**МІНІСТЕРСТВО ОСВІТИ И НАУКИ УКРАЇНИ**

**ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ**

**УНІВЕРСИТЕТ імені І.І.МЕЧНИКОВА**

**Факультет математики, фізики та інформаційних технологій**

**Індивідуальне навчально-дослідне завдання**

з дисципліни

**«Програмування»**

на тему

**«Школа»**

**Варіант 12а**

студента 1 курсу

групи 2

спеціальності 123 «Комп’ютерна інженерія»

Коваля Дениса Олександровича

(Прізвище, ім’я та по батькові)

Захищено « » 202 р. з оцінкою

Викладач:

Антоненко О.С.

(ПIБ) (Підпис)

**Одеса – 2025**

**ЗМІСТ**

[1 ПОСТАНОВКА ЗАДАЧІ 3](#_Toc199079895)

[2 ПРОЕКТНЕ РІШЕННЯ 4](#_Toc199079896)

[2.1 Діаграма класів 4](#_Toc199079897)

[2.2 Опис структури класів 5](#_Toc199079898)

[ВИСНОВКИ 13](#_Toc199079899)

[3 ПОСИЛАННЯ НА РЕПОЗИТОРІЙ 14](#_Toc199079900)

1. **ПОСТАНОВКА ЗАДАЧІ**

Мета роботи – реалізувати ієрархію класів, запропоновану викладачем та

використати її у вирішенні прикладної задачі, вказаної в завданні до роботи.

При цьому потрібно використовувати основні принципи ООП: інкапсуляцію,

успадкування, поліморфізм. Перелік класів задається у варіанті завдання.

Необхідно реалізувати конструктори класів, поля та методи класу. Поля класу

мають бути закритими, для доступу до них мають бути описані методи класів.

Задається ієрархія класів щодо відношень успадкування та композиції. Там, де

потрібно, використовувати віртуальні методи, чисті віртуальні методи та

абстрактні класи для реалізації ідеї динамічного поліморфізму.

Використовувати, де необхідно, вбудовані класи мови С++, особливо клас string та шаблонний клас vector. При некоректних вхідних даних методів,

конструкторів, функцій генерувати виняткові ситуації, та обробляти їх за

допомогою блоків try-catch у функціях більш високого рівня.

Варіант 12а. Створення ієрархії класів на тему «Школа»

Створити класи: **«Людина»** (ПІБ, вік, зріст), **«Учень»** (клас, список оцінок), **«Учитель»** (додаткове поле: предмет, який викладає), **«Оцінка»** (вид: домашня, контрольна, самостійна, іспит, підсумкова; предмет).

Рекомендується оцінки учня зберігати у векторі об'єктів типу **«Оцінка»**. Для учителя зберігати вектор вказівників на учнів, які у нього навчаються.

Визначити метод – вивести підсумкову оцінку учня (враховувати поточні оцінки та оцінку за іспит, але тільки з цього предмета). Для кожного учителя визначити середню оцінку з тих, які він виставив. Для цього перебрати оцінки учнів, які навчаються у цього учителя, і вибрати тільки оцінки з предмета, який він викладає.

# 

# **2 ПРОЕКТНЕ РІШЕННЯ**

Для користувача доступні наступні операції:

1. Додати учня;

2. Додати вчителя;

3. Додати оцінку учневі;

4. Інформація про учня;

5. Інформація про вчителя;

6. Додати учня вчителю;

7. Розрахувати підсумкову оцінку;

8. Список усіх учнів;

9. Список усіх учителів.

## **2.1 Діаграма класів**

A screenshot of a computer

AI-generated content may be incorrect.

Рисунок 2.1 Діаграма класів

## 

## **2.2 Опис структури класів**

1. Керуючий клас Person

1. Конструктори:
   1. protected Person()  
      Конструктор за замовчуванням. Доступний лише для класів-нащадків.
   2. public Person(string persName, int persAge, double persHeight)  
      Конструктор з параметрами, призначений для ініціалізації полів класу відповідними значеннями.
2. Приватні поля:
   1. persName – повне ім’я особи
   2. persAge – вік особи
   3. persHeight – зріст особи
3. Методи:
   1. InputPersInfo – введення інформації про особу з консолі.
   2. PersonInfo – виведення детальної інформації про особу.
   3. GetPersName – повертає ім’я.
   4. GetPersAge – повертає вік.
   5. GetPersHeight – повертає зріст.

2. Клас Student (успадковує Person)

1. Конструктор

public Student(string persName, int persAge, double persHeight, string studentGrade)  
Ініціалізує новий об'єкт учня з вказаними ПІБ, віком, зростом і навчальним класом. Створює новий список оцінок.

public Student()  
Конструктор за замовчуванням. Викликає конструктор базового класу Person та ініціалізує порожній список оцінок.

