Проектування та розробка Windows форм з MDI інтерфейсом в Visual Studio C#.

Мета**:** Засвоїти методику та виробити практичні навички у проектуванні та створенні форм з MDI інтерфейсом за допомогою Visual C# 2010.

**Теоретичні відомості.**

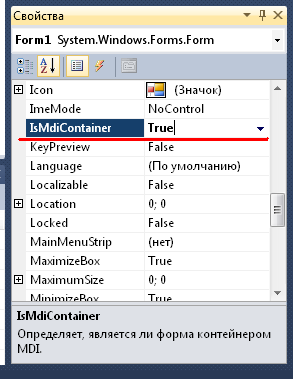
Програми з багато віконним інтерфейсом (MDI) дозволяють відображати кілька документів відразу, при цьому кожен документ відображається в окремому вікні.

Цей інтерфейс був вперше описаний в керівництві з розробки інтерфейсу користувача System Application Architecture Common User Access Advanced Interface Design Guide(SAA / CUA), створеному IBM. Інтерфейс MDI використовувався в ОС Microsoft Windows, починаючи з версії 3.0, а також в графічній оболонці Presentation Manager операційної системи OS/2. Сучасні версії ОС Microsoft Windows також дозволяють створювати додатки з інтерфейсом MDI.

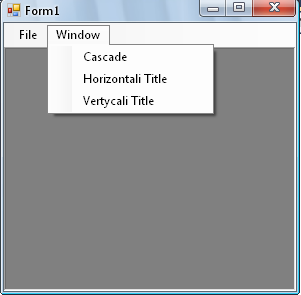
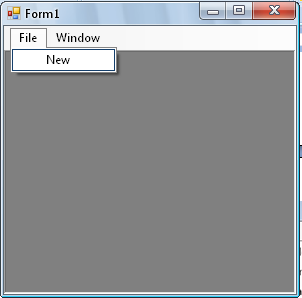
Для створення MDI-форм в проекті потрібно мати як мінімум дві форми які називаються батьківською та дочірньою.

Будь MDI-додаток містить меню Windows, призначене для управління вікнами, що відображають різні документи або різні подання одного й того ж документа. Як правило, в меню Windows є команди, за допомогою яких користувач може впорядкувати MDI-вікна, розташувавши їх з перекриттям (один за одним) або поруч один з одним. У цьому меню можуть бути й інші рядки, що керують розташуванням MDI-вікон. Рухаючи MDI-вікна за допомогою заголовка, Ви не зможете перемістити їх за межі головного вікна програми.

Основою програми з MDI-інтерфейсом є батьківська MDI-форма. Це форма містить дочірні MDI-вікна, що є "вкладеними вікнами", в яких користувачі взаємодіють з MDI-додатком. Створити батьківську MDI-форму нескладно як в конструкторі *Windows Forms*, так і програмними засобами. У вікні Властивостей установіть властивості *IsMDIContainer* значення *true.* При цьому форма призначається MDI-контейнером для дочірніх вікон.



Далі,потрібно додати в програму головне меню. Для цього перетягують з панелі елементів системи Microsoft Visual Studio. NET піктограму головного меню *MainMenu*.



За допомогою рядка меню New  створюються нові MDI-вікна з редактором тексту на базі елемента керування RichTextBox.

Дочірні MDI-форми є основним елементом програми з інтерфейсом MDI, оскільки вони знаходяться в центрі взаємодії з користувачами. За допомогою такої процедури створюються дочірні MDI-форми, що відображають елементи керування, аналогічно більшості текстових додатків.

Потрібно додати в проект програми нову форму, яка буде відігравати роль шаблону для створення дочірніх MDI-вікон.

Щоб створити нову дочірню MDI-форму використовують код.

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

{

Form2 mdiChild = new Form2();

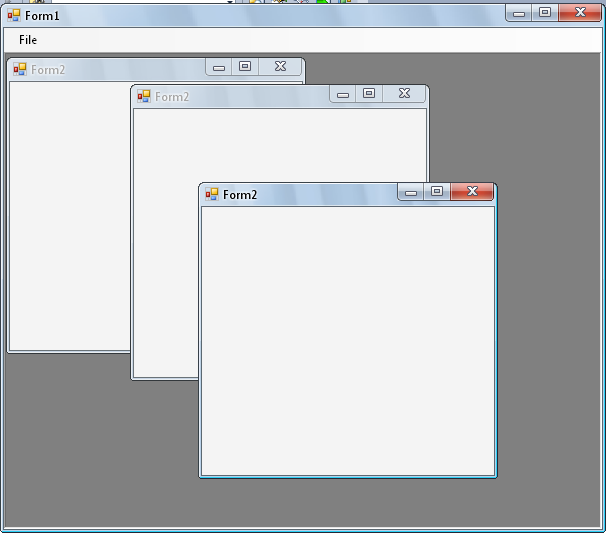
mdiChild.MdiParent = this;

mdiChild.Show();

