Лабораторна робота № 3.

Тема: ***Об'єкти й класи (конструктори, деструктори, методи класу, спадкування).***

Мета роботи:

Ознайомлення основою об'єктного підходу в мові C#, створенням об'єктів, класів і механізмом спадкування.

Теоретичні відомості

Класи та об’єкти

Клас є основою для створення об'єктів. Клас - це узагальнене поняття, що визначає характеристики і поведінку певної кількості об'єктів, які називаються екземплярами класу. "Класичний" клас містить дані, що визначають властивості об'єктів класу, і методи, що визначають їхню поведінку. У класі визначаються дані й код, який працює із цими даними. Об'єкти є екземплярами класу.

Опис класу містить ключове слово **`class`**, за яким слідує його ім'я, а далі в фігурних дужках - тіло класу. Крім того, для класу можна задати його базові класи (предки) і ряд необов'язкових атрибутів зі специфікатором, що визначає різні характеристики класу. Синтаксис опису класу:

```
[Атрибути] [специфікатори] class ім'я_класу  [: предки]
{ // тіло_класу
[Атрибути] [специфікатори] [const] тип ім'я1 [= початкове_значення1];
[Атрибути] [специфікатори] [const] тип ім'я2 [= початкове_значення2];
…
[Атрибути] [специфікатори] тип_що_повертається
ім'я_методу_1(список_параметрів)
{ тіло_методу }

[Атрибути] [специфікатори] тип імя_властивості
{
   [get код_доступу]
   [set код_доступу] / [ init [ ініціалізація ]
}
…
}
```

Специфікатори визначають властивості класу, а також доступність класу для інших елементів програми. Можливі значення специфікаторів для класів: `new, public, protected, internal, protected internal, private, static, sealed, abstract.` Клас можна описувати безпосередньо всередині простору імен або всередині іншого класу. В останньому випадку клас називається вкладеним. Залежно від місця опису класу деякі з цих специфікатором можуть бути заборонені. За замовчуванням, тобто якщо жоден специфікатор доступу не вказаний, мається на увазі специфікатор `internal`.

Методи та змінні, що належать до класу, називаються членами класу. При визначенні класу оголошуються дані, які він містить, і код, що працює із цими даними. Дані зберігаються в змінних екземпляра, які визначені класом,

а код знаходиться в методах. У C# визначені кілька специфічних різновидів членів класу. Це - змінні екземпляра, статичні змінні, сталі, методи, конструктори, деструктори, індексатори, події, оператори й властивості.

Можливі значення специфікаторів для даних та методів : new, public, protected, internal, protected internal, private, static, readonly , volatile.

Члени класу з типом доступу public доступні також за межами даного класу, з типом доступу protected – усередині членів даного класу та похідних класах , з типом доступу private - тільки для членів даного класу. Тип доступу internal застосовується для типів, доступних в межах одному складання.

Іноді потрібно створити поле, яке з одного боку, має бути доступне для використання, з іншого боку, можливість щось зробити з цим полем має обмеження. Наприклад, полю не можна присвоювати довільне значення, а лише значення з якогось діапазону. Властивість пропонує простий і зручний спосіб вирішення цієї проблеми. Значення специфікатором для властивостей і методів аналогічні. Найчастіше властивості оголошуються як відкриті (з специфікатором public). Код доступу являє собою блоки операторів, які виконуються при отриманні (get) або встановлення (set) властивості. Може бути відсутнім або частина get, або set, але не обидві одночасно. Якщо відсутня частина set, то властивість доступна тільки для читання. Якщо відсутня частина get, то властивість доступна тільки для запису.

C# 9 і пізніших версій ключове слово init визначає метод доступу у властивості або індексаторі. Метод завдання лише для ініціалізації призначає значення властивості або елемент індексатора тільки під час створення об'єкта. Це забезпечує незмінність, щоб після ініціалізації об'єкта його не можна було змінити знову.

Безпосередньо ініціалізація змінних в об'єкті (змінних екземпляра) здійснюється в конструкторові. У класі можуть бути визначені декілька конструкторів.

Приклад:

```
class Animal{
      public string Name;
      private int Weight;
      protected int Type;
      public Animal(int W, int T, string N){
            Weight=W;
            Type=T;
            Name=N;
      }
      public int Getweight(){return Weight;}
}
```

Створення об’єкта

```
      Ім'я_класу ім'я_об’єкта = new ім'я _ класу ();
```

При створенні об’єкта класу відбувається виклик відповідного конструктора класу.

