Лабораторна робота № 4

Проектування та реалізація класів засобами C#

Вимоги до роботи:

У розроблених класах необхідно реалізувати такі концепції ООП, як "наслідування" та "поліморфізм". Потрібно описати *базовий клас* (клас-предок) та наслідувати від нього.

Частину функціональності класів треба *реалізувати з використанням властивостей*. Також необхідно реалізувати підтримку *серіалізації* об'єктів класів. Крім того потрібно описати *власний інтерфейс*, описати у ньому принаймні 2 методи чи властивості та реалізувати їх у класі. Один з методів потрібно реалізувати як *метод-розширення*.

Кожний клас *обов'язково має містити* принаймні два конструктори, хоча б один статичний метод та перевизначення методу ToString().

При написанні класів *не можна використовувати класи та методи розширень із просторів імен* System.Linq, System.Collection.Generic та System.Collection (списки, асоціативні масиви, черги, стеки, множини і т. п.). У варіантах, у назві класів у яких є «*», обмеження на використання колекцій відсутнє). Використання масивів допускається у всіх варіантах.

Обов'язково протестувати усі реалізовані можливості з використанням тестового проекту та *юніт-тестів*.

Варіанти завдань

1) Написати клас для подання многочленів із дійсними коефіцієнтами. Реалізація має містити:
   a) метод обчислення значення многочлена у точці та метод ToArray();
   b) властивості Degree, MaxCoefficient та індексатор;
   c) операції додавання, віднімання і множення многочленів та властивості порівняння і неявного перетворення типів;
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

2) Написати узагальнений клас для реалізації однонапрямленого списку (без зворотних зв'язків). Реалізація має містити:
   a) функції-члени для вставки елемента в початок списку, видалення елемента з початку списку, пошуку елемента у списку, добавлення у поточний список усіх елементів іншого списку;
   b) властивості Count, Head та індексатор;
   c) операції ==, + та * (має повертати список спільних елементів двох списків);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

3) Написати реалізацію класу для подання многокутників. Реалізація класу має містити:

a) методи Insert (добавлення вершини у задану позицію) та Remove (видалення вершини);
b) властивості Perimeter — периметр та Points — вершини многокутника;
c) операції * (множення координат усіх вершин на задане число), + (зсув многокутника на заданий вектор) та "==".
d) підтримку інтерфейсів ICloneable, IEnumerable та IEquatable.

4) Написати клас для реалізації векторів 3-вимірного простору з дійсними коефіцієнтами. Реалізація має містити:
   a) метод Length обчислення довжини вектора та статичну функцію знаходження відстані між векторами;
   b) властивості для звертання до координат вектора та властивості IsInteger та IsPositive.
   c) операції над векторами (+, –, , | — паралельність, скалярний та векторний добутки);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable, IComparer та IComparable.

5) Написати параметризований клас для реалізації двонапрямленого списку (із зворотними зв'язками). Реалізація має містити:
   a) функції-члени для вставки елемента після і перед заданим елементом, видалення заданого елемента списку, пошуку елемента у списку;
   b) властивості Count, Head, Tail та індексатор;
   c) операції "==", "*" (має повертати список спільних елементів двох списків) та "-" (має повертати список елементів першого списку, які не належать другому списку);
   d) підтримку інтерфейсів ICloneable, ICollection, IEquatable.

6) Написати клас для подання числової прямокутної матриці. Реалізація має містити:
   a) метод для обчислення норми матриці та метод для множення матриці на вектор;
   b) індексатор та властивості RowCount, ColumnCount;
   c) операції над матрицями (множення на число, додавання, віднімання, множення, порівняння);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

7) Написати реалізацію класу "прямокутний паралелепіпед". Реалізація має містити:
   a) статичний метод Surface (обчислення площі поверхні) та метод Translate (паралельне перенесення поточного паралелепіпеда на заданий вектор);
   b) властивості Vertices (вершини), Volume (об'єм) та Center (центр);
   c) підтримку інтерфейсів ICloneable та IEnumerable;
   d) операції "==", ">" та "*" (множення на число).

