Розділ 2 "Мультимедійні об’єктно-орієнтовані додатки"

Змістовий модуль 3

"Базові концепції об’єктно-орієнтованого програмування"

Лабораторна робота №8

Використання простих класів

Мета роботи – отримати практичні навички розробки програм з простішими класами.

Дана лабораторна робота сприяє напрацюванню наступних **компетентностей** у відповідності до Національної рамки кваліфікації:

Знання:

основні концепції об'єктно-орієнтованого програмування;

загальна форма визначення класу;

правила створення об'єктів;

особливості застосування змінних типу посилання;

типову структуру C # - програми з кількома класами.

Уміння:

розробити програму, що містить клас (класи), який інкапсулює інформацію про заданий об'єкт (предметна область задається індивідуально);

розробити програму, що обробляє інформацію (елементарні обчислення і виведення на екран вмісту елементів-даних) про декілька об'єктів одного класу.

Комунікації:

аргументована взаємодія з клієнтами та замовниками при виборі парадигми проектування, що забезпечує здатність мислити в термінах об'єктно-орієнтованої моделі і базується на знанні сучасних інформаційних технологій та умінні використовувати інтернет-ресурси.

Автономність і відповідальність:

самостійне формулювання рекомендацій щодо декомпозицію проекту на сукупність окремих блоків;

здатність обґрунтувати структуру класу та доцільність застосування певних типів даних для його опису.

Основні положення

Усі C# -програми оформляються у вигляді класу. Клас - це шаблон, який визначає форму об'єкта. Він задає як дані, так і код, який оперує цими даними.

Об'єкти - це екземпляри класу. Важливо чітко розуміти, що клас - це логічна абстракція. Про її реалізації немає сенсу говорити доти, поки не буде створено об'єкт класу, і в пам'яті не з'явилося фізичне його подання.

Методи і змінні, складові клас, називаються елементами (членами) класу.

Загальна форма визначення класу.

Незважаючи на те, що дуже прості класи можуть включати тільки код або тільки дані, більшість реальних класів містять і те, і інше.

Дані містяться в змінних екземплярів, що визначаються класом, а код - в методах.

Требо відзначити, що клас визначає також ряд спеціальних елементів - даних і елементів - методів, наприклад статичні змінні, константи, конструктори, деструктори, індексатори, події, оператори та властивості.

Поки ми обмежимося розглядом змінних екземплярів і методів класу, а потім познайомимося з конструкторами і деструкторами.

Клас створюється за допомогою ключового слова class. Загальна форма визначення класу, який містить тільки змінні екземплярів і методи, має наступний вигляд:

```
class ім'я_класу
{
// Оголошення змінних примірників.
        доступ тип зміна1;
        доступ тип зміна2;
        //. . .
// Оголошення методів.
        доступ тип_що повертається ім’я_методу1 (параметри)
        {
                // Тіло методу
        }
        доступ тип_що повертається ім’я_методу2 (параметри)
        {
```

```
            // Тіло методу
        }
        //. . .
}
```

Класи, які ми використовували досі в курсі «Основи програмування», містили тільки один метод - Main (). Незабаром ми дізнаємося, як створювати й інші методи. Слід зазначити, що в загальній формі визначення класу метод Main ( ) не заданий. Він потрібен тільки в тому випадку, якщо клас, що створюється, є відправною точкою програми.

Приклад визначення класу.

Створимо клас, який інкапсулює інформацію про будівлі (будинки, складські приміщеннях, офіси і т.д.). В цьому класі (назвемо його Building) будуть зберігатися три елементи інформації про будівлі (кількість поверхів, загальна площа і кількість мешканців).

