МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
КАФЕДРА ИИТ

Лабораторная работа №3

По дисциплине: «Современные платформы программирования»

Выполнил:
Студент 3 курса
группы ПО-8:
Печко В.И.
Проверил:
Крощенко А.А.

Брест 2024

Цель работы: научиться создавать и использовать классы в программах на языке программирования C#.

Вариант 18

Задание 1.

8) Множество целых чисел переменной мощности – Предусмотреть возможность пересечения двух множеств, вывода на печать элементов множества, а также метод, определяющий, принадлежит ли указанное значение множеству. Класс должен содержать методы, позволяющие добавлять и удалять элемент в/из множества. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализацию множества осуществить на базе структуры ArrayList. Реализовать метод equals, выполняющий сравнение объектов данного типа.

Код программы:

```
using System;
using System.Collections.Generic;

namespace Task_1
{
   internal class MySet
   {
      private List<int> _list;
      public string name;

      public List<int> List
      {
         set { _list = value; }
         get { return _list; }
      }

      public MySet(string name, bool initialize = false)
      {
         this.name = name;
         _list = new List<int>();
         if (initialize)
         {
            _list.Add(1);
            _list.Add(2);
            _list.Add(3);
         }
      }

      public void AddToList(int value)
      {
         _list.Add(value);
      }
      public void PrintValues()
      {
```

```
    Console.WriteLine($"{this.name} elements:");

    foreach (var item in _list)
    {
        Console.WriteLine(item);
    }
}
public void Remove(int number)
{
    _list.Remove(number);
    Console.WriteLine($"{number} was removed from {this.name}");
}
public bool Contains(int number)
{
    return _list.Contains(number);
}
public void Intersection(MySet obj)
{
    Console.WriteLine($"What {this.name}'s elements intersect with {obj.name}?");
    List<int> intersection = new List<int>();
    foreach (var item in _list)
    {
        if (obj.Contains(item))
        {
            intersection.Add(item);
        }
    }
    if (intersection.Count == 0)
    {
        Console.WriteLine("No common items");
    }
    else
    {
        foreach (var item in intersection)
        {
            Console.WriteLine(item);
        }
    }
}
public override bool Equals(object obj)
{
    if (obj is MySet other)
    {
        if (_list.Count != other._list.Count) return false;
        for (int i = 0; i < _list.Count; i++)
        {
            if (_list[i] != other._list[i])
            {
                return false;
            }
        }
        return true;
```

```
            }
            return false;
        }
        public override string ToString()
        {
            return string.Join(", ", _list);
        }
    }
}
```

```
using Task_1;

MySet set1 = new MySet("Set1");
MySet set2 = new MySet("Set2", true);
set1.AddToList(3);
set1.AddToList(1);
set1.AddToList(4);
set1.AddToList(5);
set1.PrintValues();
set2.AddToList(4);
set2.AddToList(3);
set2.AddToList(6);
set2.PrintValues();
Console.WriteLine(set1.Equals(set1));
Console.WriteLine(set1.Equals(set2));
set1.Intersection(set2);
Console.WriteLine($"Does {set1.name} contain 1?");
Console.WriteLine(set1.Contains(1));
Console.WriteLine($"Does {set2.name} contain 1?");
Console.WriteLine(set2.Contains(1));
set2.Remove(1);
Console.WriteLine($"Does {set2.name} contain 1?");
Console.WriteLine(set2.Contains(1));
```

Результат программы:

```
Set1 elements:
3
1
4
5
Set2 elements:
1
2
3
4
3
6
True
False
What Set1's elements intersect with Set2?
3
1
4
Does Set1 contain 1?
True
Does Set2 contain 1?
True
1 was removed from Set2
Does Set2 contain 1?
False
```

Задание 2.

8) Автоматизированная система обработки информации об авиарейсах

Написать программу для обработки информации об авиарейсах (Airlines): Каждый рейс имеет следующие характеристики:

- Пункт назначения;
- Номер рейса;
- Тип самолета;
- Время вылета;
- Дни недели, по которым совершаются рейсы.

