**Лабораторна робота №5**

**Тема:** Успадкування в С++ та С#.

**Мета роботи:** Набуття навичок розробки програм мовами С++ та С# з використанням базового  принципу об’єктно-орієнтованого програмування – (одиночного) успадкування.

**Теоретичні відомості:** матеріали лекції.

***Завдання на лабораторну роботу***

|  |
| --- |
| *Варіант 1.* |
| 1. Описати клас, який містять вказані поля і методи.  **Клас** “Вектор у 2 *R*” – TVector2D  **поля** ▪ для зберігання координат вектора;  **методи** ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення елементів вектора;  ▪ визначення довжини вектора;  ▪ нормування вектора;  ▪ порівняння з іншим вектором;  ▪ перевантаження операторів + (додавання векторів), – (віднімання  векторів), \* (знаходження скалярного добутку). |
| 2. Створити клас-нащадок TVector3D (вектор у 3 *R*) на основі класу TVector2D. Додати третю  координату вектора та перевизначити відповідні методи. |
| 3. Створити програму-клієнт для тестування*.* |

|  |
| --- |
| *Варіант 2.* |
| 1. Описати клас, який містять вказані поля і методи.  **Клас** “Прямокутник ” – TRectangle  **поля** ▪ для зберігання довжин сторін;  **методи** ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ визначення площі;  ▪ визначення периметру;  ▪ порівняння з іншим прямокутником;  ▪ перевантаження операторів + (додавання відповідних сторін), – (віднімання довжин відповідних сторін), \* (множення сторін на деяке число). |
| 2. Створити клас-нащадок TParallelepiped (прямокутний паралелепіпед) на основі класу TRectangle. Додати поле для збереження висоти паралелепіпеда, метод знаходження об’єму  паралелепіпеда та перевизначити відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 3.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “Квадрат ” – TSquare | |
| **поля** | ▪ для зберігання довжини сторін; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ визначення площі;  ▪ визначення периметру;  ▪ порівняння з іншим квадратом;  ▪ перевантаження операторів + (додавання довжин сторін), – (віднімання  довжин сторін), \* (множення сторін на деяке число). |

|  |
| --- |
| 2. Створити клас-нащадок TCube (куб) на основі класу TSquare. Додати метод знаходження  об’єму куба та перевизначити відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 4.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “Трикутник ” – TTriangle | |
| **поля** | ▪ для зберігання довжин сторін; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами,  конструктор копіювання;  ▪ введення/виведення даних;  ▪ визначення площі;  ▪ визначення периметру;  ▪ порівняння з іншим трикутником;  ▪ перевантаження операторів + (додавання довжин сторін), – (віднімання довжин відповідних сторін), \* (множення сторін на деяке  число). |

|  |
| --- |
| 2. Створити клас-нащадок TTrianglePrizm (пряма призма, в основі якої трикутник) на  основі класу TTriangle. Додати метод знаходження об’єму призми та перевизначити  відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 5.* |
| 1. Описати клас, який містять вказані поля і методи.  **Клас** “Рівносторонній трикутник ” – TRTriangle  **поля** ▪ для зберігання довжини сторін;  **методи** ▪ конструктор без параметрів, конструктор з параметрами,  конструктор копіювання;  ▪ введення/виведення даних;  ▪ визначення площі;  ▪ визначення периметру;  ▪ порівняння з іншим трикутником;  ▪ перевантаження операторів + (додавання довжин сторін), – (віднімання довжин відповідних сторін), \* (множення сторін на деяке  число). |
| 2. Створити клас-нащадок TPiramid (правильна трикутна піраміда) на основі класу TRTriangle. Додати метод знаходження об’єму піраміди та перевизначити відповідні  методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 6.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “прямокутний трикутник ” – TPTriangle | |
| **поля** | ▪ для зберігання довжин катетів; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами,  конструктор копіювання;  ▪ введення/виведення даних;  ▪ визначення площі;  ▪ визначення периметру;  ▪ порівняння з іншим трикутником;  ▪ перевантаження операторів + (додавання довжин катетів), – (віднімання довжин відповідних катетів), \* (множення сторін на  деяке число). |