8) Написати узагальнений клас із двома параметрами для реалізації асоціативного масиву. Реалізація має містити:
   a) методи пошуку, добавлення та видалення елементів;
   b) властивості Count, Keys, Values та індексатор;
   c) операції "==" та "<=" (має повертати true, якщо усі пари ключ-значення першого асоціативного масиву належать другому масиву);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

9) Написати клас для реалізації бінарних відношень на множині $\{1,2,\ldots, N\}$. Реалізація має містити:
   a) статичний метод Inverse (має повертати обернене відношення), методи Add (для додавання пари елементів, які перебувають у відношенні), Remove (видалення пари елементів, які перебувають у відношенні), Clear та метод Contains (перевірка входження пари у бінарне відношення);
   b) властивості N, Count (кількість пар, які перебувають у відношенні), IsIdentity (є відношенням ідентичності) та індексатор;
   c) операції + (об'єднання), – (різниця) та * (добуток відношень).
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

10) Реалізувати узагальнений клас, що реалізує масив довільної кількості вимірів. Параметром класу є тип елементів масиву. Реалізація має містити:
   a) методи CopyTo та IndicesOf (повертає масив індексів);
   b) індексатор та властивість Shape, значенням якої є кортеж розмірностей масиву по кожному виміру;
   c) операції: ==, !=, >, <;
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

11) Написати реалізацію класу "булева функція". Реалізація має містити:
   a) метод GetNumber (має повертати номер функції) та GetDual (має повертати двоїсту функцію);
   b) індексатор, властивості IdenticallyTrue (загальнозначуща), IdenticallyFalse (тотожно фальшива);
   c) операції !, &, |, ^ та ==;
   d) підтримку інтерфейсів IEquatable, ICloneable, IEnumerable.

12) Написати узагальнений клас, який реалізує множину елементів заданого типу. Реалізація має містити:
   a) методи для добавлення, вилучення та пошуку елементів;
   b) властивості Count та IsSingletonSet (містить один елемент);
   c) операції + (об'єднання), * (перетин) та − (різниця);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

13) Написати реалізацію класу "еліпс". Вважати, що осі еліпсі паралельні осям координат. Реалізація має містити:
   a) методи Area — обчислення площі еліпса та Contains — перевірка належності точки фігурі, яка обмежується еліпсом;
   b) властивості Center, A, B, Eccentricity;
   c) операції + (зсув еліпса на заданий вектор), == та * (масштабування еліпса);
   d) підтримку інтерфейсів IEquatable, ICloneable.

14) Написати клас для реалізації векторів *n*-вимірного простору з дійсними коефіцієнтами. Реалізація має містити:
   a) статичний метод знаходження відстані між векторами та метод ToArray();
   b) індексатор та властивість Length;
   c) операції "==", "+", "–", "*" (скалярний добуток векторів).
   d) підтримку інтерфейсів ICloneable, IEnumerable, IComparable.

15) Написати узагальнений клас для реалізації впорядкованого списку. Реалізація має містити:
   a) функції-члени для вставки елемента у список, видалення елемента із списку, IndexOf (знаходження позиції першого входження елемента);
   b) властивості Count, Min, Max та індексатор;
   c) операції ==, ^ (симетрична різниця двох списків), <=;
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

16) Написати клас для реалізації точок у 2-вимірному просторі. Реалізація має містити:
   a) функцію, яка повертає номер чверті, у якій лежить точка, GetSymmetric (повертає точку, симетричну відносно заданої координатної осі чи точки) та статичну функцію знаходження відстані між точками;
   b) властивості для звертання до координат точки та властивості OnAxes (перевірка того, чи лежить точка на осі координат);
   c) покоординатні операції над точками (+, –, *);
   d) підтримку інтерфейсів ICloneable, IComparable, IEquatable.

17) Написати реалізацію класу "трикутник". Реалізація класу має містити:
   a) статичний метод Area — знаходження площі та метод Interior — перевірка того, чи лежить точка усередині трикутника;
   b) властивості Vertices (вершини трикутника), IsIsosceles та IsRightAngled (перевірка, чи є трикутник рівнобедреним чи прямокутним);
   c) операції "+" — паралельний перенос трикутника на заданий вектор, "==" та "!=";
   d) підтримку інтерфейсів ICloneable та IEquatable.

