Ильяхова Алиса, Б9122-02.03.01сцт

Лабораторная работа №6. Передача данных потокам

Вариант 9

1. Цель и задачи работы

Цель:

Научиться создавать многопоточные приложения с возможностью передачи параметров в параллельно выполняемые потоки.

Задачи:

- Изучить механизм передачи параметров в поток;
- Научиться использовать пул потоков (ThreadPool);
- Реализовать обработку коллекции элементов с использованием пула потоков.

2. Реализация индивидуального задания

2.1. Условие варианта 9

Согласно таблице на стр. 57:

- **Задача**: Метод находит логическое значение, указывающее, существует ли заданное число в массиве целых случайных чисел.
- **Тип элемента коллекции**: Класс, описывающий массив случайных чисел (размер выбирается случайно, искомое число задаётся в конструкторе).

2.2. Класс ArrayItem — элемент коллекции

Реализован класс ArrayItem, содержащий:

- int[] Array — массив случайных чисел;
- int Target — искомое число.

Конструктор генерирует массив заданного размера со случайными значениями от 1 до 99.

2.3. Класс ArrayProcessor — обработчик

Метод ProcessItem принимает параметр типа object (требование ParameterizedThreadStart) и:

- Приводит его к типу ArrayItem;
- Выполняет поиск числа в массиве;
- Имитирует долгую операцию через Thread.Sleep(1500);
- Выводит результат с указанием ID потока.

2.4. Использование пула потоков

В методе Main:

- Создаётся 5 объектов ArrayItem с разными размерами массивов и искомыми числами.
- Каждый объект передаётся в пул потоков через:ThreadPool.QueueUserWorkItem(new ArrayProcessor().ProcessItem, item);

- Основной поток ожидает завершения через Thread.Sleep(3000).

3. Ответы на контрольные вопросы

1. **Какие существуют способы передачи параметров в поток?**
    a. Через делегат ParameterizedThreadStart (параметр типа object);
    b. Через замыкание (лямбда-выражение или анонимный метод);
    c. Через члены класса (если поток запускает метод экземпляра).

2. **Можно ли передать в несколько потоков один и тот же параметр (ссылку на объект)?**

Да, можно. Все потоки получат ссылку на один и тот же объект. Однако это может привести к **гонке данных**, если объект изменяемый. В нашем случае объекты неизменяемы после создания, поэтому безопасны.

3. **Для чего используется пул потоков?**

Пул потоков позволяет повторно использовать уже созданные потоки, что снижает накладные расходы на создание/уничтожение потоков и повышает производительность.

4. **Какой метод используется для отправки задачи в пул потоков?**

ThreadPool.QueueUserWorkItem(WaitCallback callBack, object state).

5. **Как обеспечить синхронизацию завершения всех задач пула потоков?**

В учебных целях используется Thread.Sleep. В реальных приложениях применяются CountdownEvent, Task.WhenAll или ManualResetEvent.

4. Экранные формы и листинг программы

4.1. Консольный вывод программы

```
=== Лабораторная работа №6. Вариант 9 ===
Передача данных потокам с использованием пула потоков

Все задачи отправлены в пул потоков.
Ожидание завершения (3 секунды)...
[Поток 5]
Искомое число: 11
[Поток 9]
Найдено: False
Искомое число: 26
Найдено: False
[Поток 10]
Искомое число: 6
Найдено: False
[Поток 8]
[Поток 7]
Искомое число: 6
Искомое число: 36
2 Найдено: False
Массив: [56, 53, 28, 33, 60]
Массив: [57, 54, 22, 63, 90, 41, 99, 55, 68, 49, 45]
Массив: [2, 69, 6, 23, 19, 50, 89, 98, 29, 37, 87, 40]
Массив: [61, 50, 8, 5, 89, 51, 33, 44, 53, 26, 45, 73]
Массив: [76, 50, 43, 88, 68, 43, 57, 22, 11]
5 ------------------------------------------------

 Готово.
```

4.2. Полный листинг программы с комментариями

```
using System;
using System.Threading;

// Лабораторная работа №6. Передача данных потокам
```

```
// Вариант 9

// Класс-элемент коллекции (содержит массив и искомое число)
class ArrayItem
{
    public int[] Array { get; private set; }
    public int Target { get; private set; }

    public ArrayItem(int size, int target)
    {
        var rand = new Random();
        Array = new int[size];
        for (int i = 0; i < size; i++)
        {
            Array[i] = rand.Next(1, 100);
        }
        Target = target;
    }
}

// Класс для обработки элемента с использованием пула потоков
class ArrayProcessor
{
    public void ProcessItem(object item)
    {
        var arrayItem = (ArrayItem)item;
        bool found = false;

        // Имитация долгой операции
        Thread.Sleep(1500);

        foreach (int value in arrayItem.Array)
        {
            if (value == arrayItem.Target)
            {
                found = true;
                break;
            }
        }

        // Вывод результата
        Console.WriteLine($"[Поток {Thread.CurrentThread.ManagedThreadId}]");
        Console.WriteLine($"Искомое число: {arrayItem.Target}");
        Console.WriteLine($"Найдено: {found}");
        Console.WriteLine($"Массив: [{string.Join(", ", arrayItem.Array)}]");
        Console.WriteLine(new string('-', 50));
    }
}

class Program
{
```

```
    static void Main()
    {
        Console.WriteLine("=== Лабораторная работа №6. Вариант 9 ===");
        Console.WriteLine("Передача данных потокам с использованием пула потоков\n");

        Random rand = new Random();

        // Создаём 5 элементов коллекции
        for (int i = 0; i < 5; i++)
        {
            int arraySize = rand.Next(5, 15); // размер от 5 до 14
            int targetNumber = rand.Next(1, 100);
            var item = new ArrayItem(arraySize, targetNumber);

            // Отправляем задачу в пул потоков
            ThreadPool.QueueUserWorkItem(new ArrayProcessor().ProcessItem, item);
        }

        Console.WriteLine("Все задачи отправлены в пул потоков.");
        Console.WriteLine("Ожидание завершения (3 секунды)...");

        // Даём время на выполнение всех задач
        Thread.Sleep(3000);

        Console.WriteLine("\n Готово.");
    }
}
```

5. Вывод

В ходе выполнения лабораторной работы №6 были:

- Реализован класс ArrayItem для хранения данных;
- Создан обработчик ArrayProcessor для поиска числа в массиве;
- Продемонстрирована передача параметров в поток через ThreadPool.QueueUserWorkItem;
- Показано использование пула потоков для параллельной обработки коллекции;
- Обеспечена визуализация выполнения с указанием ID потоков.