Нижче представлена перша версія класу Building. В ньому визначено три змінні екземпляра: floors, area і occupants. Зверніть увагу на те, що клас Building не містить жодного методу. Тому поки його можна вважати класом даних. (У наступних лабораторних роботах ми доповнимо його методами.)

```
class Building
{
        public int floors;          // кількість поверхів
        public int area;            // загальна площа основи будівлі
        public int occupants;       // кількість мешканців
}
```

Змінні екземпляру, які визначені в класі Building, ілюструють загальний спосіб їх оголошення. Формат оголошення змінної примірника такий:

доступ тип ім'я_змінної;

Тут елемент «доступ» представляє специфікатор доступу, елемент «тип» - тип змінної екземпляру, а елемент «ім'я_змінної» - ім'я цієї змінної. Таким чином, якщо не вважати специфікатор доступу, то змінна екземпляра оголошується так само, як локальна змінна.

У класі Building всі змінні екземпляра оголошені з використанням

модифікатора доступу public, який дозволяє отримувати до них доступ з боку коду, розташованого навіть поза класом Building.

Визначення class створює новий тип даних. В даному випадку цей новий тип даних називається Building. Це ім'я можна використовувати для оголошення об'єктів типу Building.

Оголошення class - це лише опис типу; воно не створює реальних об'єктів. Таким чином, попередній код не означає існування об'єктів типу Building.

Щоб реально створити об'єкт класу Building можливо використовувати таку інструкцію:

```
Building house = new Building ( );
```

Після виконання цієї інструкції house стане екземпляром класу Building, тобто знайде «фізичну» реальність. Детально цю інструкцію ми розглянемо нижче.

При кожному створенні екземпляра класу створюється об'єкт, який містить власну копію кожної змінної екземпляра, яка визначена цим класом.

Таким чином, кожен об'єкт класу Building буде утримувати власні копії змінних екземпляра floors, area і occupants. Для доступу до цих змінних використовується оператор «крапка» (.). Він пов'язує ім'я об'єкта з ім'ям його елемента.

Загальний формат цього оператора має такий вигляд:

```
об’єкт.елемент
```

Об'єкт вказується зліва від оператора «крапка», а його елемент - справа.

Наприклад, щоб привласнити змінної floors значення 2, використається наступна інструкція:

```
house.floors = 2;
```

У загальному випадку оператор «крапка» можна використовувати для доступу як до змінних екземплярів, так і методам.

Приклад 1. Лістинг програми, в якій використовується клас Building.

```
using System;
class Building
{
        public int floors;          // Кількість поверхів
        public int area;            // Загальна площа основи будівлі
        public int occupants;       // Кількість мешканців
}
class BuildingDemo
{
public static void Main ()
{
Building house = new Building (); // створення об'єкта типу Building
int areaPP;
// Площа, що приходить на одного мешканця
// Присвоюється значення полям у об'єкті house
house.occupants = 4;
house.area = 2500;
house.floors = 2;
// Обчислюємо площу, що припадає на одного мешканця будинку
areaPP = house.area / house.occupants;
Console.WriteLine ("Будинок має: \n" +
house.floors + " поверху \n" +house.occupants + " мешканця \n" +
house.area + " квадратних метрів загальної площі, з них \n" +
areaPP + " припадає на одну людину");
}
}
```

Результат виконання програми.

```
Будинок має:
2 поверху
4 мешканця
2500 квадратних метрів загальної площі, з них
625 припадає на одну людину
```

Приклад роботи з двома об'єктами одного класу.

Згідно основного принципу програмування класів, кожен об'єкт класу має власні копії змінних екземпляра, які визначені у цьому класі. Таким чином, вміст змінних в одному об'єкті може відрізнятися від вмісту аналогічних змінних в іншому.

Між двома об'єктами одного класу немає зв'язку, за винятком того, що вони є об'єктами одного і того ж типу. Наступна програма демонструє це.