18) Написати реалізацію класу "ламана лінія". Реалізація має містити:
   a) методи Length (обчислення довжини), PassThrough (перевірка проходження через точку);
   b) властивості Vertices — вершини ламаної та SelfCrossing — перевірка наявності самоперетинів;
   c) операцію + (паралельний перенос ламаної на заданий вектор), * (множення усіх координат на задане число) та "==";
   d) підтримку інтерфейсів IEquatable, ICloneable та IEnumerable.

19) Написати реалізацію класу "Номер у готелі*". Реалізація має містити:
   a) методи MakeEmpty (виселити усіх жильців з номеру), Contains (перевірка того, що особа проживає у номері);
   b) властивості Count — поточна кількість людей у номері, Capacity — максимальна місткість номера, Dwellers (список імен мешканців кімнати);
   c) операції "+" та "−" (поселення та виселення жильця у номер);
   d) підтримку інтерфейсів IEquatable, ICloneable та IEnumerable.

20) Написати реалізацію класу "відрізок". Реалізація має містити:
   a) методи Length (повертає довжину відрізка), Translate (зсув на заданий вектор) та статичний метод Parallels — перевірка паралельності відрізків.
   b) Властивості EndPoints (кінці) та Intersects (перевірка того, чи перетинає відрізок координатні осі);
   c) операції == та <= (вважати, що коротший відрізок є "більшим");
   d) підтримку інтерфейсів IEquatable, ICloneable та IComparable.

21) Написати реалізацію класу "Автобус*". Реалізація має містити:
   a) методи посадки та висадки одного або кількох пасажирів, методи Accelerate та Brake (збільшення та зменшення швидкості). Реалізація має враховувати MaxSpeed та Capacity;
   b) властивості Speed (швидкість), Capacity (максимальна кількість пасажирів), MaxSpeed, HasEmptySeats (є вільні місця), Passengers (список імен пасажирів);
   c) операції "+" та "−" (посадка та висадка пасажира із заданим прізвищем);
   d) підтримку інтерфейсів ICloneable, IEquatable.

22) Написати параметризований клас з двома параметрами для реалізації упорядкованих бінарних дерев, у вершинах яких зберігаються елементи заданого типу (один параметр відповідає типу ключів вершин, другий — типу елементів). Реалізація має містити:
   a) методи вставки та видалення вершин з дерева та метод пошуку у дереві;
   b) властивості Count та Root (значення у вершині);
   c) операції ==, <= (приймає значення true, якщо 1-ий аргумент є піддеревом 2-го аргументу);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

23) Написати реалізацію класу "коло". Реалізація має містити:
   a) статичний метод Intersect (перевірка того, що два кола перетинаються), методи Length — обчислення довжини кола та Contains — перевірка належності точки кругу, який обмежується колом;
   b) властивості Center, Radius, CompletelyInSameQuarter (перевірка того, що коло цілком лежить у якійсь одній чверті);
   c) операції + (зсув кола на заданий вектор), == та * (розтяг/стиснення кола);
   d) підтримку інтерфейсів IEquatable, ICloneable та IComparable (вважати, що коло є "більшим" за усі кола, які у ньому містяться).

24) Написати клас "Перестановка*" для реалізації перестановок на множині $\{1,2,\ldots, N\}$. Реалізація має містити:
   a) методи InvariantElements (повертає масив елементів, які залишаються на місті), GetCycles (повертає розклад перестановки у добуток циклів);
   b) властивості *N* та IsEven (перевірка парності перестановки) та індексатор;
   c) операції "−" (знаходження оберненої перестановки), "*" (добуток перестановок).
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

25) Написати клас "наддовге ціле число" для реалізації цілого числа з довільною (як завгодно великою) кількістю цифр (при виконанні завдання не можна використовувати бібліотечний клас BigInteger). Реалізація має містити:
   a) методи ToDouble та Parse;
   b) властивості DigitCount та IsEven.
   c) операції додавання, віднімання, множення, операцію неявного перетворення із типу int у тип "наддовге ціле число";
   d) підтримку інтерфейсів ICloneable, IEquatable і IComparable.

