToLower();

if (!addr.Contains(searchText))

ok = false;

}

if (ok && typeFilter != PropertyTypeFilter.Всі)

{

if (it.Type.ToString() != typeFilter.ToString())

ok = false;

}

if (ok && statusFilter != DealStatusFilter.Всі)

{

if (it.Status.ToString() != statusFilter.ToString())

ok = false;

}

if (ok)

{

dataGridView1.Rows.Add(

it.Type.ToString(),

it.Address,

it.Area,

it.Rooms,

it.YearBuilt,

it.Price,

it.Currency.ToString(),

it.Status.ToString()

);

}

}

}

private void dataGridView1\_ColumnHeaderMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

string col = dataGridView1.Columns[e.ColumnIndex].Name;

if (lastSortColumn == col)

lastSortAsc = !lastSortAsc;

else

{

lastSortColumn = col;

lastSortAsc = true;

}

Comparison<PropertyItem> cmp = (a, b) =>

{

int r = 0;

switch (col)

{

case "colType": r = string.Compare(a.Type.ToString(), b.Type.ToString(), StringComparison.OrdinalIgnoreCase); break;

case "colAddress": r = string.Compare(a.Address, b.Address, StringComparison.OrdinalIgnoreCase); break;

case "colArea": r = a.Area.CompareTo(b.Area); break;

case "colRooms": r = a.Rooms.CompareTo(b.Rooms); break;

case "colYear": r = a.YearBuilt.CompareTo(b.YearBuilt); break;

case "colPrice": r = a.Price.CompareTo(b.Price); break;

case "colCurrency": r = string.Compare(a.Currency.ToString(), b.Currency.ToString(), StringComparison.OrdinalIgnoreCase); break;

case "colStatus": r = string.Compare(a.Status.ToString(), b.Status.ToString(), StringComparison.OrdinalIgnoreCase); break;

}

if (!lastSortAsc) r = -r;

return r;

};

properties.Sort(cmp);

ApplyView();

}

private void UpdateMenuState()

{

bool hasSelection = dataGridView1.SelectedRows.Count > 0;

mnuEdit.Enabled = hasSelection && dataGridView1.SelectedRows.Count == 1;

mnuDelete.Enabled = hasSelection;

}

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

{

if (dataGridView1.SelectedRows.Count != 1) return;

var row = dataGridView1.SelectedRows[0];

var item = properties[row.Index];

cmbType.SelectedItem = item.Type;

txtStreetName.Text = item.Address;

numArea.Value = item.Area;

numRooms.Value = item.Rooms;

numYear.Value = item.YearBuilt;

numPrice.Value = item.Price;

cmbCurrency.SelectedItem = item.Currency;

cmbStatus.SelectedItem = item.Status;

btnAdd.Text = "Зберегти";

}

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

{

if (dataGridView1.SelectedRows.Count == 0) return;

if (MessageBox.Show("Видалити вибрані записи?", "Підтвердження",

MessageBoxButtons.YesNo, MessageBoxIcon.Question) != DialogResult.Yes) return;

foreach (DataGridViewRow row in dataGridView1.SelectedRows)

{

if (row.Index >= 0 && row.Index < properties.Count)

properties.RemoveAt(row.Index);

}

ApplyView();

}

}

}  
  
**Робота програми  
Виводим список з бінарного файлу**Зображення, що містить текст, знімок екрана, програмне забезпечення, Мультимедійне програмне забезпечення

Вміст на основі ШІ може бути неправильним. **Результат виводу  
Зображення, що містить текст, знімок екрана, програмне забезпечення, Комп’ютерна піктограма

Вміст на основі ШІ може бути неправильним.  
Додаєм новий обєкт  
Зображення, що містить текст, знімок екрана, програмне забезпечення, Комп’ютерна піктограма

Вміст на основі ШІ може бути неправильним.  
Результат додавання  
  
Пошук  
Зображення, що містить текст, знімок екрана, програмне забезпечення, Комп’ютерна піктограма

Вміст на основі ШІ може бути неправильним.  
  
Фільтрація  
Зображення, що містить знімок екрана, текст, програмне забезпечення, Комп’ютерна піктограма

Вміст на основі ШІ може бути неправильним.**



**Редагування  
Зображення, що містить текст, знімок екрана, програмне забезпечення, Комп’ютерна піктограма

Вміст на основі ШІ може бути неправильним.**

**Видалення**

**Видалення**

**Зображення, що містить текст, знімок екрана, програмне забезпечення, Комп’ютерна піктограма

Вміст на основі ШІ може бути неправильним.**

**Зображення, що містить текст, знімок екрана, програмне забезпечення, Комп’ютерна піктограма

Вміст на основі ШІ може бути неправильним.**

**Зображення, що містить текст, знімок екрана, програмне забезпечення, Комп’ютерна піктограма

Вміст на основі ШІ може бути неправильним.  
  
Висновок**: я навчилась працювати з класами на мовах с++ та с#.