Приклад 2. Робота з двома об'єктами класу Building.

```
using System;
class Building
{
        public int floors;          // кількість поверхів
        public int area;            // загальна площа основи будівлі
        public int occupants;       // кількість мешканців
}
class BuildingDemo
{
public static void Main ()
{
Building house = new Building ();
Building office = new Building ();
int areaPP; // площа, яка припадає на одного мешканця
// Присвоюється значення полям у об'єкті house
house.occupants = 4;
house.area = 2500;
house.floors = 2;
// Присвоюється значення полям у об'єкті office
office.occupants = 25;
office.area = 4200;
office.floors = 3;
// Обчислюється площа, що припадає на одного мешканця.
areaPP = house.area / house.occupants;
Console.WriteLine (" Будинок має: \n" +
house.floors + " поверху \n" +
house.occupants + " мешканця \n" +
house.area + " кв.м. загальної площі, з них \n" +
```

```
areaPP + " припадає на одну людину");
Console.WriteLine ();
// Обчислюється площа, що припадає на одного працівника офісу.
areaPP = office.area / office.occupants;
Console.WriteLine (" Офіс має: \n" +
office.floors + " поверху \n" +
office.occupants + " працівників \n" +
office.area + " кв.м. загальної площі, з них \n" +
areaPP + " припадає на одну людину");
}
}
```

Результат виконання програми.

Будинок має:
2 поверху
4 мешканця
2500 кв.м. загальної площі, з них
625 припадає на одну людину

Офіс має:
3 поверху
25 працівників
4200 кв.м. загальної площі, з них
168 припадає на одну людину

Звернути увагу на особливості операторів визначення класів і доступу до елементів-даних екземплярів класу.

Порядок виконання лабораторної роботи

Загальна частина.

1. Набрати, відкомпілювати і запустити на виконання приклади програм, які були наведені в розділі «Основні положення» даної лабораторної роботи.

2. Проекспериментуйте з програмами:

змініть вихідні дані;

досліджуйте, як впливають синтаксичні помилки на результат компіляції програми. Які при цьому виникають помилки компіляції?

Індивідуальна частина.

1. З таблиці 1 вибрати індивідуальний варіант предметної області.

Розробити відповідно предметної області клас (по аналогії з прикладом 1). Створити два екземпляри поточного класу і здійснюйте їх обробку (по аналогії з прикладом 2).

2. На базі таблиці 1 сформуйте індивідуальний варіант завдання з двома предметними областями.

Для кожної предметної області розробити відповідні два класи. Створити два об'єкта кожного класу і здійснюйте їх обробку.

Таблиця 1

Індивідуальні варіанти предметної області

Варіант	Предметна область
1	Поліграфічні верстати.
2	Папір для друку.
3	Фарби.
4	Принтери.
5	Автомобілі.
6	Літаки.
7	Домашні тварини.
8	Домашні птахи.
9	Готельний бізнес.
10	Викладачі.
11	Студенти.
12	Туризм.
13	Квіти.

Кожна програма з індивідуального завдання повинна:

здійснювати елементарні операції (в методі Main ()) з елементами-даними екземплярів відповідних об'єктів. Формули для обчислень визначити самостійно, вони повинні відповідати обраній предметній області.

виводити на екран результати обчислень і вміст елементів-даних всіх екземплярів визначених вище об'єктів.

3. Проаналізуйте вирази для обробки елементів-даних: визначить допустимі діапазони зміни вхідних величин, їх розмірність і тип.

4. Підготуйте контрольні приклади, які повною мірою характеризують аналізовані вирази.

5. Розробить алгоритм обчислення і намалювати його графічну схему (блок-схему).

6. Набрати і відкомпілювати текст програми, усуваючи при необхідності помилки.

Зміст звіту

1. Титульний лист.

2. Цілі лабораторного заняття і вказівка, які навички та вміння передбачається отримати в результаті його виконання.

3. Тексти налагоджених програм загальної частини лабораторного заняття з необхідними коментарями і результатом виконання.

4. Аналіз предметної області індивідуального завдання й докладного опису спроектованого класу / класів.

5. Тексти налагоджених програм з результатом виконання всіх контрольних прикладів індивідуального завдання.

6. Висновки.

Контрольні питання

1. У чому відмінність процедурного та об'єктно-орієнтованого стилю програмування?

2. Дайте визначення класу. Опишіть можливі елементи класу.

3. Що розуміється під об'єктної моделлю, які її властивості?

4. Опишіть структури найпростішої об'єктно-орієнтованої програми.

5. Перелічіть основні етапи розробки об'єктно-орієнтованих програм.