26) Написати узагальнений клас, який реалізує мультимножину елементів заданого типу (елементи можуть повторюватися кілька разів). Реалізація має містити:
   a) методи для добавлення і вилучення елементів та метод, який повертає кількість входжень заданого елемента;
   b) властивості Count та Set (набір різних елементів, які входять у мультимножину);
   c) операції + (об’єднання), * (перетин) та − (різниця);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

27) Написати клас "квадратний тричлен". Реалізація має містити:
   a) метод обчислення значення тричлена у точці та метод GetRoots();
   b) властивості Vertex (вершина) та DeadSquare (перевірка того, що тричлен є точним квадратом);
   c) операції додавання, віднімання і порівняння тричленів;
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

28) Написати клас "пряма". Реалізація має містити:
   a) статичний метод перевірки паралельності прямих, метод знаходження точки перетину прямих, метод обчислення відстані від точки до прямої;
   b) властивості IsVertical та IsHorizontal;
   c) операцію порівняння прямих;
   d) підтримку інтерфейсів ICloneable, IEquatable.

29) Написати узагальнений клас, який реалізує впорядковану множину елементів заданого типу. Реалізація має містити:
   a) методи для добавлення, вилучення та пошуку елементів;
   b) властивість Count;
   c) операції + (об'єднання), * (перетин) та ^ (симетрична різниця);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

30) Написати реалізацію класу "промінь". Реалізація має містити:
   a) статичний метод перевірки паралельності променів;
   b) властивості Vertex (вершина променя), Cross0X та Cross0Y (перевірка того, чи перетинає промінь відповідні осі координат);
   c) операцію порівняння променів та операцію "+" (зсув вершини променя);
   d) підтримку інтерфейсів ICloneable, IEquatable.

31) Написати реалізацію класу "чотирикутник". Реалізація класу має містити:
   a) метод Perimeter — знаходження периметру та метод GetRect, який має повертати мінімальний прямокутник, який містить у собі чотирикутник;
   b) властивості Verices (вершини чотирикутника), IsParallelogram та IsRect (перевірка, чи є чотирикутник паралелограмом чи прямокутником);
   c) Операції "+" — паралельний перенос чотирикутника на заданий вектор, "==" та "!=";
   d) підтримку інтерфейсів ICloneable та IEquatable.

32) Написати реалізацію класу "предикат". Реалізація має містити:
   a) метод TruthSet (має повертати множину істинності предиката) та GetBasicSet(int id) (повертає відповідну базисну множину предиката);
   b) властивості IdenticallyTrue (загальнозначущий), IdenticallyFalse (тотожно фальшивий) та Homogeneous (однорідний);
   c) операції !, &&, ||, та ^;
   d) підтримку інтерфейсів IEquatable, ICloneable, IEnumerable.

33) Написати узагальнений клас із двома параметрами для реалізації асоціативного масиву, пари елементів якого упорядковані за ключем. Реалізація має містити:
   a) методи пошуку, добавлення та видалення елементів;
   b) властивості Count, Keys, Values та індексатор;
   c) операції "==" та "&&" (має повертати перетин двох асоціативних масивів, який містить лише ті пари "ключ-значення", що входять у обидва аргументи);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IComparable.

34) Написати реалізацію класу "книжкова полиця*" (попередньо написати клас книга). Реалізація має містити:
   a) методи для додавання, пошуку та видалення книг із полиці;
   b) властивості Count, Authors (автори книг) та Titles (назви книг);
   c) операції ==, + (результат — полиця, які містить усі книги полиць аргументів), <= (усі книги з першої полиці містяться на другій полиці);
   d) підтримку інтерфейсів IEquatable, ICloneable, IEnumerable.