1. Приватні поля:
   1. studentGrade – клас, у якому навчається учень
   2. marks – список оцінок учня.
2. Методи:
   1. AddMark – додає нову оцінку учню.
   2. PrintMarks – виводить усі оцінки учня.
   3. PersonInfo – перевизначений метод; виводить інформацію про

особу, клас та оцінки.

* 1. GetStudentName – повертає ім’я учня.
  2. GetStudentGrade – повертає клас.
  3. GetFinalMark – повертає підсумкову оцінку за вказаним предметом

(розрахунок з урахуванням іспиту та середнього значення поточних оцінок).

3. Клас SchoolMark

1. Конструктор
2. Приватні поля:
   1. subject – предмет, з якого виставлена оцінка.
   2. type – тип оцінки (Домашня, Контрольна, Самостійна, Іспит,

Підсумкова).

* 1. markValue – значення оцінки (від 0 до 100).
  2. MarkPrint – повертає детальну інформацію про оцінку.

4. Клас Teacher (успадковує Person)

1. Конструктори

public Teacher(string persName, int persAge, double persHeight, string subject)  
Конструктор з параметрами. Ініціалізує поля persName, persAge, persHeight, subject, створює порожній список учнів.

public Teacher()  
Конструктор без параметрів. Викликає конструктор базового класу та ініціалізує порожній список учнів.

1. Приватні поля:
   1. subject – предмет, який викладає вчитель.
   2. students – список учнів, що прикріплені до цього викладача.
2. Методи:
   1. InputPersInfo – перевизначений метод; вводить базову інформацію

та предмет.

* 1. PersonInfo – перевизначений метод; виводить базову інформацію та

предмет.

* 1. AddStudent – додає учня до списку учнів викладача.
  2. RemoveStudent – видаляє учня зі списку.
  3. PrintStudents – друкує список учнів.
  4. GetStudents – повертає список учнів.
  5. GetSubject – повертає назву предмету.

5. Клас MenuController

1. Приватні поля:
   1. students – список усіх учнів.
   2. teachers – список усіх викладачів.
2. Методи:
   1. Run – головна функція програми, реалізує консольний інтерфейс з

10 пунктами меню.

* 1. AddStudent – додає нового учня.
  2. AddTeacher – додає нового викладача.
  3. AddMarkToStudent – додає оцінку учневі за вказаним предметом.
  4. ShowStudentInfo – виводить детальну інформацію про обраного

учня.

* 1. ShowTeacherInfo – виводить інформацію про викладача та

прикріплених учнів.

* 1. AssignStudentToTeacher – прикріплює учня до викладача.
  2. CalculateFinalMark – обчислює підсумкову оцінку учня за

предметом.

* 1. ShowAllStudents – друкує список усіх учнів.
  2. ShowAllTeachers – друкує список усіх викладачів.
  3. ShowStudentsList – допоміжний метод для виводу списку учнів.
  4. ShowTeachersList – допоміжний метод для виводу списку

викладачів.

* 1. IsValidIndex – перевіряє правильність індексу при виборі.
  2. Pause – викликає паузу для читання повідомлень на екрані.

**3 ІНТЕРФЕЙС**

Під час запуску програми користувач бачить наступне (Рис. 3.1)

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Рис 3.1 Вибір дії

Під час введення номеру вибраного користувачем перевіряється правильність цього номеру та можливість виконання дії. Перед кожною дією перевіряється наявність необхідних об’єктів (наприклад, не можна додати оцінку, якщо немає вчителів або учнів). Всі введення перевіряються на коректність (try-catch). У разі помилки виводиться повідомлення та повернення до вибору дії.

1. Додати учня (Рис 3.2):

Користувач вводить:

* повне ім’я;
* вік (ціле число);
* зріст (в см);
* клас (рядок).

Після успішного введення створюється новий об’єкт типу Student, який додається до списку учнів. У випадку помилки введення виводиться повідомлення про помилку.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рис 3.2. Додавання нового учня

2. Додати вчителя(Рис 3.3):

Користувач вводить:

* повне ім’я;
* вік (ціле число);
* зріст (в см);
* клас (рядок).

Изображение выглядит как текст, Шрифт, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рис 3.3. Додавання нового вчителя

На основі введених даних створюється об’єкт Teacher, що додається до списку вчителів. При некоректному введенні виводиться повідомлення про помилку.

3. Додати оцінку учневі (Рис 3.4)

Перед додаванням перевіряється, чи є у системі хоча б один учень і хоча б один вчитель.

* Виводиться список учнів — користувач обирає учня за номером.
* Формується список унікальних предметів, які викладаються вчителями — користувач обирає предмет.

