Дисциплина «Программирование корпоративных систем» Рабочая тетрадь 6

Делегаты, лямбда-выражения, события

|  |  |
| --- | --- |
| Ознакомьтесь с теоретическим материалом лекции №6 | |
| **Задание 1** | |
| ***Задача:*** | |
|  | **Использование делегатов для сортировки массива**  Напишите программу, которая сортирует массив строк по разным критериям с использованием делегатов. Программа должна предложить пользователю выбрать один из следующих способов сортировки:   * По длине строки (от самой короткой до самой длинной). * По алфавиту. * По количеству гласных букв в строке.   Создайте соответствующий делегат и передайте его в метод сортировки.  Сопроводите решение текстовым объяснением использования вами механизма делегатов. |
| ***Решение:*** | |
|  | using System;  using System.Linq;  class Program  {  delegate string[] SortDelegate(string[] arr);  static void Main()  {  string[] strings = { "appleee", "banana", "kiw", "cherry", "orangeee" };  while (true)  {  Console.WriteLine("Выберите способ сортировки:\n1. По длине строки\n2. По алфавиту\n3. По количеству гласных букв");  int choice = int.Parse(Console.ReadLine());  SortDelegate sorter = choice switch  {  1 => arr => arr.OrderBy(s => s.Length).ToArray(),  2 => arr => arr.OrderBy(s => s).ToArray(),  3 => arr => arr.OrderBy(s => s.Count(c => "aeiou".Contains(char.ToLower(c)))).ToArray(),  \_ => throw new ArgumentException("Неверный выбор")  };  strings = sorter(strings);  Console.WriteLine("Отсортированный массив: " + string.Join(", ", strings));  }  }  }  Данная программа на языке C# демонстрирует использование делегатов для сортировки массива строк различными способами. Программа предлагает пользователю выбрать метод сортировки из трех предложенных вариантов: по длине строки, по алфавиту или по количеству гласных букв в строке. Делегат SortDelegate объявлен для представления метода, который принимает массив строк и возвращает отсортированный массив строк. Отображается меню с вариантами сортировки. В зависимости от выбора пользователя, устанавливается соответствующий метод сортировки и выводится результат на экран. |
| ***Ответ:*** | |
|  |  |
| **Задание 2** | |
| ***Задача:*** | |
|  | **Фильтрация строк с использованием лямбда-выражений**  Напишите программу, которая фильтрует строки в массиве по указанным пользователем критериям с использованием лямбда-выражений. Программа должна предложить пользователю выбрать один из следующих вариантов фильтрации:   * Строки, длина которых больше указанного числа. * Строки, начинающиеся с определенной буквы. * Строки, содержащие заданное количество гласных букв или больше.   Реализуйте фильтрацию с помощью лямбда-выражений.  Сопроводите решение текстовым объяснением использования вами механизма лямбда-выражений. |
| ***Решение:*** | |
|  | using System;  using System.Linq;  class FilterProgram  {  static void Main()  {  string[] strings = { "appleee", "banana", "kiw", "cherry", "orangeee" };  while (true)  {  Console.WriteLine("Выберите способ фильтрации:\n1. Длина больше заданного числа\n2. Начинается с буквы\n3. Содержит заданное количество гласных или больше");  int choice = int.Parse(Console.ReadLine());  int minLength = 0;  string startLetter = "";  int minVowels = 0;  switch (choice)  {  case 1:  minLength = ReadInt("Введите минимальную длину: ");  break;  case 2:  startLetter = ReadString("Введите начальную букву: ");  break;  case 3:  minVowels = ReadInt("Введите минимальное количество гласных: ");  break;  default:  throw new ArgumentException("Неверный выбор");  }  var filtered = choice switch  {  1 => strings.Where(s => s.Length > minLength).ToArray(),  2 => strings.Where(s => s.StartsWith(startLetter)).ToArray(),  3 => strings.Where(s => s.Count(c => "aeiou".Contains(char.ToLower(c))) >= minVowels).ToArray(),  \_ => Array.Empty<string>()  };  Console.WriteLine("Отфильтрованные строки: " + string.Join(", ", filtered));  }  }  static int ReadInt(string message) => int.Parse(ReadString(message));  static string ReadString(string message)  {  Console.Write(message);  return Console.ReadLine();  }  }   Программа использует лямбда-выражения для фильтрации строк по заданным критериям. Пользователь может выбирать критерии фильтрации несколько раз, пока не закроет программу. Лямбда-выражение s => s.Length > minLength определяет условие фильтрации, где s представляет каждую строку в массиве, и фильтрация выполняется для строк, длина которых больше заданного minLength.  Лямбда-выражение s => s.StartsWith(startLetter) определяет условие фильтрации, где s представляет каждую строку в массиве, и фильтрация выполняется для строк, начинающихся с заданного символа startLetter.  Лямбда-выражение s => s.Count(c => "aeiou".Contains(char.ToLower(c))) >= minVowels определяет условие фильтрации, где s представляет каждую строку в массиве, и фильтрация выполняется для строк, содержащих заданное количество гласных букв или больше. |
| ***Ответ:*** | |
|  |  |