35) Написати реалізацію класу "рівнобічна трапеція". Реалізація має містити:
   a) методи Length — обчислення периметру та Area — обчислення площі;
   b) властивості для довжин основ та бічної сторони (задання сторін не має порушувати нерівності: будь-яка сторона менша за суму інших сторін);
   c) операції == та * (розтяг/стиснення трапеції);
   d) підтримку інтерфейсів IEquatable, ICloneable та IComparable (вважати, що трапеція є "більшою" за усі трапеції з меншою площею).

36) Написати реалізацію класу "клієнт банку*" (попередньо написати клас "банківська операція"). Реалізація має містити:
   a) методи для виконання операцій зарахування та зняття (з перевіркою допустимості виконання операції) коштів з рахунку;
   b) властивості Name (ім’я клієнта), Account, Balance (поточний баланс), Operations (список операцій);
   c) операції > (порівняння балансів), + (злиття двох рахунків того самого клієнта);
   d) підтримку інтерфейсів IEquatable, ICloneable, IEnumerable (ітерація по операціям клієнта).

37) Написати реалізацію класу "магазин*" (попередньо написати клас товар). Реалізація має містити:
   a) методи для додавання, пошуку та видалення товарів у магазин і метод, який повертає загальну вартість усіх товарів;
   b) властивості Count, Prices (ціни) та Names (назви товарів);
   c) операції ==, % (збільшення ціни на заданий відсоток);
   d) підтримку інтерфейсів IEquatable, ICloneable, IEnumerable.

38) Написати реалізацію узагальненого класу "черга".
Реалізація має містити:
   a) методи для добавлення (у кінець черги), видалення (з початку черги) та пошуку елемента черги;
   b) властивості Count, Head, Tail (останні дві властивості відповідають за елементи, який розташовані на початку та у кінці черги відповідно);
   c) операції == та + (конкатенація черг);
   d) підтримку інтерфейсів IEquatable, ICloneable та IEnumerable.

39) Написати реалізацію класу "офіс*" (попередньо написати клас "працівник" із полями код, прізвище, дата народження, посада, безпосередній керівник). Реалізація має містити:
   a) методи для додавання, пошуку та видалення працівника і метод GetSubordinates, який має повертати список підпорядкованих працівників;
   b) властивості Count, Employees;
   c) операції ==, + та − (включення або виключення працівника із штату);
   d) підтримку інтерфейсів IEquatable, ICloneable, IEnumerable.

40) Написати реалізацію класу "булевий вектор". Реалізація має містити:
   a) статичний метод Distance для знаходження відстані Хеммінга між двома булевими векторами та методи RotateLeft, RotateRight (циклічні зсуви);
   b) індексатор, властивості Dim та Norm (норма Хеммінга);
   c) операції !, &&, ||, ^, <<, та >>;
   d) підтримку інтерфейсів IEquatable, ICloneable, IEnumerable.

41) Написати клас для реалізації діапазону цілих чисел. Реалізація має містити:
   a) Функції-члени Contains, Increase (збільшення кінця діапазону на задане значення), статичну функцію Parse (передбачити можливість включати / виключати кінці діапазону за допомогою символів дужок "[" та "(" відповідно);
   b) властивості From, To, Length та Step;
   c) операції ==, + (має повертати зсунутий діапазон, до початку та кінця якого додані відповідні значення), * (має повертати масштабований діапазон, початок та кінець якого помножені на числовий множник);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

42) Написати узагальнений клас Histogram, який реалізує гістограму (полігон частот) елементів заданого типу. Реалізація має містити:
   a) методи для добавлення і вилучення потрібного числа екземплярів заданого елемента у гістограму та метод, який повертає елементи максимальної частоти;
   b) властивості Count і Set (набір різних елементів, які входять у мультимножину) та індексатор;
   c) операції + (об’єднання), * (перетин) та − (різниця);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

43) Написати реалізацію узагальненого класу "черга з пріоритетом".
Реалізація має містити:
   a) методи для добавлення, видалення (елемента з найбільшим пріоритетом з початку черги) та пошуку елемента черги;
   b) властивості Count, Max, Min (останні дві властивості відповідають за елементи, який розташовані на початку та у кінці черги відповідно);
   c) операції == та + (конкатенація черг);
   d) підтримку інтерфейсів IEquatable, ICloneable та IEnumerable.