Пропонується вибір типу оцінки:

1. Домашня
2. Контрольна
3. Самостійна
4. Іспит
5. Підсумкова

Користувач вводить значення оцінки (від 0 до 100).

На основі вибору створюється об'єкт SchoolMark, який додається до учня методом AddMark.

У разі будь-яких помилок введення (невірний індекс, неправильне значення оцінки) виводиться повідомлення про помилку.

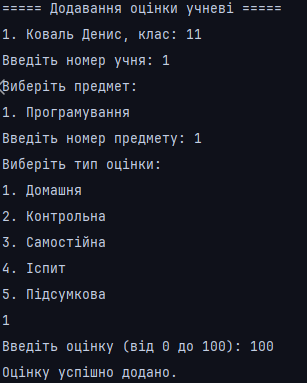


Рис 3.4 Додавання оцінки учневі

4. Інформація про учня(Рис 3.5)

Виводиться список усіх учнів із нумерацією. Користувач вводить номер учня, після чого виводиться повна інформація:

* ім’я, вік, зріст;
* клас;
* список усіх оцінок з предметами, типами та значенням.

Якщо індекс учня невірний, виводиться повідомлення про помилку.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рис 3.5 Інформація про учня

5. Інформація про вчителя (Рис 3.6)

Користувачу виводиться список учителів для вибору. Після вибору відображається:

* ім’я, вік, зріст;
* предмет;
* кількість учнів, прикріплених до цього вчителя;
* список учнів, що навчаються в нього.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рис 3.6 Інформація про вчителя

6. Додати учня вчителю(Рис 3.7)

Перед початком перевіряється наявність хоча б одного учня та вчителя.

1. Виводиться список учнів — користувач обирає учня.
2. Виводиться список учителів — користувач обирає вчителя.
3. Учень додається до списку учнів вчителя методом AddStudent.

Якщо введено неправильний індекс, виводиться повідомлення про помилку.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рис 3.7 Додати учня вчителю

7. Розрахувати підсумкову оцінку учня (Рис 3.8)

Перед початком перевіряється наявність учнів і вчителів.

1. Користувач обирає учня зі списку.
2. Формується список предметів, які викладають учителі.
3. Користувач обирає предмет.
4. Викликається метод GetFinalMark(subject):
   1. Обчислюється середнє значення поточних оцінок (домашні, самостійні, контрольні).
   2. Якщо є іспит — додається з вагою 25%.
   3. Результат округлюється до 2 знаків після коми.

Після обчислення виводиться загальна оцінка. У випадку помилок виводиться повідомлення.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рис 3.8 Розрахування підсумкової оцінки учня

8. Список усіх учнів (3.9)

Виводиться нумерований список усіх учнів у системі.

Якщо учнів немає — виводиться відповідне повідомлення.

Изображение выглядит как текст, Шрифт, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рис 3.9 Список усіх учнів

9. Список усіх учителів (3.10)

Виводиться нумерований список усіх вчителів.

Якщо вчителів немає — виводиться повідомлення.

Изображение выглядит как снимок экрана, текст, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рис 3.10 Список усіх учителів

0. Вихід

Завершує роботу програми. Підтвердження виходу не потрібне.

# **ВИСНОВКИ**

У ході виконання курсової роботи було реалізовано консольний застосунок для моделювання процесу управління шкільною системою із використанням об’єктно-орієнтованого програмування на мові C#.

У програмі реалізовано ієрархію класів: Person (базовий абстрактний клас), Student, Teacher, а також допоміжні класи SchoolMark і MenuController, які забезпечують повний функціонал керування даними.

Під час виконання роботи було реалізовано такі можливості:

* + додавання та перегляд інформації про учнів і вчителів;
  + призначення учнів до вчителів;
  + додавання оцінок різного типу;
  + розрахунок підсумкової оцінки на основі контрольних, домашніх робіт і іспиту;
  + перегляд повного списку учнів та вчителів;
  + обробка помилкових дій та введень з допомогою блоків try-catch.

Програма має зручний консольний інтерфейс, у якому користувач обирає дію через меню. Також забезпечено перевірку коректності вибору, захист від виняткових ситуацій (відсутність даних, некоректні оцінки, помилки введення).

Під час виконання роботи були набуті такі навички:

* застосування принципів об’єктно-орієнтованого програмування: успадкування, інкапсуляція, поліморфізм;
* створення ієрархічної структури класів із використанням абстрактних методів і віртуальних перевизначень;
* реалізація логіки взаємодії між об’єктами (учень—вчитель—оцінка);
* написання структурованого консольного застосунку з обробкою виключень (try-catch).

# **3 ПОСИЛАННЯ НА РЕПОЗИТОРІЙ**

https://github.com/Ensvalio/indz