|  |  |
| --- | --- |
| **Задание 3** | |
| ***Задача:*** | |
|  | **Напишите программу, которая моделирует работу термостата.**  Устройство отслеживает изменение температуры и генерирует события, если:   * Температура превышает заданное максимальное значение. * Температура падает ниже заданного минимального значения.   Программа должна:   * Позволить пользователю задать минимальную и максимальную температуры. * Реализовать два обработчика событий: один для предупреждения о перегреве, другой — о переохлаждении. * Симулировать изменение температуры в случайных пределах.   Сопроводите решение текстовым объяснением использования вами механизма событий. |
| ***Решение:*** | |
|  | using System;  using System.Threading;  class Thermostat  {  public event Action<string> TemperatureAlert;  private int minTemp;  private int maxTemp;  private Random rand = new Random();  private bool isRunning;  public Thermostat(int min, int max)  {  minTemp = min;  maxTemp = max;  }  public void SimulateTemperature()  {  isRunning = true;  int currentTemp = rand.Next(minTemp, maxTemp);  Thread simulationThread = new Thread(() =>  {  while (isRunning)  {  CheckTemperature(currentTemp);  Thread.Sleep(2000);  }  });  simulationThread.Start();  Console.WriteLine("Нажмите любую клавишу для остановки...");  Console.ReadKey();  isRunning = false;  simulationThread.Join();  }  private void CheckTemperature(int currentTemp)  {  currentTemp += rand.Next(-50, 50);  Console.WriteLine($"Текущая температура: {currentTemp}°C");  if (currentTemp < minTemp) TemperatureAlert?.Invoke("Предупреждение: переохлаждение!");  if (currentTemp > maxTemp) TemperatureAlert?.Invoke("Предупреждение: перегрев!");  }  }  class ThermostatProgram  {  static void Main()  {  Console.Write("Введите минимальную температуру: ");  int min = int.Parse(Console.ReadLine());  Console.Write("Введите максимальную температуру: ");  int max = int.Parse(Console.ReadLine());  Thermostat thermostat = new Thermostat(min, max);  thermostat.TemperatureAlert += msg => Console.WriteLine(msg);  thermostat.SimulateTemperature();  }  }  Программа моделирует работу термостата, который отслеживает температуру в заданных пользователем пределах. Пользователь вводит минимальную и максимальную допустимые температуры, после чего начинается симуляция.  Текущая температура обновляется случайным образом в пределах от -50 до +50 градусов. Если температура выходит за заданные границы, термостат генерирует события:  Перегрев: если температура превышает максимальное значение, выводится предупреждение о перегреве.  Переохлаждение: если температура опускается ниже минимального значения, отображается предупреждение о переохлаждении.  Механизм событий реализован с использованием делегата Action<string>, который вызывает событие TemperatureAlert при обнаружении критической температуры. Симуляция выполняется в отдельном потоке с использованием класса Thread, что позволяет периодически обновлять температуру и проверять её состояние, пока пользователь не остановит программу. |
| ***Ответ:*** | |