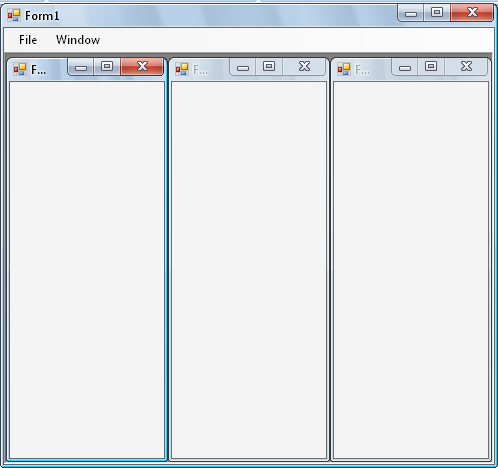
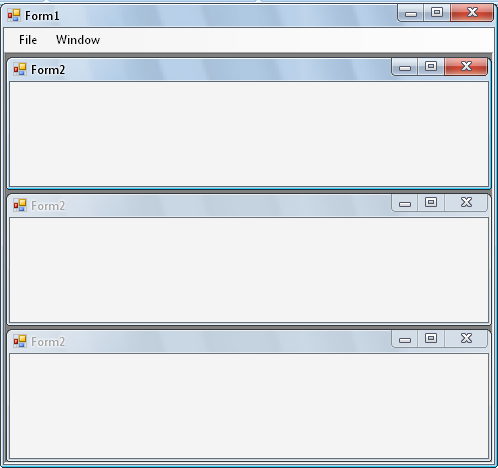
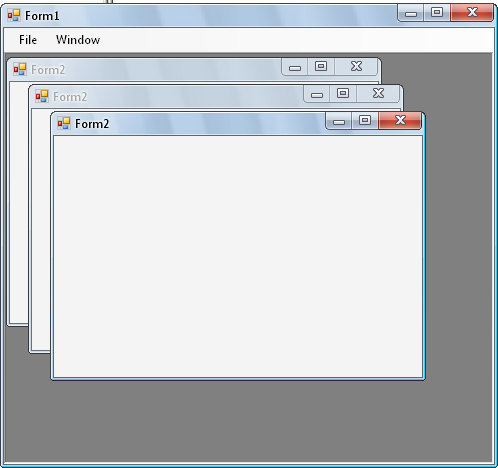
}

Тут ми спочатку створюємо нову форму як об'єкт класу Form2, а потім зберігаємо посилання на цю форму в змінній *newMDIChild.*Властивість *MdiParent* цього вікна повинна містити посилання на батьківське вікно програми MDI, тому записують в нього посилання на об'єкт класу Form1, використовуючи ключуву властивість this.

Для тог  щоб MDI-вікно з'явилося на екрані, його необхідно відобразити явним чином за допомогою методу *Show*.



Засоби керування дочірніми MDI-вікнами дозволяють легко впорядкувати ці вікна одним із трьох способів. Перш за все, вікна можуть бути впорядковані з перекриттям (каскадним чином). Крім цього, можливо розташування вікон поруч по горизонталі і по вертикалі.



Щоб користувач міг впорядковувати дочірні MDI-вікна, в меню Window вносять рядки Cascade, Horizontally Tile і Vertically Tile і створюють необхідні обробники подій.

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

{

this.LayoutMdi(MdiLayout.Cascade);

}

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

{

this.LayoutMdi(MdiLayout.TileHorizontal);

}

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

{

this.LayoutMdi(MdiLayout.TileVertical);

}

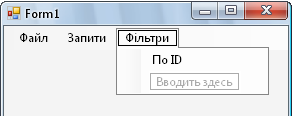
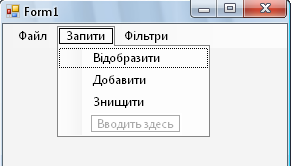
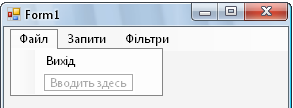
**Приклад виконання лабораторної роботи.**

**Завдання.** Спроектувати та розробити програму на основі MDI інтерфейсів. Програма має включати пошук по вибірці, та впорядкування вікон.

Приклад на основі інформаційної системи про книги

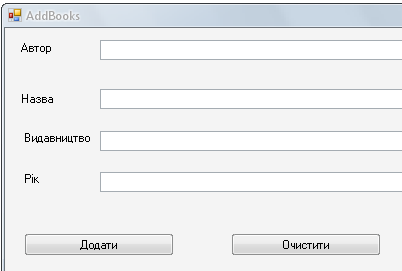
Хід виконання прикладу:

1. У вікні властивостей міняємо властивість форми *IsMDIContainer на true.*
2. Переносимо на головну форму елемент MenuStrip і заповнюємо відповідним чином.

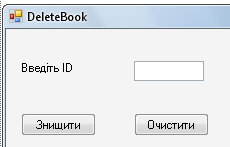


1. Створюємо відповідні форми:

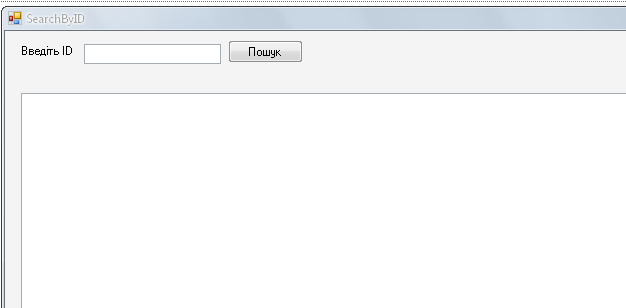
* Для відображення всієї літератури створюємо форму з назвою *viewBooks з* елементом *TextBox* з властивістю *MultiLine – true і ScrollBars – Both.*
* Для додавання літератури створюємо форму з назвою *AddBooks* з 4-ма елементами *Label* і *TextBox* та 2-ма елементами *Button*.



* Для знищення елементу по id літератури створюємо форму з назвою DeleteBook з елементами *Label* і *TextBox* та 2-ма елементами *Button*.



* Для пошуку по ID літератури створюємо форму з назвою SearchByID з елементами *Label* і *Button* та 2-ма елементами *TextBox*. Один елемент *TextBox* з властивістю *MultiLine – true*



1. У обробники подій меню батьківського MDI вікна вносимо код, котрий відображає відповідні дочірні MDI вікна. Кожне дочірне MDI вікно працює з класом яке основане на технології ADO.NET.

**Код програми:**

Клас 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.Windows.Forms;

namespace lab2

{

public partial class Form1 : Form

{

//Стрічка підєднання до БД Access

string conString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=lab2.accdb;Persist Security Info=False;";

public Form1()

{

InitializeComponent();

}

//Процедура викликається при запуску програми

private void Form1\_Load(object sender, EventArgs e)

{

IsMdiContainer = true;

//задає довжину і ширину батьківської MDI-форми

this.Width = 1000;

this.Height = 600;

}

//Виконується коли виберем команду Вихід з меню

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

{

this.Close();

}

//Виконується коли виберем команду Відобразити з меню

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

{

viewBooks vBooks = new viewBooks();

vBooks.MdiParent = this;

//використовуємо Ado.net клас для того що б взнати кількість записів

UseDB uDB = new UseDB(conString);

uDB.OpenCon(); //Підєднання до БД

//Міняємо назву дочірньої MDI-форми яка відображає всі записи в БД

vBooks.Text = "В БД "+ uDB.Cout().ToString()+" записів!!!!";

vBooks.Show();

uDB.CloseCon(); // Відєднання від БД

}

//Виконується коли виберем команду Добавити з меню

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

{

AddBooks abook = new AddBooks();

abook.MdiParent = this;

abook.Show();

}

//Виконується коли виберем команду Знищити з меню

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

{

DeleteBook dBook = new DeleteBook();

dBook.MdiParent = this;

dBook.Show();

}

//Виконується коли виберем команду Сортувати по ID з меню

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

{

SearchByID sbID = new SearchByID();

sbID.MdiParent = this;

sbID.Show();

}

}

}

Файл viewBooks.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace lab2

{

public partial class viewBooks : Form

{

//Стрічка підєднання до БД Access

string conString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=lab2.accdb;Persist Security Info=False;";

public viewBooks()