Конструктор і деструктор

Конструктор класу – метод для ініціалізації об'єкта при його створенні. Він має те ж ім'я, що і його клас. У конструкторах тип значення, що вертається, не вказується явно. Конструктори використовуються для присвоювання початкових значень змінним екземпляра, певним класом, і для виконання будь-яких інших процедур ініціалізації, необхідних для створення об'єкта. Всі класи мають конструктори незалежно від того, визначений він чи ні. За замовчуванням у C# передбачено наявність конструктора, який присвоює нульові значення всім змінним класу (для змінних звичайних типів) і значення null (для змінних посилального типу). Але якщо конструктор явно визначений у класі, то конструктор за замовчуванням використовуватися не буде.

```
Ім'я_класу(список_параметрів) { тіло _ конструктора }
```

Розглянемо основні властивості конструкторів:

1. Конструктор не повертає значення, навіть типу **void**.
2. Клас може мати кілька конструкторів з різними параметрами для різних видів ініціалізації.
3. Якщо програміст не вказав жодного конструктора або якісь поля не були ініціалізований, полям присвоюється нуль або значення **null**.

Деструктор – метод, що викликається автоматично при знищенні об’єкта класу (безпосередньо перед “ складанням сміття ”). Деструктор не має параметрів значення, що й вертається.

```
~ім'я_ класу () { тіло _ деструктора }
```

Спадкування

Спадкування — ця властивість, за допомогою якого один об'єкт може набувати властивості іншого. При цьому підтримується концепція ієрархічної класифікації, що має напрямок зверху вниз. Використовуючи спадкування, об'єкт повинен визначити тільки ті якості, які роблять його унікальним у межах свого класу. Він може успадковувати загальні атрибути від своїх батьківських класів. Синтаксис:

```
class ім'я_класу : ім'я_батьківського_класу
{ тіло _ класу }
```

Приклад:

```
class Predator : Animal {
private int speed;
public  int Speed { get; set; }

}
```

За допомогою спадкування створюється ієрархія класів (відношення ‘бути’). Крім того, можна побудувати ще одну структуру – ієрархію об'єктів (тоді, коли один об'єкт є частиною іншого – відношення ‘ частина - ціле ’).

Завдання до лабораторної роботи:

Порядок виконання роботи:

1) Клонувати репозиторій https://classroom.github.com/a/LGghmBVp.
2) Розробити класи, методи та властивості згідно з варіантом завдання.
3) Додати код для тестування всіх можливосте створених класів із виведення відповідної інформації.
4) Підготувати звіт в електронному виді та надіслати мудл(https://moodle.chnu.edu.ua/course/view.php?id=3371) .
5) Буди готовим відповідати на контрольні питання (дивись внизу).

Завдання 1. Варіанти задач. Для розв'язання задачі згідно варіанту створити клас із полями, конструкторами, методами та властивостями. До запропонованих полів, методів та властивосте можна додавати власні.

1.1. Задано масив точок. Вивести в консоль інформацію про точку та відстань до центра координат. Точки які знаходяться більше середньої відстані, перемістити на заданий вектор. Створити клас Point, розробивши такі елементи класу:

- Поля (захищені):
  - координати точки (int x, y);
  - колір точки (int c);
- Конструктори, що дозволяють створити екземпляр класу:
  - з нульовими координатами;
  - із заданими координатами.
- Методи, що дозволяють:
  - вивести координати точки на екран;
  - розрахувати відстань від початку координат до точки;
  - перемістити точку на вектор з координатами (x_1, y_1).
- Властивості:
  - отримати-встановити координати точки (доступні для читань і запису);
  - отримати колір точки (доступна тільки для читання).

1.2. Задано масив трикутників. Вивести в консоль інформацію про трикутники та їх площу і периметр. Створити клас Triangle (трикутник), розробивши такі елементи класу:

- Поля (захищені):
  - сторони (int a, b, c);
  - колір трикутника (int color);
- Конструктор, що дозволяє створити екземпляр класу з заданими довжинами сторін.
- Методи, що дозволяють:
  - вивести довжини сторін трикутника;

- розрахувати периметр трикутника;
- розрахувати площу трикутника.

- Властивості:
  - дозволяє отримати-встановити довжини сторін трикутника (доступні для читання і запису), запис повинен здійснюватися з існування трикутника;
  - отримати колір (доступна тільки для читання).

1.3. Задано масив чотирикутників. Вивести в консоль інформацію про чотирикутники та їх площу і периметр . Визначити скільки є квадратів. Створити клас Rectangle (прямокутник), розробивши такі елементи класу:

- Поля (захищені):
  - сторони (int a, b);
  - колір прямокутника (int c);
- Конструктор, що дозволяє створити екземпляр класу з заданими довжинами сторін.
- Методи, що дозволяють:
  - вивести довжини сторін прямокутника на екран;
  - розрахувати периметр прямокутника;
  - розрахувати площу прямокутника;
  - дозволяє встановити, чи є даний прямокутник квадратом.
- Властивості:
  - отримати-встановити довжини сторін прямокутника (доступні для читання і запису);
  - отримати колір (доступна тільки для читання).