|  |
| --- |
| 2. Створити клас-нащадок TPPiramid (прямокутна трикутна піраміда, у якій бічне ребро  перпендикулярне до катетів) на основі класу TPTriangle. Додати метод знаходження  об’єму піраміди та перевизначити відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 7.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “коло” – TCircle | |
| **поля** | ▪ для зберігання радіуса; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ визначення площі круга (обмеженого колом);  ▪ визначення довжини кола;  ▪ порівняння з іншим колом;  ▪ перевантаження операторів + (додавання радіусів), – (віднімання  радіусів), \* (множення радіуса на число). |

|  |
| --- |
| 2. Створити клас-нащадок TCylinder (циліндр) на основі класу TCircle. Додати метод  знаходження об’єму циліндра та перевизначити відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 8.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “коло” – TCircle | |
| **поля** | ▪ для зберігання радіуса; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ визначення площі круга (обмеженого колом), площі сектора; ▪ визначення довжини кола;  ▪ порівняння з іншим колом;  ▪ перевантаження операторів + (додавання радіусів), – (віднімання  радіусів), \* (множення радіуса на число). |

|  |
| --- |
| 2. Створити клас-нащадок TCone (конус) на основі класу TCircle. Додати поле для збереження  висоти конуса, метод знаходження об’єму конуса та перевизначити відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 9.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “коло” – TCircle | |
| **поля** | ▪ для зберігання радіуса; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами,  конструктор копіювання;  ▪ введення/виведення даних;  ▪ визначення площі круга (обмеженого колом), площі сектора; ▪ визначення довжини кола;  ▪ порівняння з іншим колом;  ▪ перевантаження операторів + (додавання радіусів), – (віднімання  радіусів), \* (множення радіуса на число). |

|  |
| --- |
| 2. Створити клас-нащадок TSphere (сфера) на основі класу TCircle. Додати метод  знаходження площі поверхні сфери та перевизначити відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 10.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “круг” – TCircle | |
| **поля** | ▪ для зберігання радіуса;  ▪ для зберігання центра кола |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ визначення площі круга;  ▪ перевірка належності точки кругу;  ▪ перевантаження операторів + (додавання радіусів), – (віднімання  радіусів), \* (множення радіуса на число). |

|  |
| --- |
| 2. Створити клас-нащадок TBall (куля) на основі класу TCircle. Додати метод знаходження об’єму  кулі та перевизначити відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 11.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “одновимірний масив” – TArray | |
| **поля** | ▪ для зберігання елементів масиву;  ▪ для зберігання кількості елементів. |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ знаходження найбільшого/найменшого елемента;  ▪ сортування масиву;  ▪ знаходження суми елементів;  ▪ перевантаження операторів + (додавання елементів), – (віднімання  елементів), \* (множення масиву на число). |

|  |
| --- |
| 2. Створити клас-нащадок TOderedArray (упорядкований масив) на основі класу TArray. Додати  методи додавання та вилучення елементів (перевизначивши оператори додавання та віднімання  числа). |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 12.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “матриця ” – TMatrix | |
| **поля** | ▪ для зберігання елементів матриці;  ▪ для зберігання розмірності матриці. |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ знаходження найбільшого/найменшого елемента;  ▪ знаходження суми елементів. |

|  |
| --- |
| 2. Створити клас-нащадок TOpMatrix (матриця, для якої перевантажено оператори +,–,\*) на основі  класу TMatrix.. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 13.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “квадратна матриця ” – TSMatrix | |
| **поля** | ▪ для зберігання елементів матриці;  ▪ для зберігання розмірності матриці. |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ знаходження найбільшого/найменшого елемента;  ▪ знаходження суми елементів.  ▪ перевантаження операторів + (додавання елементів), – (віднімання  елементів). |