44) Написати реалізацію класу "коло" та його нащадка — класу "циліндр". Реалізація останнього класу має містити визначення циліндра у 3-вимірному координатному просторі, вісь якого паралельна осі OZ. Передбачити:
   a) функції-члени SurfaceArea, Volume, Contains (перевірка належності точки циліндру);
   b) властивості Radius, Height та індексатор (має надавати доступ до центрів основ циліндра);
   c) операції "==", "+" (зсув циліндра на заданий вектор), "*" (має повертати циліндр з масштабованим радіусом), < (має повертати True, якщо 1-й циліндр міститься усередині 2-го);
   d) підтримку інтерфейсів ICloneable, IEquatable, IComparable (вважати, що більшим є циліндр більшого об’єму).

45) Написати реалізацію класу "Маршрутне таксі*". Реалізація має містити:
   a) методи посадки та висадки кількох пасажирів, методи ChangeSpeed (збільшення чи зменшення швидкості). Реалізація має враховувати MaxSpeed та Capacity;
   b) властивості Speed (швидкість), Capacity (максимальна кількість пасажирів), MaxSpeed, IsFull (усі місця зайняті), Seats (словник, ключами якого є номери місць, а значеннями — імена пасажирів);
   c) операції "+" та "−" (посадка та висадка пасажира із заданим іменем);
   d) підтримку інтерфейсів ICloneable, IEquatable, IEnumerable.

46) Написати реалізацію класу "коло" та його нащадка — класу "конус". Реалізація останнього класу має містити визначення геометричного тіла у 3-вимірному координатному просторі, вісь якого паралельна осі OZ. Передбачити:
   a) функції-члени SurfaceArea, Volume, Contains (перевірка належності точки конусу);
   b) властивості Radius, Height, SlantHeight та Vertex;
   c) операції "==", "+" (зсув на заданий вектор), "*" (має повертати конус з масштабованим радіусом), < (має повертати True, якщо 1-й конус має меншу площу бічної поверхні ніж 2-й);
   d) підтримку інтерфейсів ICloneable, IEquatable, IComparable (вважати, що більшим є конус більшого об’єму).

47) Написати параметризований (з двома параметрами) клас для реалізації упорядкованих бінарних дерев, у вершинах яких зберігаються елементи заданого типу. Реалізація має містити:
   a) методи вставки та видалення елементів з дерева та метод пошуку у дереві за ключем;
   b) властивості Count та Root (значення у вершині);
   c) операції ==, <= (приймає значення true, якщо 1-ий аргумент є піддеревом 2-го аргументу);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

48) Написати узагальнений клас із двома параметрами для реалізації асоціативного масиву. Реалізація має містити:
   a) методи пошуку, добавлення та видалення елементів;
   b) властивості Count, Keys, Values та індексатор;
   c) операції "==" та "<=" (має повертати true, якщо усі пари ключ-значення першого асоціативного масиву належать другому масиву);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

49) Написати реалізацію класу для подання многокутників. Реалізація класу має містити:
   a) методи Insert та Remove (добавлення та видалення вершин у задану позицію);
   b) властивості Perimeter — периметр та Points — вершини многокутника;
   c) операції * (множення координат усіх вершин на задане число), + (зсув многокутника на заданий вектор) та "==".
   d) підтримку інтерфейсів IEnumerable та IEquatable.

50) Написати реалізацію класу "прямокутний паралелепіпед". Реалізація має містити:
   a) методи Surface (обчислення площі поверхні) та Translate (паралельний перенос на заданий вектор);
   b) властивості Vertices (вершини) та Volume (об’єм);
   c) підтримку інтерфейсів ICloneable та IEnumerable;
   d) операції "==", ">" та "*" (множення на число).