Программа должна обеспечить:

- Генерацию списка рейсов;
- Вывод списка рейсов для заданного пункта назначения;
- Вывод списка рейсов для заданного дня недели;

Вывод списка рейсов для заданного дня недели, время вылета для которых больше заданного;

- Все рейсы самолетов некоторого типа;

Группировка рейсов по числу пассажиров (маломестные - 1-100 чел, средместные (100- 200), крупные рейсы (200-350));

- Все рейсы самолетов туда-обратно.

Код программы:

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;

public class Flight
{
   public string Destination { get; set; }
   public string FlightNumber { get; set; }
   public string AircraftType { get; set; }
   public TimeSpan DepartureTime { get; set; }
   public List<string> DaysOfWeek { get; set; }
   public int PassengerCount { get; set; }
}

public class FlightSystem
{
   private List<Flight> flights = new List<Flight>();

   public void LoadFlightsFromFile(string filename)
   {
      var lines = File.ReadAllLines(filename);
      foreach (var line in lines)
      {
         var data = line.Split(';');
         var flight = new Flight
         {
```

```
            Destination = data[0],
            FlightNumber = data[1],
            AircraftType = data[2],
            DepartureTime = TimeSpan.Parse(data[3]),
            DaysOfWeek = data[4].Split(',').ToList(),
            PassengerCount = int.Parse(data[5])
        };
        flights.Add(flight);
    }
}

class Program
{
    static void Main(string[] args)
    {
        FlightSystem flightSystem = new FlightSystem();
        flightSystem.LoadFlightsFromFile("flights.txt");

        Console.WriteLine("Список всех рейсов:");
        foreach (var flight in flightSystem.flights)
        {
            Console.WriteLine($"Рейс {flight.FlightNumber} в {flight.Destination} вылетает в
{flight.DepartureTime}");
        }

        var flightsToParis = flightSystem.GetFlightsByDestination("Paris");
        Console.WriteLine("\nРейсы в Париж:");
        foreach (var flight in flightsToParis)
        {
            Console.WriteLine($"Рейс {flight.FlightNumber} вылетает в
{flight.DepartureTime}");
        }

        var flightsOnMonday = flightSystem.GetFlightsByDayOfWeek("Monday");
        Console.WriteLine("\nРейсы в понедельник:");
        foreach (var flight in flightsOnMonday)
        {
            Console.WriteLine($"Рейс {flight.FlightNumber} в {flight.Destination} вылетает в
{flight.DepartureTime}");
        }

        var flightsOnMondayAfter10 = flightSystem.GetFlightsByDayAndTime("Monday", new
TimeSpan(10, 0, 0));
        Console.WriteLine("\nРейсы в понедельник после 10 утра:");
        foreach (var flight in flightsOnMondayAfter10)
        {
            Console.WriteLine($"Рейс {flight.FlightNumber} в {flight.Destination} вылетает в
{flight.DepartureTime}");
        }

        var boeingFlights = flightSystem.GetFlightsByAircraftType("Boeing 737");
        Console.WriteLine("\nРейсы на Boeing 737:");
```

```
        foreach (var flight in boeingFlights)
        {
            Console.WriteLine($"Рейс {flight.FlightNumber} в {flight.Destination} вылетает в
{flight.DepartureTime}");
        }

        var groupedFlights = flightSystem.GroupFlightsByPassengerCount();
        Console.WriteLine("\nРейсы, сгруппированные по числу пассажиров:");
        foreach (var group in groupedFlights)
        {
            Console.WriteLine($"\n{group.Key}:");
            foreach (var flight in group.Value)
            {
                Console.WriteLine($"Рейс {flight.FlightNumber} в {flight.Destination} вылетает в
{flight.DepartureTime}");
            }
        }