|  |
| --- |
| 2. Створити клас-нащадок TDeterminant2 (визначник квадратної матриці порядку 2) на основі  класу TSMatrix. Розробити метод для знаходження визначника. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 14.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “квадратна матриця ” – TSMatrix | |
| **поля** | ▪ для зберігання елементів матриці;  ▪ для зберігання розмірності матриці. |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ знаходження найбільшого/найменшого елемента;  ▪ знаходження суми елементів.  ▪ перевантаження операторів + (додавання матриць), – (віднімання  матриць). |

|  |
| --- |
| 2. Створити клас-нащадок TMSMatrix (клас доповнюється новими методами: транспонування,  перевантаження оператора \* (множення матриці на матрицю) та перевантаження оператора \*  (множення матриці на число) ) на основі класу TSMatrix. Розробити метод для знаходження  визначника. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 15.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “звичайний дріб” – TFraction | |
| **поля** | ▪ для зберігання чисельника і знаменника; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ скорочення дробів (якщо чисельник і знаменник містять спільні  множники);  ▪ перевантаження операторів +, –, \*, / . |

|  |
| --- |
| 2. Створити клас-нащадок TMixFraction (мішані дроби – ціла частина, чисельник і знаменник) на  основі класу TFraction. Додати поле для збереження цілої частини та перевизначити відповідні  методи. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 16.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “комплексне число” – TComplex | |
| **поля** | ▪ для зберігання дійсної і уявної частин; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ перевантаження операторів +, –, \*, / . |

|  |
| --- |
| 2. Створити клас-нащадок TMComplex (комплексне число на площині) на основі класу TComplex. Додати методи визначення квадранта, у який попадає комплексне число, метод  визначення відстані до початку координат. |
| 3. Створити програму-клієнт для тестування. |

|  |
| --- |
| *Варіант 17.* |

1. Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “пряма на площині” – TLine2D | |
| **поля** | ▪ для зберігання коефіцієнтів канонічного рівняння прямої; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами, конструктор  копіювання;  ▪ введення/виведення даних;  ▪ знаходження точки перетину з іншою прямою;  ▪ визначення належності точки прямій;  ▪ перевантаження операторів + (додавання коефіцієнтів прямих), – (віднімання коефіцієнтів прямих). |

|  |
| --- |
| 2. Створити клас-нащадок TLine3D (пряма у просторі) на основі класу TLine2D. Додати ще одне  поле для збереження коефіцієнта при невідомій z та перевизначити відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

*Варіант 18.* Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “відрізок на площині” – TInterval2D | |
| **поля** | ▪ для зберігання координат початку і кінця відрізка; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами,  конструктор копіювання;  ▪ введення/виведення даних;  ▪ знаходження точки перетину з іншим відрізком;  ▪ визначення довжини відрізка;  ▪ визначення середини відрізка;  ▪ перевантаження операторів + (утворюється відрізок початок  якого є початком першого, а кінець – кінцем другого), \* (збільшення  довжини відрізка у вказану кількість разів зберігаючи початок  відрізка незмінним). |

|  |
| --- |
| 2. Створити клас-нащадок TInterval3D (відрізок у просторі) на основі класу TInterval2D.  Додати поля для збереження третьої координати точок початку і кінця та перевизначити  відповідні методи. |
| 3. Створити програму-клієнт для тестування. |

*Варіант 19.* Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “точка на площині” – TPoint2D | |
| **поля** | ▪ для зберігання довжин сторін; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами,  конструктор копіювання;  ▪ введення/виведення даних;  ▪ визначення відстані до іншої точки;  ▪ визначення квадранту, до якого належить точка;  ▪ порівняння з іншою точкою;  ▪ перевантаження операторів + (додавання відповідних координат),  – (віднімання відповідних координат). |

|  |
| --- |
| 2. На основі класу TPoint2D створити клас TCRectangle (прямокутник задається  координатами вершин). Клас повинен містити чотири поля типу TPoint2D і методи для  знаходження площі, периметру прямокутника та методу визначення того, чи лежить  вказана точка всередині прямокутника. |
| 3. Створити програму-клієнт для тестування. |

