ООП. Инкапсуляция

Реализовать предложенные классы с полями и методами, используя инкапсуляцию.

Вариант 1: Банковский счет

Реализовать систему банковского счета с полной защитой финансовых данных и операций.

Бизнес-логика для реализации:

- Баланс не может быть отрицательным
- Номер счета и владелец не могут изменяться после создания
- Все операции записываются в историю с timestamp
- При переводе проверять, что счет назначения существует
- Сумма операций должна быть положительной

ПОЛЯ у класса BankAccount:

- string accountNumber
- string ownerName
- decimal balance
- string[] transactionHistory
- DateTime createdDate

ОСНОВНЫЕ МЕТОДЫ:

void Deposit(decimal amount) -> Внесение средств
bool Withdraw(decimal amount) -> Снятие средств (возвращает успех)
bool Transfer(BankAccount targetAccount, decimal amount) -> Перевод

ВСПОМОГАТЕЛЬНЫЕ МЕТОДЫ:

private bool ValidateAmount(decimal amount) -> Проверка суммы (> 0)
private void LogTransaction(string type, decimal amount) -> Логирование операции

Вариант 2: Инвентарь магазина

Реализовать систему управления складом товаров с защитой данных о ценах и количестве.

Бизнес-логика для реализации:

- Цена товара не может быть отрицательной
- Количество товара не может быть отрицательным
- Артикул генерируется автоматически при создании товара
- Нельзя продать больше товара, чем есть на складе
- Скидка не может делать цену отрицательной

Требуемые классы и методы
Класс Product - ПОЛЯ:

- string name
- decimal price
- int quantity
- string sku (артикул)
- string category

ОСНОВНЫЕ МЕТОДЫ:

void UpdatePrice(decimal newPrice) -> Изменить цену
bool IncreaseQuantity(int amount) -> Увеличить количество
bool DecreaseQuantity(int amount) -> Уменьшить количество
void ApplyDiscount(decimal percent) -> Применить скидку

Вариант 3: Система бронирования

Реализовать упрощенную систему бронирования номеров в отеле.

Бизнес-логика:

- Дата выезда должна быть после даты заезда
- Нельзя отменить завершенное бронирование
- Цена за ночь не может быть отрицательной
- ID бронирования генерируется автоматически
- Статусы меняются только в правильной последовательности

Класс HotelBooking - ПОЛЯ:

- string bookingId
- string guestName
- int roomNumber
- DateTime checkInDate (Дата заезда)
- DateTime checkOutDate (Дата выезда)
- decimal pricePerNight
- string status

ОСНОВНЫЕ МЕТОДЫ:

public void ConfirmBooking() -> Подтвердить бронирование
public void CancelBooking() -> Отменить бронирование
public void MarkAsCompleted() -> Отметить как завершенное
public bool IsAvailableForDates(DateTime checkIn, DateTime checkOut) -> Проверить доступность дат

РАСЧЕТЫ:

public decimal CalculateTotalPrice() -> Рассчитать общую стоимость
public int GetNightsCount() -> Получить количество ночей
public bool IsActiveNow() -> Активно ли бронирование сейчас

Вариант 4: Фитнес-трекер

Реализовать упрощенную систему отслеживания фитнес-активности

Бизнес-логика:

- Шаги не могут быть отрицательными
- Цель по шагам должна быть не менее 1000
- Вес должен быть в разумных пределах (30-200 кг)
- При переходе на новый день сохраняется история
- Серия дней сбрасывается при пропуске цели

Класс FitnessTracker - ПОЛЯ:

- string userName
- int dailyStepGoal (Дневная цель по шагам)
- int stepsToday
- double weight
- int[] dailySteps
- double[] weightHistory
- DateTime currentDate

ОСНОВНЫЕ МЕТОДЫ:

void AddSteps(int steps) -> Добавить шаги
void ResetDailySteps() -> Сбросить дневные шаги
void UpdateWeight(double newWeight) -> Обновить вес
void SetNewGoal(int newGoal) -> Установить новую цель
void NewDay() -> Переход на новый день

СТАТИСТИКА:

void PrintDailySummary() -> Вывести дневную статистику
void PrintWeeklySummary() -> Вывести недельную статистику
bool IsGoalAchieved() -> Достигнута ли цель за сегодня