{

InitializeComponent();

}

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

{

String str = "";

DataTable table = new DataTable();

this.textBox1.Clear();

UseDB uDB = new UseDB(conString);

uDB.OpenCon();

table = uDB.AllBooks();

str=str+"| ID | Autor | Name | Vudannj | Rik |\r\n"; str=str+"|===========================================================================|\r\n";

foreach (DataRow dataRow in table.Rows)

{

str=str + String.Format("{0,-7}", dataRow[0].ToString()) + "| " + String.Format("{0,-25}", dataRow[1]) + "| " + String.Format("{0,-28}", dataRow[2]) + "| " + String.Format("{0,-10}", dataRow[3]) + "| " + String.Format("{0,-7}", dataRow[4]) + "\r\n";

}

textBox1.Text = str;

}

}

}

Файл AddBooks.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace lab2

{

public partial class AddBooks : Form

{

public AddBooks()

{

InitializeComponent();

}

//Викликається коли користувач нажме на кнопку очистити

private void button2\_Click(object sender, EventArgs e)

{

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

}

//Стрічка підєднання до БД Access

string conString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=lab2.accdb;Persist Security Info=False;";

//Викликається коли користувач нажме на кнопку додати

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

{

UseDB uDB = new UseDB(conString);

Book myBook = new Book();

int row;

myBook.autor = textBox1.Text;

myBook.Name = textBox2.Text;

myBook.Vudannj = textBox3.Text;

myBook.rik = Convert.ToInt16(textBox4.Text);

uDB.OpenCon();

row=uDB.InsertBook(myBook);

if(row!=0) MessageBox.Show("Запит був доданий!!!");

else MessageBox.Show("Запит не був доданий!!!");

uDB.CloseCon();

}

}

}

Файл DeleteBook.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace lab2

{

public partial class DeleteBook : Form

{

//Стрічка підєднання до БД Access

string conString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=lab2.accdb;Persist Security Info=False;";

public DeleteBook()

{

InitializeComponent();

}

//Викликається коли користувач нажме на кнопку очистити

private void button2\_Click(object sender, EventArgs e)

{

textBox1.Clear();

}

//Викликається коли користувач нажме на кнопку знищити

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

{

UseDB uDB = new UseDB(conString);

int id;

uDB.OpenCon();

id = Convert.ToInt16(textBox1.Text);

uDB.DeleteBook(id);

}

}

}

Файл SearchByID.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace lab2

{

public partial class SearchByID : Form

{

public SearchByID()

{

InitializeComponent();

}

//Стрічка підєднання до БД Access

string conString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=lab2.accdb;Persist Security Info=False;";

//Викликається коли користувач нажме на кнопку пошук

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

{

UseDB uDB = new UseDB(conString);

string str = "";

int id = Convert.ToInt16(textBox1.Text);

uDB.OpenCon();

DataTable table = uDB.SearchByID(id);

foreach (DataRow dataRow in table.Rows)

{

str=str+String.Format("{0,-5}", dataRow[0].ToString()) + String.Format("{0,-15}", dataRow[1]) + String.Format("{0,-28}", dataRow[2]) + String.Format("{0,-10}", dataRow[3]) + String.Format("{0,-7}", dataRow[4]);

}

uDB.CloseCon();

textBox2.Text = str;

uDB.CloseCon();

}

}

}

Клас UseDB

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data.OleDb;

using System.Data;

using System.Data.Common;

namespace lab2

{

class UseDB

{

OleDbConnection conn;

DataSet ds = new DataSet();

DbCommand comm;

public UseDB(string conString)

{

conn = new OleDbConnection(conString);

}

//метод підєднання до БД

public void OpenCon()

{

conn.Open();

}

//Метод визначення кількості записів в БД

public int Cout()

{

int count;

comm = conn.CreateCommand();

comm.CommandText = "Select count(\*) from books";

count = (int)comm.ExecuteScalar();

return count;

}

//Метод який вибирає всі елементи з БД

public DataTable AllBooks()

{

DbDataReader dr;

DataTable temp = new DataTable();

comm = conn.CreateCommand();

comm.CommandText = "Select \* from books";

dr = comm.ExecuteReader();

temp.Load(dr);

return temp;

}

//метод пошуку елементів по ІД

public DataTable SearchByID(int id)