*Варіант 20.* Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “точка на площині” – TPoint2D | |
| **поля** | ▪ для зберігання координат; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами,  конструктор копіювання;  ▪ введення/виведення даних;  ▪ визначення відстані до іншої точки;  ▪ визначення квадранту, до якого належить точка;  ▪ порівняння з іншою точкою;  ▪ перевантаження операторів + (додавання відповідних координат),  – (віднімання відповідних координат). |

|  |
| --- |
| 2. На основі класу TPoint2D створити клас TCTriangle (трикутник задається  координатами вершин). Клас повинен містити три поля типу TPoint2D і методи для  знаходження площі, периметру трикутника та методу визначення того, чи лежить вказана  точка всередині трикутника. |
| 3. Створити програму-клієнт для тестування. |

*Варіант 21.* Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “арифметична прогресія” – TArProgression | |
| **поля** | ▪ для зберігання першого члена арифметичної прогресії та різниці; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами,  конструктор копіювання;  ▪ введення/виведення даних;  ▪ знаходження *n* -го члена;  ▪ знаходження суми *n*перших членів прогресії;  ▪ перевантаження операторів + (додавання перших членів та  різниць), – (віднімання перших членів та різниць). |

|  |
| --- |
| 2. На основі класу TArProgression створити клас-нащадок TArProgressionM. Цей клас  повинен містити метод, який дозволяє визначити, чи утворює послідовність цілих чисел  арифметичну прогресію та метод, який дозволяє встановити, чи є вказане число членом  даної арифметичної прогресії. |
| 3. Створити програму-клієнт для тестування. |

*Варіант 22.* Описати клас, який містять вказані поля і методи.

|  |  |
| --- | --- |
| **Клас** “геометрична прогресія” – TGeomProgression | |
| **поля** | ▪ для зберігання першого члена геометричної прогресії та  знаменника; |
| **методи** | ▪ конструктор без параметрів, конструктор з параметрами,  конструктор копіювання;  ▪ введення/виведення даних;  ▪ знаходження *n* -го члена;  ▪ знаходження суми *n*перших членів прогресії;  ▪ перевантаження операторів + (додавання перших членів та  знаменників), – (віднімання перших членів та знаменників). |

|  |
| --- |
| 2. На основі класу TGeomProgression створити клас-нащадок TGeomProgressionM. Цей  клас повинен містити метод, який дозволяє визначити, чи утворює послідовність додатних  цілих чисел геометричну прогресію та метод, який дозволяє встановити, чи є вказане  число членом даної геометричної прогресії. |
| 3. Створити програму-клієнт для тестування. |

**Завдання на лабораторну роботу**

Створити консольну програму, що реалізує основні операції з одно-, дво- та *n*-вимірними  масивами: пошук, вставку, видалення, сортування заданим алгоритмом, заповнення за заданим  алгоритмом, трансформацію за заданим алгоритмом, копіювання в інший масив елементів, що  задовольняють певному критерію.

*Зауваження*: алгоритми заповнення і трансформації масиву, критерії елементів щодо  копіювання задаються викладачем при захисті роботи.

***Завдання для самостійної роботи***

У кожному із завдань для тестування створити програму-клієнт.

1. Створити клас, що містить методи знаходження площі, периметра та типу опуклого  чотирикутника, який може задаватися довжинами сторін або координатами вершин.  Створити відповідні інтерфейси для випадку завдання чотирикутника довжинами сторін та  координатами вершин.

2. Створити клас, що містить методи знаходження площі, периметра, радіуса вписаного і  описаного кола для рівностороннього, рівнобедреного, різностороннього та прямокутного  трикутників. Для різного типу трикутників передбачити відповідні інтерфейси.

3. Створити клас, що містить методи визначення паралельності та перпендикулярності  прямих на площині, що задані різними способами. Для різних способів задання прямих  передбачити відповідні інтерфейси.

4. Створити клас, що містить методи знаходження розв’язків системи двох лінійних рівнянь з  двома невідомими та системи трьох лінійних рівнянь з трьома невідомими (коефіцієнти

6

можуть бути задані як десяткові дробів або як раціональні дроби). Для системи двох  лінійних рівнянь з двома невідомими та системи трьох лінійних рівнянь з трьома  невідомими передбачити відповідні інтерфейси.

5. Створити клас, що містить методи розв’язання квадратного рівняння (знаходження  розв’язків розглядати як у *R*, так і просторі комплексних чисел). Для різних випадків  передбачити відповідні інтерфейси.

6. Створити клас, що містить методи додавання, віднімання, множення та ділення  раціональних дробів та такі ж методи для роботи з комплексними числами. Для випадку  раціональних дробів та випадку комплексних чисел передбачити відповідні інтерфейси.

7. Створити клас, що містить методи для знаходження відсотку від числа, збільшення,  зменшення числа на певну кількість відсотків (число може бути десятковим або  раціональним дробом). При створенні класу використати відповідні інтерфейси.

8. Створити клас, що містить методи для знаходження суми і різниці чисел, які можуть бути  або раціональними дробами, або числами, записаними за допомогою римських цифр. При  створенні класу використати відповідні інтерфейси.

9. Створити клас, що містить методи для знаходження суми і різниці цифр, які можуть бути  арабськими, римськими або задані прописом. При створенні класу використати відповідні  інтерфейси.

10. Створити клас, що містить методи знаходження суми, добутку та середнього  арифметичного елементів масиву, який може бути статичним або динамічно створеним.  При створенні класу використати відповідні інтерфейси.

11. Створити інтерфейс, що містить опис методів для роботи з векторами (знаходження  довжини вектора, суми векторів, множення вектора на число, скалярного добутку) та класи,  які містять реалізації методів інтерфейсу у випадку двовимірних та тривимірних векторів.

12. Створити інтерфейс, що містить опис методів для роботи з двовимірними векторами  (знаходження довжини вектора, суми векторів, множення вектора на число, скалярного  добутку) та класи, які містять реалізації методів інтерфейсу у випадку, коли вектор задано  його координатами та координатами початку і кінця.

13. Створити інтерфейс, що містить опис методів для знаходження *n* -го члена та суми перших  *n*членів прогресії. На основі цього інтерфейсу створити класи, що містять реалізації  методів інтерфейсу у випадку арифметичної та геометричної прогресій.

14. Створити інтерфейс, що містить опис методів для знаходження суми цифр числа, а  визначення кількості нулів у записі числа. На основі цього інтерфейсу створити класи, що  містять реалізації методів інтерфейсу у випадку цілого та дійсного чисел

7

15. Створити інтерфейс, що містить опис методів додавання, вилучення та пошуку елементів,  що є цілими числами типу Byte. На основі цього інтерфейсу створити класи, що містять  реалізації методів інтерфейсу, зберігаючи елементи у множині та одновимірному масиві.

16. Створити інтерфейс, що містить опис методів додавання, вилучення та пошуку елементів,  що є цілими числами. На основі цього інтерфейсу створити класи, що містять реалізації  методів інтерфейсу, зберігаючи елементи у стеку та черзі.

17. Створити інтерфейс, що містить опис методів додавання, вилучення та пошуку дійсних  чисел, які зберігаються у файлі. На основі цього інтерфейсу створити класи, що містять  реалізації методів інтерфейсу, зберігаючи числа у текстовому та типізованому файлі  дійсних чисел.

18. Створити інтерфейс, що містить опис методів роботи з матрицями (знаходження  детермінанта, сліду, рангу матриці), та класи, що містять реалізації методів інтерфейсу у  випадку квадратних матриць порядку 2 і 3.

19. Створити інтерфейс, що містить опис методів знаходження суми, найбільшого,  найменшого елементів. На основі цього інтерфейсу створити класи, що містять реалізації  методів інтерфейсу, зберігаючи елементи у одновимірному та двовимірному масивах.

