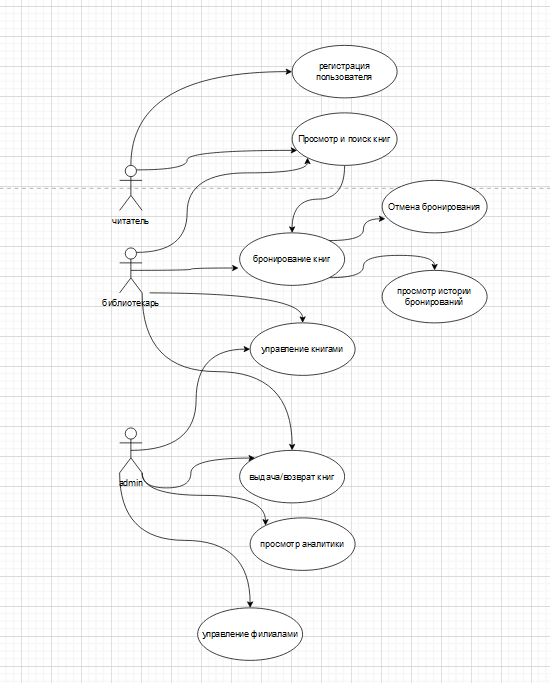
Задание №1. Диаграммы вариантов использования



Задание №2. Диаграмма состояния

Код:

using System;

// Интерфейс для состояний

public interface IState

{

    void SelectTicket(TicketMachine machine);

    void InsertMoney(TicketMachine machine, decimal amount);

    void DispenseTicket(TicketMachine machine);

    void CancelTransaction(TicketMachine machine);

}

// Состояние: Ожидание

public class IdleState : IState

{

    public void SelectTicket(TicketMachine machine)

    {

        Console.WriteLine("Билет выбран. Внесите деньги.");

        machine.SetState(new WaitingForMoneyState());

    }

    public void InsertMoney(TicketMachine machine, decimal amount)

    {

        Console.WriteLine("Сначала выберите билет.");

    }

    public void DispenseTicket(TicketMachine machine)

    {

        Console.WriteLine("Сначала выберите билет и оплатите.");

    }

    public void CancelTransaction(TicketMachine machine)

    {

        Console.WriteLine("Транзакция не может быть отменена, так как она еще не началась.");

    }

}

// Состояние: Ожидание внесения денег

public class WaitingForMoneyState : IState

{

    public void SelectTicket(TicketMachine machine)

    {

        Console.WriteLine("Билет уже выбран. Внесите деньги.");

    }

    public void InsertMoney(TicketMachine machine, decimal amount)

    {

        Console.WriteLine($"Деньги внесены: {amount} руб.");

        machine.SetState(new MoneyReceivedState());

    }

    public void DispenseTicket(TicketMachine machine)

    {

        Console.WriteLine("Сначала внесите деньги.");

    }

    public void CancelTransaction(TicketMachine machine)

    {

        Console.WriteLine("Транзакция отменена.");

        machine.SetState(new TransactionCanceledState());

    }

}

// Состояние: Деньги получены

public class MoneyReceivedState : IState

{

    public void SelectTicket(TicketMachine machine)

    {

        Console.WriteLine("Вы уже выбрали билет.");

    }

    public void InsertMoney(TicketMachine machine, decimal amount)

    {

        Console.WriteLine("Деньги уже внесены.");

    }

    public void DispenseTicket(TicketMachine machine)

    {

        Console.WriteLine("Билет выдан. Спасибо за покупку!");

        machine.SetState(new TicketDispensedState());

    }

    public void CancelTransaction(TicketMachine machine)

    {

        Console.WriteLine("Транзакция отменена. Возврат денег.");

        machine.SetState(new TransactionCanceledState());

    }

}

// Состояние: Билет выдан

public class TicketDispensedState : IState

{

    public void SelectTicket(TicketMachine machine)

    {

        Console.WriteLine("Транзакция завершена. Для нового билета начните сначала.");

    }

    public void InsertMoney(TicketMachine machine, decimal amount)

    {

        Console.WriteLine("Транзакция завершена. Для нового билета начните сначала.");

    }

    public void DispenseTicket(TicketMachine machine)

    {

        Console.WriteLine("Билет уже выдан.");

    }

    public void CancelTransaction(TicketMachine machine)

    {

        Console.WriteLine("Транзакция завершена. Отмена невозможна.");

    }

}

// Состояние: Транзакция отменена

public class TransactionCanceledState : IState

{

    public void SelectTicket(TicketMachine machine)

    {

        Console.WriteLine("Транзакция отменена. Для нового билета начните сначала.");

        machine.SetState(new IdleState());

    }

    public void InsertMoney(TicketMachine machine, decimal amount)

    {

        Console.WriteLine("Транзакция отменена. Для нового билета начните сначала.");

    }

    public void DispenseTicket(TicketMachine machine)

    {

        Console.WriteLine("Транзакция отменена. Билет не выдан.");

    }

    public void CancelTransaction(TicketMachine machine)

    {

        Console.WriteLine("Транзакция уже отменена.");

    }

}

// Класс автомата по продаже билетов

public class TicketMachine

{

    private IState \_currentState;

    public TicketMachine()

    {

        \_currentState = new IdleState();

    }

    public void SetState(IState state)

    {

        \_currentState = state;

    }

    public void SelectTicket()

    {

        \_currentState.SelectTicket(this);

    }

    public void InsertMoney(decimal amount)

    {

        \_currentState.InsertMoney(this, amount);

    }

    public void DispenseTicket()

    {

        \_currentState.DispenseTicket(this);

    }

    public void CancelTransaction()

    {

        \_currentState.CancelTransaction(this);

    }

}

// Тестирование работы автомата

public class Program

{

    public static void Main(string[] args)

    {

        var ticketMachine = new TicketMachine();

        ticketMachine.SelectTicket();

        ticketMachine.InsertMoney(100);

        ticketMachine.DispenseTicket();

        ticketMachine.CancelTransaction();

    }

}

Диаграмма:

