Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Новосибирский государственный технический университет»

Кафедра Автоматизированных систем управления

**Практическая РАБОТА № 4**

по дисциплине «Программирование»

|  |  |
| --- | --- |
| Выполнил:  Попова Дарья Алексеевна  Ткачук Фёдор Андреевич  Группа АВТ-312  Факультет АВТФ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г. | Проверил:  Достовалов Дмитрий Николаевич  Балл: \_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г. |

Новосибирск 2025

Вариант 6

|  |  |
| --- | --- |
| **№ вар.** | **Класс** |
| 1 | Создать класс Дата с полями: день (1-31), месяц (1-12), год (целое число). |

|  |  |
| --- | --- |
| **№ вар.** | **Запросы** |
| 1 | список дат для заданного года  список дат в заданном месяце  количество дат в определённом диапазоне  максимальную дату  первую дату для заданного дня  упорядоченный список дат (хронологически) |

**Листинг программы**

Program.cs

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  public class Date  {  public int Day { get; set; }  public int Month { get; set; }  public int Year { get; set; }  public Date(int day, int month, int year)  {  if (day < 1 || day > 31) throw new ArgumentOutOfRangeException(nameof(day), "Day must be between 1 and 31.");  if (month < 1 || month > 12) throw new ArgumentOutOfRangeException(nameof(month), "Month must be between 1 and 12.");  Year = year;  Day = day;  Month = month;  }  public override string ToString()  {  return $"{Day:D2}.{Month:D2}.{Year}"; // Форматирование для удобного вывода  }  }  public class DateQueries  {  public static void Main(string[] args)  {  // Пример данных  List<Date> dates = new List<Date>()  {  new Date(10, 10, 2023),  new Date(15, 05, 2024),  new Date(20, 10, 2023),  new Date(01, 01, 2024),  new Date(28, 02, 2023),  new Date(12, 05, 2024),  new Date(31, 12, 2023)  };  // 1. Список дат для заданного года  int targetYear = 2023;  var datesForYear = GetDatesForYear(dates, targetYear);  Console.WriteLine($"Даты для года {targetYear}:");  foreach (var date in datesForYear)  {  Console.WriteLine(date);  }  Console.WriteLine();  // 2. Список дат в заданном месяце  int targetMonth = 05;  var datesForMonth = GetDatesForMonth(dates, targetMonth);  Console.WriteLine($"Даты в месяце {targetMonth}:");  foreach (var date in datesForMonth)  {  Console.WriteLine(date);  }  Console.WriteLine();  // 3. Количество дат в определённом диапазоне  Date startDate = new Date(01, 01, 2023);  Date endDate = new Date(31, 12, 2023);  int dateCountInRange = GetDateCountInRange(dates, startDate, endDate);  Console.WriteLine($"Количество дат в диапазоне {startDate} - {endDate}: {dateCountInRange}");  Console.WriteLine();  // 4. Максимальную дату  Date maxDate = GetMaxDate(dates);  Console.WriteLine($"Максимальная дата: {maxDate}");  Console.WriteLine();  // 5. Первую дату для заданного дня  int targetDay = 10;  Date firstDateForDay = GetFirstDateForDay(dates, targetDay);  Console.WriteLine($"Первая дата для дня {targetDay}: {firstDateForDay}");  Console.WriteLine();  // 6. Упорядоченный список дат (хронологически)  var sortedDates = GetSortedDates(dates);  Console.WriteLine("Упорядоченный список дат:");  foreach (var date in sortedDates)  {  Console.WriteLine(date);  }  Console.WriteLine();  }  // 1. Список дат для заданного года  public static IEnumerable<Date> GetDatesForYear(List<Date> dates, int year)  {  return dates.Where(date => date.Year == year);  }  // 2. Список дат в заданном месяце  public static IEnumerable<Date> GetDatesForMonth(List<Date> dates, int month)  {  return dates.Where(date => date.Month == month);  }  // 3. Количество дат в определённом диапазоне  public static int GetDateCountInRange(List<Date> dates, Date startDate, Date endDate)  {  return dates.Count(date =>  date.Year >= startDate.Year && date.Year <= endDate.Year &&  date.Month >= (date.Year == startDate.Year ? startDate.Month : 1) &&  date.Month <= (date.Year == endDate.Year ? endDate.Month : 12) &&  date.Day >= (date.Year == startDate.Year && date.Month == startDate.Month ? startDate.Day : 1) &&  date.Day <= (date.Year == endDate.Year && date.Month == endDate.Month ? endDate.Day : 31)  );  }  // 4. Максимальную дату  public static Date GetMaxDate(List<Date> dates)  {  return dates.OrderByDescending(date => date.Year)  .ThenByDescending(date => date.Month)  .ThenByDescending(date => date.Day)  .FirstOrDefault();  }  // 5. Первую дату для заданного дня  public static Date GetFirstDateForDay(List<Date> dates, int day)  {  return dates.Where(date => date.Day == day).OrderBy(date => date.Year).ThenBy(date => date.Month).ThenBy(date => date.Day).FirstOrDefault();  }  // 6. Упорядоченный список дат (хронологически)  public static IEnumerable<Date> GetSortedDates(List<Date> dates)  {  return dates.OrderBy(date => date.Year)  .ThenBy(date => date.Month)  .ThenBy(date => date.Day);  }  } |

**Скриншоты работы программы**