{

DbDataReader dr;

DataTable temp = new DataTable();

comm = conn.CreateCommand();

comm.CommandText = "Select \* from books where id=" + id;

dr = comm.ExecuteReader();

temp.Load(dr);

return temp;

}

//Метод запису елементів в БД

public int InsertBook(Book tempBook)

{

comm = conn.CreateCommand();

//INSERT IN TO DATABASE

comm.CommandText = "INSERT INTO books (Autor, Name, Vudannj, Rik)VALUES ('" + tempBook.autor + "','" + tempBook.Name + "','" + tempBook.Vudannj + "'," + tempBook.rik + ");";

int rows = comm.ExecuteNonQuery();

return rows;

}

//метод знищення елементу по ІД

public int DeleteBook(int id)

{

comm = conn.CreateCommand();

//Delete IN TO DATABASE

comm.CommandText = "DELETE FROM books WHERE id=" + id;

int rows = comm.ExecuteNonQuery();

return rows;

}

//метод Відєднання від БД

public void CloseCon()

{

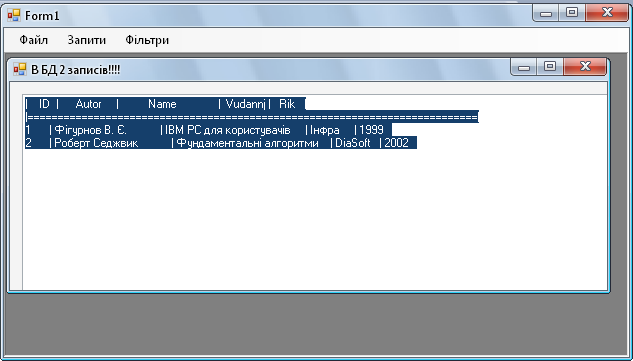
conn.Close();