51) Написати клас для реалізації бінарних відношень на множині {1,2,…, N}. Реалізація має містити:
   a) статичний метод Inverse (має повертати обернене відношення), метод Add (для додавання пари елементів, які перебувають у відношенні), метод Remove (видалення пари елементів, які перебувають у відношенні) та метод Contains (перевірка входження пари у бінарне відношення);
   b) властивості Count (кількість пар, які перебувають у відношенні), N та індексатор;
   c) операції + (об'єднання), – (різниця) та * (добуток відношень).
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

52) Реалізувати узагальнений клас, що реалізує масив довільної кількості вимірів. Параметром класу є тип елементів масиву. Реалізація має містити:
   a) методи CopyTo та IndicesOf (повертає масив індексів);
   b) індексатор та властивість DimCount, значенням якої є кількість вимірів масиву;
   c) операції: ==, !=, >, <;
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

53) Написати узагальнений клас, який реалізує множину елементів заданого типу. Реалізація має містити:
   a) методи для додавання, вилучення та пошуку елементів;
   b) властивість Count;
   c) операції + (об'єднання), * (перетин) та − (різниця);
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

54) Написати реалізацію класу "еліпс". Вважати, що осі еліпсі паралельні осям координат. Реалізація має містити:
   a) методи Area — обчислення площі еліпса та Contains — перевірка належності точки фігурі, яка обмежується еліпсом;
   b) властивості Center, A, B, Eccentricity;
   c) операції + (зсув еліпса на заданий вектор), == та * (розтяг/стиснення еліпса);
   d) підтримку інтерфейсів IEquatable, ICloneable.

55) Написати клас для реалізації векторів n-вимірного простору з дійсними коефіцієнтами. Реалізація має містити:
   a) статичний метод знаходження відстані між векторами та метод ToArray();
   b) індексатор та властивість Length;
   c) операції "==", "+", "–", "*" (скалярний добуток векторів).
   d) підтримку інтерфейсів ICloneable, IEnumerable, IComparable.

56) Написати узагальнений клас для реалізації впорядкованого списку. Реалізація має містити:
   a) функції-члени для вставки елемента у список, видалення елемента із списку, IndexOf (знаходження позиції першого входження елемента);
   b) властивості Count, Min, Max та індексатор;
   c) операції ==, ^ (симетрична різниця двох списків), <=;
   d) підтримку інтерфейсів ICloneable, IEnumerable, IEquatable.

57) Написати клас для реалізації точок у 2-вимірному просторі. Реалізація має містити:
   a) функцію, яка повертає номер чверті, у якій лежить точка, та статичну функцію знаходження відстані між точками;
   b) властивості для звертання до координат точки та властивості OnAxes (перевірка того, чи лежить точка на осі координат);
   c) покоординатні операції над точками (+, –, *);
   d) підтримку інтерфейсів ICloneable, IComparable, IEquatable.

58) Написати реалізацію класу "трикутник". Реалізація класу має містити:
   a) статичний метод Area — знаходження площі та метод Interior — перевірка того, чи лежить точка усередині трикутника;
   b) властивості Verices (вершини трикутника), IsIsosceles та IsRightAngled (перевірка, чи є трикутник рівнобедреним чи прямокутним);
   c) Операції "+" — паралельний перенос трикутника на заданий вектор, "==" та "!=";
   d) підтримку інтерфейсів ICloneable та IEquatable.

59) Написати реалізацію класу "ламана лінія". Реалізація має містити:
   a) методи Length (обчислення довжини), PassThrough (перевірка проходження через точку);
   b) властивості Vertices — вершини ламаної та SelfCrossing — перевірка наявності само перетинів;
   c) операцію + (паралельний перенос ламаної на заданий вектор), * (множення усіх координат на задане число) та "==";
   d) підтримку інтерфейсів IEquatable, ICloneable та IEnumerable.

60) Написати реалізацію класу "Номер у готелі*". Реалізація має містити:
   a) методи MakeEmpty (виселити усіх жильців з номеру), Contains (перевірка того, що особа проживає у номері);
   b) властивості Count — поточна кількість людей у номері, Capacity — максимальна місткість номера, Dwellers (список імен мешканців кімнати);
   c) операції "+" та "−" (поселення та виселення жильця у номер);
   d) підтримку інтерфейсів IEquatable, ICloneable та IEnumerable.