        var roundTrips = flightSystem.GetRoundTrips();
        Console.WriteLine("\nРейсы туда-обратно:");
        foreach (var flight in roundTrips)
        {
            Console.WriteLine($"Рейс {flight.FlightNumber} в {flight.Destination} вылетает в
{flight.DepartureTime} и возвращается обратно");
        }
    }
}

public List<Flight> GetFlightsByDestination(string destination)
{
    return flights.Where(f => f.Destination == destination).ToList();
}

public List<Flight> GetFlightsByDayOfWeek(string dayOfWeek)
{
    return flights.Where(f => f.DaysOfWeek.Contains(dayOfWeek)).ToList();
}

public List<Flight> GetFlightsByDayAndTime(string dayOfWeek, TimeSpan time)
{
    return flights.Where(f => f.DaysOfWeek.Contains(dayOfWeek) && f.DepartureTime >
time).ToList();
}

public List<Flight> GetFlightsByAircraftType(string aircraftType)
{
    return flights.Where(f => f.AircraftType == aircraftType).ToList();
}

public Dictionary<string, List<Flight>> GroupFlightsByPassengerCount()
{
    return new Dictionary<string, List<Flight>>
```

```
        {
            {"Маломестные", flights.Where(f => f.PassengerCount >= 1 && f.PassengerCount <=
100).ToList()},
            {"Средместные", flights.Where(f => f.PassengerCount > 100 && f.PassengerCount <=
200).ToList()},
            {"Крупные", flights.Where(f => f.PassengerCount > 200 && f.PassengerCount <=
350).ToList()}
        };
    }

    public List<Flight> GetRoundTrips()
    {
        return flights.Where(f1 => flights.Any(f2 =>
f2.Destination.Trim().Equals(f1.FlightNumber.Trim(), StringComparison.OrdinalIgnoreCase)
&& f2.FlightNumber.Trim().Equals(f1.Destination.Trim(),
StringComparison.OrdinalIgnoreCase))).ToList();
    }
}
```

Спецификация ввода

flights.txt

ПунктНазначения;НомерРейса;ТипСамолета;ВремяВылета;ДниНедели;КоличествоПассажиров

Пример

Paris;AF123;Boeing 737;10:00;Monday,Tuesday,Wednesday,Thursday,Friday;150
London;BA456;Airbus A320;15:30;Monday,Wednesday,Friday;180
New York;DL789;Boeing 777;20:00;Tuesday,Thursday,Saturday;300
Tokyo;JL012;Boeing 787;23:30;Monday,Wednesday,Friday,Sunday;250
Paris;AF124;Boeing 737;12:00;Tuesday,Thursday,Saturday;90
London;BA457;Airbus A320;16:30;Tuesday,Thursday,Sunday;210
New York;DL790;Boeing 777;21:00;Monday,Wednesday,Friday;350
Tokyo;JL013;Boeing 787;00:30;Tuesday,Thursday,Saturday,Sunday;100
AF123;Paris;Boeing 737;15:00;Monday,Tuesday,Wednesday,Thursday,Friday;150
BA456;London;Airbus A320;20:30;Monday,Wednesday,Friday;180
DL789;New York;Boeing 777;01:00;Tuesday,Thursday,Saturday;300
JL012;Tokyo;Boeing 787;04:30;Monday,Wednesday,Friday,Sunday;250
AF124;Paris;Boeing 737;17:00;Tuesday,Thursday,Saturday;90
BA457;London;Airbus A320;21:30;Tuesday,Thursday,Sunday;210
DL790;New York;Boeing 777;02:00;Monday,Wednesday,Friday;350
JL013;Tokyo;Boeing 787;05:30;Tuesday,Thursday,Saturday,Sunday;100

Результат программы:

```
Список всех рейсов:
Рейс AF123 в Paris вылетает в 10:00:00
Рейс BA456 в London вылетает в 15:30:00
Рейс DL789 в New York вылетает в 20:00:00
Рейс JL012 в Tokyo вылетает в 23:30:00
Рейс AF124 в Paris вылетает в 12:00:00
Рейс BA457 в London вылетает в 16:30:00
Рейс DL790 в New York вылетает в 21:00:00
Рейс JL013 в Tokyo вылетает в 00:30:00
Рейс Paris в AF123 вылетает в 15:00:00
Рейс London в BA456 вылетает в 20:30:00
Рейс New York в DL789 вылетает в 01:00:00
Рейс Tokyo в JL012 вылетает в 04:30:00
Рейс Paris в AF124 вылетает в 17:00:00
Рейс London в BA457 вылетает в 21:30:00
Рейс New York в DL790 вылетает в 02:00:00
Рейс Tokyo в JL013 вылетает в 05:30:00

Рейсы в Париж:
Рейс AF123 вылетает в 10:00:00
Рейс AF124 вылетает в 12:00:00

Рейсы в понедельник:
Рейс AF123 в Paris вылетает в 10:00:00
Рейс BA456 в London вылетает в 15:30:00
Рейс JL012 в Tokyo вылетает в 23:30:00
Рейс DL790 в New York вылетает в 21:00:00
Рейс Paris в AF123 вылетает в 15:00:00
Рейс London в BA456 вылетает в 20:30:00
Рейс Tokyo в JL012 вылетает в 04:30:00
Рейс New York в DL790 вылетает в 02:00:00

Рейсы в понедельник после 10 утра:
Рейс BA456 в London вылетает в 15:30:00
Рейс JL012 в Tokyo вылетает в 23:30:00
Рейс DL790 в New York вылетает в 21:00:00
Рейс Paris в AF123 вылетает в 15:00:00
Рейс London в BA456 вылетает в 20:30:00
```

```
Рейсы на Boeing 737:
Рейс AF123 в Paris вылетает в 10:00:00
Рейс AF124 в Paris вылетает в 12:00:00
Рейс Paris в AF123 вылетает в 15:00:00
Рейс Paris в AF124 вылетает в 17:00:00

Рейсы, сгруппированные по числу пассажиров:

Маломестные:
Рейс AF124 в Paris вылетает в 12:00:00
Рейс JL013 в Tokyo вылетает в 00:30:00
Рейс Paris в AF124 вылетает в 17:00:00
Рейс Tokyo в JL013 вылетает в 05:30:00

Средместные:
Рейс AF123 в Paris вылетает в 10:00:00
Рейс BA456 в London вылетает в 15:30:00
Рейс Paris в AF123 вылетает в 15:00:00
Рейс London в BA456 вылетает в 20:30:00

Крупные:
Рейс DL789 в New York вылетает в 20:00:00
Рейс JL012 в Tokyo вылетает в 23:30:00
Рейс BA457 в London вылетает в 16:30:00
Рейс DL790 в New York вылетает в 21:00:00
Рейс New York в DL789 вылетает в 01:00:00
Рейс Tokyo в JL012 вылетает в 04:30:00
Рейс London в BA457 вылетает в 21:30:00
Рейс New York в DL790 вылетает в 02:00:00

Рейсы туда-обратно:
Рейс AF123 в Paris вылетает в 10:00:00 и возвращается обратно
Рейс BA456 в London вылетает в 15:30:00 и возвращается обратно
Рейс DL789 в New York вылетает в 20:00:00 и возвращается обратно
Рейс JL012 в Tokyo вылетает в 23:30:00 и возвращается обратно
Рейс AF124 в Paris вылетает в 12:00:00 и возвращается обратно
Рейс BA457 в London вылетает в 16:30:00 и возвращается обратно
Рейс DL790 в New York вылетает в 21:00:00 и возвращается обратно
Рейс JL013 в Tokyo вылетает в 00:30:00 и возвращается обратно
```

Вывод: научились создавать и использовать классы в программах на языке программирования C#.