1.4. Задано масив ромбів. Вивести в консоль інформацію про та їх площу і периметр. Визначити скільки є квадратів. Створити клас Romb (ромб), розробивши такі елементи класу:

- Поля (захищені):
  - сторона та діагональ (int a, d1);
  - колір ромба (int c);
- Конструктор, що дозволяє створити екземпляр класу з заданими довжинами.
- Методи, що дозволяють:
  - вивести довжини на екран;
  - розрахувати периметр ромба;
  - розрахувати площу ромба;
  - дозволяє встановити, чи є даний ромб є квадратом;.
- Властивості:

- отримати-встановити довжини (доступні для читання і запису);
- отримати колір (доступна тільки для читання).

1.5. Задано масив монета(Money) який моделює гаманець та масив товарів. Визначити кожного з товарів можна купити за кошти які знаходяться в гаманці. Створити клас Money, розробивши такі елементи класу:

- Поля(захищені):
  - номінал купюри (int nominal), //
  - кількість купюр (int num); //
- Конструктор, що дозволяє створити екземпляр класу з заданими значеннями полів.
- Методи, що дозволяють:
  - вивести номінал і кількість купюр;
  - визначити, чи вистачить коштів на купівлю товару на суму N гривень.
  - визначити, скільки шт товару вартості N гривень можна придбати на наявні грошові кошти.
- Властивості:
  - дозволяє отримати-встановити значення полів (доступні для читання і запису);
  - дозволяє розрахувати суму грошей (доступні тільки для читання).

1.6. Задано масив ромбів. Вивести в консоль інформацію про ромби та їх площу і периметр. Визначити скільки є квадратів. Створити клас DRomb(ромб), розробивши такі елементи класу:

- Поля: (захищені):
  - діагоналі (int d1, d2);
  - колір ромба (int c);
- Конструктор, що дозволяє створити екземпляр класу з заданими довжинами.
- Методи, що дозволяють:
  - вивести довжини на екран;
  - розрахувати периметр ромба;
  - розрахувати площу ромба дозволяє встановити, чи є даний ромб є квадратом;.
- Властивості:
  - отримати-встановити довжини (доступні для читання і запису);
  - отримати колір (доступна тільки для читання).

1.7. Задано масив трапецій. Вивести в консоль інформацію про трапеції та їх площу і периметр. Визначити скільки є квадратів. Створити клас Trapeze (трапеція), розробивши такі елементи класу:
- Поля:
  - основи та висота (int a, b , h);
  - колір трапеції (int c);
- Конструктор, що дозволяє створити екземпляр класу з заданими довжинами.
- Методи, що дозволяють:
  - вивести довжини на екран;
  - розрахувати периметр трапеції;
  - розрахувати площу трапеції
  - дозволяє встановити, чи є даний трапеція квадратом (доступні тільки для читання).
- Властивості:
  - отримати-встановити довжини (доступні для читання і запису);
  - отримати колір (доступна тільки для читання).

1.8. Задано масив рівнобедрених трикутників. Вивести в консоль інформацію про трикутники та їх площу і периметр. Визначити скільки є правильних трикутників. Створити клас ITriangle (isosceles triangle, рівнобедрений трикутник), розробивши такі елементи класу:
- Поля (захищені):
  - сторона основи та бічна сторона (int a, b);
  - колір трикутника (int c);
- Конструктор, що дозволяє створити екземпляр класу з заданими довжинами сторін.
- Методи, що дозволяють:
  - вивести довжини сторін трикутника;
  - розрахувати периметр трикутника;
  - розрахувати площу трикутника
  - перевіряє чи трикутник є правильним.
- Властивості:
  - дозволяє отримати-встановити довжини сторін трикутника (доступні для читання і запису);
  - отримати колір (доступна тільки для читання).

1.9. Задано масив прямокутних трикутників. Вивести в консоль інформацію про трикутники та їх площу і периметр. Визначити скільки є рівнобедрених трикутників. Створити клас ATriangle (angled triangle, прямокутний трикутник), розробивши такі елементи класу:

- Поля (захищені):
  - катети (int a, b);
  - колір трикутника (int c);
- Конструктор, що дозволяє створити екземпляр класу з заданими довжинами сторін.
- Методи, що дозволяють:
  - вивести довжини сторін трикутника;
  - розрахувати периметр трикутника;
  - розрахувати площу трикутника
  - перевіряє чи трикутник є рівнобедреним.
- Властивості:
  - дозволяє отримати-встановити довжини сторін трикутника (доступні для читання і запису);
  - отримати колір (доступна тільки для читання).

1.10. Задано масив дат. Перевірити коректність дат. Вивести в консоль дати впорядкованими за зростання. Знайти найбільшу кількість днів між датами у масиві. Створити клас Date (дата), розробивши такі елементи класу(створити власний клас і не використовувати стандартний) :

- Поля(захищені):
  - день (1-31),
  - місяць (1-12), //
  - рік (ціле число); //
- Конструктор, що дозволяє створити екземпляр класу з заданими значеннями полів.
- Методи, що дозволяють:
  - друку за шаблоном: “5 січня 2022 року”;
  - друку за шаблоном “05.01.2022”;
  - коректність дати;
  - кількість між двома датами.
- Властивості:
  - дозволяє отримати-встановити значення полів (доступні для читання і запису), встановлення полів класу повинні перевіряти коректність параметрів, що задаються;
  - дозволяє отримати століття (доступна тільки для читання).

Завдання 2. Варіанти задач. Побудувати ієрархію класів відповідно до варіанта завдання. Згідно завдання вибрати базовий клас та похідні. В класах задати поля, які характерні для кожного класу. Для всіх класів розробити метод Show(), який виводить дані про об’єкт класу. Створити масив базового класу та написати функцію наповняє масив різними

об’єктами похідних класів. Вивести масив впорядкований за деяким критерієм який характеризує всі об’єкти масиву.

2.1. Студент, викладач, персона, завідувач кафедри.
2.2. Службовець, персона, робітник, інженер.
2.3. Робітник, кадри, інженер, адміністрація.
2.4. Деталь, механізм, виріб, вузол.
2.5. Організація, страхова компанія, нафтогазова компанія, завод.
2.6. Журнал, книга, друковане видання, підручник.
2.7. Тест, іспит, випускний іспит, випробування.
2.8. Місце, область, місто, мегаполіс.
2.9. Іграшка, продукт, товар, молочний продукт.
2.10. Квитанція, накладна, документ, рахунок.
2.11. Автомобіль, поїзд, транспортний засіб, експрес.
2.12. Двигун, двигун внутрішнього згоряння, дизель, реактивний двигун.
2.13. Республіка, монархія, королівство, держава.
2.14. Савець, парнокопитне, птах, тварина.
2.15. Корабель, пароплав, вітрильник, корвет.

Контрольні питання

1) Що розуміється під терміном «клас»?
2) Які елементи визначаються в складі класу?
3) Яке співвідношення понять «клас» і «об'єкт»?
4) Що розуміється під терміном «члени класу»?
5) Які члени класу Вам відомі?
6) Які члени класу містять код?
7) Які члени класу містять дані?
8) Перелічите п'ять різновидів членів класу специфічних для мови C#.
9) Що розуміється під терміном «конструктор»?
10) Скільки конструкторів може містити клас мови C#?
11) Приведіть синтаксис опису класу в загальному виді. Проілюструйте його фрагментом програми мовою C#.
12) Які модифікатори типу доступу Вам відомі?
13) У чому полягають особливості доступу членів класу з модифікатором public?
14) У чому полягають особливості доступу членів класу з модифікатором private?
15) У чому полягають особливості доступу членів класу з модифікатором protected?
16) У чому полягають особливості доступу членів класу з модифікатором internal?
17) Яке ключове слово мови C# використовується при створенні об'єкта?

18) Приведіть синтаксис створення об'єкта в загальному виді. Проілюструйте його фрагментом програми мовою C#.
19) У чому полягає призначення конструктора?
20) Чи кожний клас мови C# має конструктор?
21) Які умовчання для конструкторів прийняті в мові C#?
22) Яким значенням інінціалізуються за замовчуванням значення посилального типу?
23) У якому випадку конструктор за замовчуванням не використовується?
24) Приведіть синтаксис конструктора класу в загальному виді. Проілюструйте його фрагментом програми мовою C#.
25) Що розуміється під терміном «деструктор»?
26) У чому полягає призначення деструктора?
27) Приведіть синтаксис деструктора класу в загальному виді. Проілюструйте його фрагментом програми мовою C#.
28) Що розуміється під терміном «успадкування»?
29) Яка класифікація об'єктів відповідає успадкуванню?
30) Що загального має дочірній клас із батьківським?
31) У чому полягає відмінність між дочірнім і батьківським класами?
32) Приведіть синтаксис опису спадкування класів у загальному виді. Проілюструйте його фрагментом програми мовою C#.
33) Якому відношенню відповідає ієрархія класів?
35) Якому відношенню відповідає ієрархія об'єктів?