20. Створити інтерфейс, що містить опис методів визначення відстані від точки до площини в  3 *R*та метод перевірки належності точки деякій площині. На основі цього інтерфейсу  створити класи, що містять реалізації методів інтерфейсу у випадку задання площини  канонічним рівнянням та за допомогою трьох точок.

21. Створити інтерфейс, що містить опис методів знаходження площі поверхні, площі основи  та об’єму правильної піраміди. На основі цього інтерфейсу створити класи, що містять  реалізації методів інтерфейсу у випадку трикутної та чотирикутної пірамід.

***Варіанти завдань*:**

1. Написати програму, яка обчислює кількість голосних та приголосних літер у файлі. Вміст  текстового файлу заноситься до масиву символів. Кількість голосних та приголосних літер визначається проходом по масиву. Передбачити метод, вхідний параметр якого є масив  символів. Метод обчислює кількість голосних та приголосних букв. Завдання також виконати  за допомогою *колекції List<T>*.

2. Написати програму, яка обчислює кількість прописних літер у файлі. Вміст текстового файлу  заноситься до масиву символів. Кількість прописних літер визначається проходом по масиву.  Передбачити метод, вхідний параметр якого є масив символів. Метод обчислює кількість  прописних букв. Завдання також виконати за допомогою *колекції List<T>*.

3. Написати програму, яка обчислює кількість знаків пунктуації у файлі. Вміст текстового файлу  заноситься до масиву символів. Кількість знаків пунктуації визначається проходом по масиву.  Передбачити метод, вхідний параметр якого є масив символів. Метод обчислює кількість  знаків пунктуації. Завдання також виконати за допомогою *колекції List<T>*.

4. Написати програму, яка обчислює кількість слів у файлі. Вміст текстового файлу заноситься до  масиву символів. Кількість слів визначається проходом по масиву. Передбачити метод,  вхідний параметр якого є масив символів. Метод обчислює кількість слів. Завдання також виконати за допомогою *колекції List<T>*.

5. Написати програму, що реалізує множення двох матриць, заданих як двовимірні масиви. У  програмі передбачити два методи: метод множення матриць (на вхід дві матриці, значення, що

7

повертається – матриця), метод виводу матриці на екран. Завдання також виконати за  допомогою *колекцій LinkedList<LinkedList<T>>*.

6. Написати програму, що реалізує векторний добуток, заданих як одновимірні масиви. У  програмі передбачити два методи: метод векторного добутку (на вхід два вектори, значення,  що повертається – вектор), метод виводу векторного добутку на екран. Завдання також виконати за допомогою *колекцій LinkedList<LinkedList<T>>*.

7. Написати програму, що реалізує зовнішній векторний добуток, заданих як одновимірні масиви.  У програмі передбачити два методи: метод зовнішнього векторного добутку (на вхід два вектори, значення, що повертається – матриця), метод виводу зовнішнього векторного добутку  на екран. Завдання також виконати за допомогою *колекцій LinkedList<LinkedList<T>>*.

8. Написати програму, що з довільного двовимірного масиву, утвореного нулями та одиницями,  утворює два одновимірних масиви: один з нулів, другий – з одиниць, що містяться у вхідному  масиві. У програмі передбачити два методи: метод розділення (на вхід двовимірний масив,  значення, що повертається – два одновимірних масиви), метод виводу отриманих масивів на  екран. Завдання також виконати за допомогою *колекцій LinkedList<LinkedList<T>>*.

9. Написати програму, яка обчислює середню температуру протягом року. Створити двовимірний  рандомний масив *temperature*[12,30], в якому зберігається температура для кожного дня місяця  (передбачається, що в кожному місяці 30 днів). Згенерувати значення температури випадковим  чином. Для кожного місяця надрукувати середню температуру. Для цього написати метод,  який за масивом *temperature*[12,30] для кожного місяця обчислює середню температуру в  ньому, і як результат повертає масив середніх температур. Отриманий масив середніх  температур відсортувати за зростанням. Завдання також виконати з допомогою *класу Dictionary<TKey, TValue>*. Як ключі вибрати рядки – назви місяців, а як значення – масив значень температур по днях.