Лекция 7. Оператор цикла foreach

Цель лекции: Формирование у студентов знаний, связанных с использованием оператора цикла foreach с одномерными массивами.

В результате изучения данной лекции студенты будут способны:

- Продемонстрировать знание синтаксиса использования оператора foreach
- Продемонстрировать понимание особенности работы оператора foreach

Оператор foreach служит для циклического обращения к элементам *коллекции,* представляющей собой группу объектов. В C# определено несколько видов коллекций, каждая из которых является массивом. Ниже приведена общая форма оператора цикла foreach.

foreach (*тип имя_переменной_цикла* in *коллекция*) *оператор;*

Здесь *тип имя_переменной_цикла* обозначает тип и имя переменной управления циклом, которая получает значение следующего элемента коллекции на каждом шаге выполнения цикла foreach. А *коллекция* обозначает циклически опрашиваемую коллекцию, которая здесь и далее представляет собой массив. Следовательно, *тип* переменной цикла должен соответствовать типу элемента массива. Кроме того, *тип* может обозначаться ключевым словом var. В этом случае компилятор определяет тип переменной цикла, исходя из типа элемента массива. Но, как правило, тип указывается явным образом.

Оператор цикла foreach действует следующим образом. Когда цикл начинается, первый элемент массива выбирается и присваивается переменной цикла. На каждом последующем шаге итерации выбирается следующий элемент массива, который сохраняется в переменной цикла. Цикл завершается, когда все элементы массива окажутся выбранными. Следовательно, оператор foreach циклически опрашивает массив по отдельным его элементам от начала и до конца.

Следует, однако, иметь в виду, что переменная цикла в операторе foreach служит только для чтения. Это означает, что, присваивая этой переменной новое значение, нельзя изменить содержимое массива.

Ниже приведен простой пример применения оператора цикла foreach. В этом примере сначала создается целочисленный массив и задается ряд его первоначальных

значений, а затем эти значения выводятся, а по ходу дела вычисляется их сумма.

```
// Использовать оператор цикла foreach.
using System;
class ForeachDemo {
static void Main() {
int sum = 0;
int[] nums = new int[10];
```

```
// Задать первоначальные значения элементов массива nums.
for(int i = 0; i < 10; i++)
nums[i] = i;
// Использовать цикл foreach для вывода значений
// элементов массива и подсчета их суммы.
foreach(int x in nums) {
Console.WriteLine("Значение элемента равно: " + x);
sum += x;
}
Console.WriteLine("Сумма равна: " + sum);
}
}
```

Выполнение приведенного выше кода дает следующий результат.

Значение элемента равно: 0

Значение элемента равно: 1

Значение элемента равно: 2

Значение элемента равно: 3

Значение элемента равно: 4

Значение элемента равно: 5

Значение элемента равно: 6

Значение элемента равно: 7

Значение элемента равно: 8

Значение элемента равно: 9

Сумма равна: 45

Как видите, оператор foreach циклически опрашивает массив по порядку индексирования от самого первого до самого последнего его элемента.

Несмотря на то что цикл foreach повторяется до тех пор, пока не будут опрошены все элементы массива, его можно завершить преждевременно, воспользовавшись оператором break. Ниже приведен пример программы, в которой суммируются только пять первых элементов массива nums.

```
// Использовать оператор break для преждевременного завершения цикла
foreach.
using System;
class ForeachDemo {
static void Main() {
int sum = 0;
int[] nums = new int[10];
// Задать первоначальные значения элементов массива nums.
for(int i = 0; i < 10; i++)
nums[i] = i;
// Использовать цикл foreach для вывода значений
// элементов массива и подсчета их суммы.
foreach(int x in nums) {
Console.WriteLine("Значение элемента равно: " + x);
sum += x;
if(x == 4) break; // прервать цикл, как только индекс массива достигнет 4
}
Console.WriteLine("Сумма первых 5 элементов: " + sum);
}
}
```

Вот какой результат дает выполнение этой программы.

Значение элемента равно: 0

Значение элемента равно: 1

Значение элемента равно: 2

Значение элемента равно: 3

Значение элемента равно: 4

Сумма первых 5 элементов: 10

Совершенно очевидно, что цикл foreach завершается после выбора и вывода значения пятого элемента массива.

Заключение

В рамках данной лекции был изучен оператор цикла `foreach`, предназначенный для последовательного перебора элементов одномерных массивов и коллекций в языке C#. Были рассмотрены его синтаксис, механизм автоматического извлечения элементов от начала до конца массива, правила типизации переменной цикла (включая применение ключевого слова `var`), а также возможности досрочного завершения работы цикла с помощью оператора `break`.

Оператор `foreach` является удобным, наглядным и безопасным инструментом для чтения и обработки всех элементов массива (например, при выводе данных или подсчете суммы), так как избавляет программиста от необходимости вручную управлять счетчиком и следить за границами индексирования. Однако важно учитывать его ключевое ограничение: переменная цикла доступна только для чтения, поэтому для изменения элементов массива или обхода элементов в обратном порядке следует использовать классический цикл `for`.

Вопросы для самопроверки

1. **Назначение оператора:** Для решения каких задач предназначен оператор цикла `foreach` в языке C# и в чем его преимущество перед циклом `for` при простом просмотре элементов массива?

2. **Синтаксис и типизация:** Приведите общую форму записи оператора `foreach`. Каким требованиям должен соответствовать тип переменной цикла, и как в заголовке этого цикла можно использовать ключевое слово `var`?

3. **Порядок обхода:** В какой последовательности оператор `foreach` обращается к элементам одномерного массива? Можно ли с его помощью организовать обход массива от последнего элемента к первому?

4. **Ограничение на изменение данных:** Можно ли внутри тела цикла `foreach` изменить значение элемента массива путем присваивания нового значения переменной цикла? Обоснуйте свой ответ.

5. **Досрочный выход:** Как преждевременно завершить выполнение цикла `foreach`, не дожидаясь перебора всех элементов коллекции? Приведите пример такой ситуации.

6. **Выбор оператора цикла:** В каких случаях при работе с одномерным массивом следует предпочесть цикл `foreach`, а в каких необходимо использовать цикл `for`?

Список используемых источников

1. **Шилдт Г.** C# 4.0: полное руководство. — М.: Вильямс, 2019. — 1056 с. *(Глава 5: Управляющие операторы; Глава 7: Массивы и строки — подробный разбор цикла `foreach`).*

2. **Троелсен Э., Джепикс Ф.** Язык программирования C# и платформа .NET. — СПб.: Диалектика, 2022. — 1312 с. *(Раздел: Конструкции итерации `for` и `foreach`).*

3. **Албахари Д., Албахари Б.** C#. Справочник. Полное описание языка. — СПб.: БХВ-Петербург, 2023. — 1088 с. *(Спецификация инструкции `foreach` и перечислителей).*

4. **Павловская Т. А.** C#. Программирование на языке высокого уровня: Учебник для вузов. — СПб.: Питер, 2020. — 432 с. *(Глава: Обработка массивов с использованием цикла `foreach`).*

5. **Официальная документация Microsoft Learn:** Инструкция `foreach` (справочник по C#) [Электронный ресурс]. — URL: microsoft.com (дата обращения: 13.11.2026).