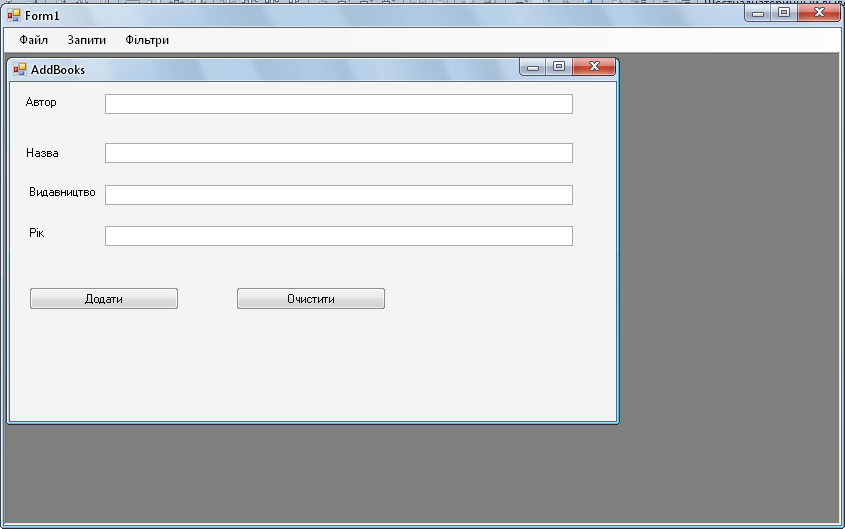
}

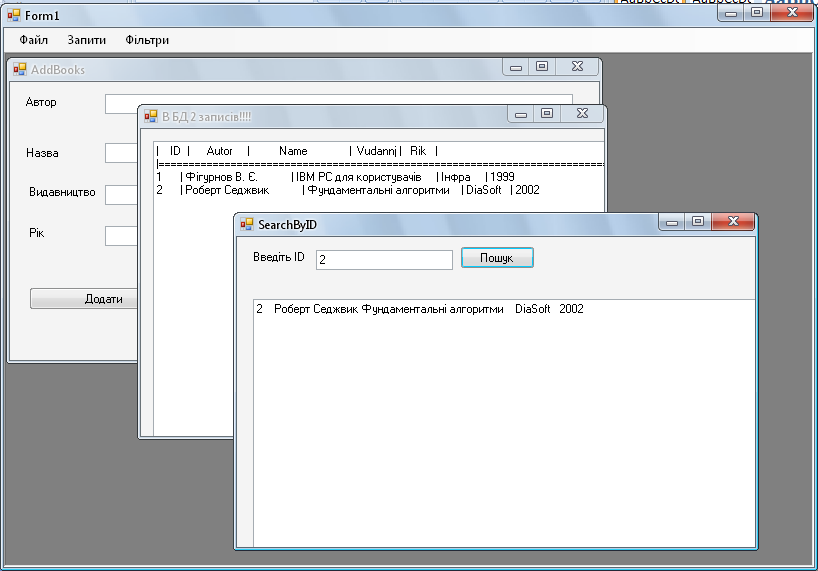
}

}

Результат виконання програми

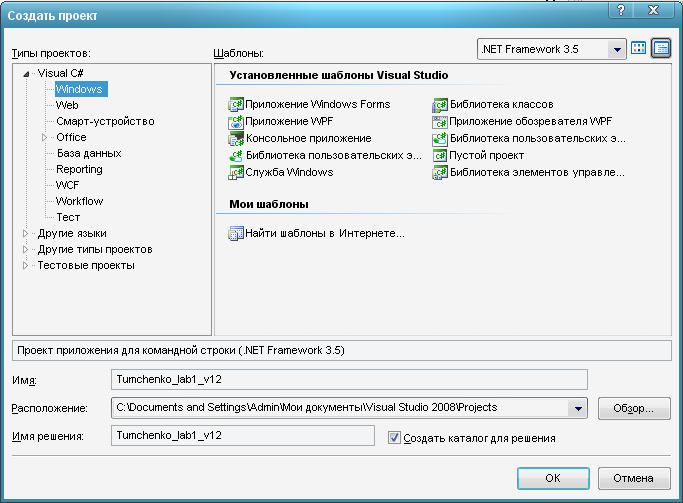






**Хід роботи.**

1. Ознайомитися з теоретичними відомостями та прикладом виконання лабораторної роботи.
2. За номером у журналі вибрати власний варіант.
3. Розробити блок схему програми чи UML схему.
4. Запустити Visual Studio 2003/2005/2008. Вибрати в меню **Файл/Создать/Проект,** у вікні, що відкриється вибрати тип проекту – Windows Forms проект і ввести назву яка відповідає призвіщу студента, його варіанту і номеру лабораторної роботи.



1. Реалізувати програму згідно завдання та варіанту.
2. Розробити звіт. Звіт має включати, номер і назву лабораторної роботи, мету, теоретичні відомості, задачу, розробку структури даних (опис змінних та їх тип), блок-схему чи UML діаграми (діаграму класів, діаграму прецедентів, діаграма послідовності), код програми, результат виконання програми та висновок.
3. Захист лабораторної роботи.

**Варіанти завдань до лабораторної роботи №2.**

|  |  |  |
| --- | --- | --- |
| Варіант | Назва інформаційної БД | Поля |
| 1 | Автомобілі | Id, марка автомобіля, рік випуску, тип двигуна, об’єм двигуна, розхід палива на 100 км. |
| 2 | Комп’ютери | Id, материнська плата, ОП, тип ОП, об’єм вінчестера, відеокарта |
| 3 | Кондиціонери | Id, назва, площа кондиціонування, потужність охолодження, потужність обігріву, колір |
| 4 | Фільми | Id, назва, жанр, кіностудія, якість, тривалість |
| 5 | Музичний альбом | Id, виконавець, назва альбому, жанр, кількість пісень, тривалість |
| 6 | Телевізори | Id, виробник, діагональ, тип телевізора, потужність |
| 7 | Носії інформації  (Flash) | Id, фірма, об’єм, тип, швидкість запису, швидкість читання |
| 8 | Комп’ютерні ігри | Id, назва, жанр, необхідна операційна система, мова інтерфейсу, розробник |
| 9 | Принтери | Id, фірма, модель, тип, швидкість друку, резерв сторінок |
| 10 | Мобільні телефони | Id, модель, дисплей, карта пам’яті, вага, колір |
| 11 | Підприємства | Id, назва, сфера діяльності, адреса, № телефону, кількість працівників |
| 12 | Планшетні ПК | Id, процесор, ОП, пам'ять, дисплей, вага, колір |
| 13 | Програмне забезпечення | Id, назва, версія, фірма виготовлення, призначення, платформа |
| 14 | Фотоапарати | Id, назва, матриця, зум, дисплей, вага |
| 15 | Велосипеди | Id, назва, тип, розмір рами, кількість швидкостей, діаметр коліс |
| 16 | Літаки | Id, назва, тип, швидкість, пасажироміскість, дальність польоту |
| 17 | Локомотиви | Id, назва, тип, швидкість, потужність, робоча маса |
| 18 | Готелі | Id, назва, кількість зірок, ціна за добу, адреса |
| 19 | Зброя | Id, назва, тип, калібр, вага, термін гарантії |
| 20 | Житло | Id, район, кількість кімнат, ціна за квадратний метр, площа |
| 21 | Ресторани | Id, назва, кухня, адреса, години роботи, кількість працівників |
| 22 | Футбол | Id, назва команди, країна, група, ліга, очки |
| 23 | Посуд | Id, тип, матеріал, об’єм, виробник, ціна |
| 24 | Канцтовари | Id, назва канцтоварів, група канцтоварів, ціна, виробник, |
| 25 | Косметика | Id, назва косметичного товару, ціна, фірма, вид косметики |
| 26 | Гелікоптери | Id, назва, тип, швидкість, пасажироміскість, дальність польоту |
| 27 | Відпочинкові бази | Id, назва бази, місто, ціна кімнати за добу, відстань до моря, |
| 28 | Навчальні заклади | Id, назва, рівень акредитації, кількість працівників, кількість учнів, |
| 29 | Напої | Id, вид, назва, об’єм, ціна, |
| 30 | Взуття | Id, тип, назва, ціна, виробник, матеріал, колір |

**Вимоги до звітів.**

Варіант Практичної роботи обирається згідно списку в журналі.

З усіх чотирьох боків аркуша повинні бути залишені поля розміром 20 мм, заповненість сторінки має становити 75% від її загального обсягу. Текст набирають на комп'ютері, шрифтом ґарнітурою Times New Roman. Заголовок розділу, підрозділу від попереднього тексту відокремлюють трьома інтервалами. Відступ від назви підрозділу (пункту, підпункту) до наступного тексту повинен складати півтора інтервала. Назви розділів та підрозділів для зручності читання потрібно виділяти напівжи-рним шрифтом, текст звіту вирівнювати по ширені, а визначення понять у тексті та термінів – курсивом.

Звіт має включати, номер і назву лабораторної роботи, мету, теоретичні відомості, задачу, розробку структури даних (опис змінних та їх тип), блок-схему чи UML діаграми (діаграму класів, діаграму прецедентів, діаграма послідовності), код програми, результат виконання програми та висновок.

Звіт здається скріпленими скріпкою у фай лику.

**Оцінювання практичної роботи.**

Практична робота оцінюється на **9-10 балів** коли студент: повністю зробив програму згідно варіанту, та оформив звіт згідно вимог; при захисті звіту дає повну, розгорнуту, вичерпну відповідь на питання; здатний здійснювати - порівняльний аналіз різних теорій, концепцій, підходів та самостійно робить логічні висновки й узагальнення; здатний висловлювати та аргументувати власне ставлення до альтернативних поглядів на дану тему.

Практична робота оцінюється на **7-8 балів** коли студент: повністю зробив програму згідно варіанту, та оформив звіт згідно вимог; при захисті звіту розкриває зміст питання в цілому правильно, але все ж таки студентом допущені помилки, посилання на конкретні історичні періоди та факти, неточності у формулюванні термінів і категорій, проте з допомогою викладача він швидко орієнтується й знаходить правильні відповіді

Практична робота оцінюється на **4-6 балів** коли студент: повністю зробив програму згідно варіанту, та оформив звіт згідно вимог; при захисті звіту відповідь малообґрунтована, неповна; студент лише з допомогою викладача може зрозуміти та виправити свої помилки.

Практична робота оцінюється на **1-3 бал** коли студент: повністю зробив програму згідно варіанту, та оформив звіт згідно вимог; при захисті звіту у відповіді відсутні належні докази й аргументи, зроблені висновки не відповідають загальноприйнятим, хибні; характер відповіді дає підставу стверджувати, що студент не знає правильної відповіді; допущені грубі помилки й студент не може їх виправити.

При здачі та захисту звіту знімається **по 1 балу** коли студент:

* Не оформив звіт відповідно до вимог.
* Недоробив програму згідно завдання чи варіанту.
* Немає UML діаграм чи блок-схем, вони не відповідають реалізованій програмі або генеровані автоматизованими засобами проектування.
* В звіті присутній незрозумілий текст.