JOURNAL OF UNIVERSAL SCIENCE RESEARCH

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2024 SJIF 2024 = 5.073/Volume-3, Issue-1

C# DASTURLASH TILIDA KOLLEKSIYALARGA MASSIV SIFATIDA MUROJAAT QILISH: INDEKSATORLAR

Zikirova Fotima Murtazoyevna,
Buxoro davlat universiteti 1-2ATT-22 guruh talabasi
fotimazikirova4@gmail.com

Annotatsiya. *Ushbu maqola C# dasturlash tilida indeksatorlar bilan ishlashni o'rganish va qo'llash haqida batafsil ma'lumot beradi. Indeksatorlar orqali kolleksiyalarga massivlar kabi murojaat qilish imkoniyati yaratiladi. Maqola indeksatorlarni aniqlash va ulardan foydalanish jarayonini misollar yordamida tushuntiradi. Shuningdek, string turidagi indekslar yordamida massivga murojaat qilish imkonini beruvchi* ***KeyedArray*** *sinfi yaratilib, uning ishlash mantiqi va amaliyotda qo‘llanilishi batafsil yoritilgan. Ushbu usullar C# dasturlashida kolleksiyalarni boshqarishning qulay va samarali usullarini yaratishga yordam beradi.*

Kalit so’zlar: *C# dasturlash tili, indeksatorlar, kolleksiyalar, massivga murojaat qilish, string indekslar, KeyedArray sinfi, Find metodi, FindEmpty metodi, ma'lumotlar boshqaruvi, obyektlar kolleksiyasi, indeksatorlarni ortiqcha yuklash, konstruktorlar.*

ОБРАЩЕНИЕ К КОЛЛЕКЦИЯМ КАК К МАССИВАМ В ЯЗЫКЕ ПРОГРАММИРОВАНИЯ C#: ИНДЕКСАТОРЫ

Аннотация. *В данной статье подробно рассматривается работа с индексаторами в языке программирования C#. Индексаторы предоставляют возможность обращаться к коллекциям так же, как к массивам. Статья объясняет процесс определения и использования индексаторов с помощью примеров. Также создается класс KeyedArray, который позволяет обращаться к массиву с использованием строковых индексов, подробно освещается его логика работы и применение на практике. Эти методы способствуют созданию удобных и эффективных способов управления коллекциями в программировании на C#.*

Ключевые слова: *язык программирования C#, индексаторы, коллекции, доступ к массиву, строковые индексы, класс KeyedArray, метод Find, метод FindEmpty, управление данными, коллекция объектов, перегрузка индексаторов, конструкторы.*

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2024 SJIF 2024 = 5.073/Volume-3, Issue-1

ACCESSING COLLECTIONS AS ARRAYS IN C# PROGRAMMING LANGUAGE: INDEXERS

Annotation. *This article provides detailed information about working with indexers in the C# programming language. Indexers allow collections to be accessed similarly to arrays. The article explains the process of defining and using indexers with examples. Additionally, a KeyedArray class is created, which enables accessing an array using string indices, with its logic and practical application described in detail. These methods help create convenient and efficient ways to manage collections in C# programming.*

Keywords: *C# programming language, indexers, collections, array access, string indices, KeyedArray class, Find method, FindEmpty method, data management, object collections, overloading indexers, constructors.*

Kirish. Massiv elementlariga kirish juda oddiy va tushunarlidir: container[n] buyrug'i container massivining n-elementiga kirishni ta'minlaydi. Indekslar yordamida boshqa turdagi kolleksiyalarga ham murojaat qilish mumkin.

C# tili indekslash operatsiyasining o'z dasturini yozishga imkon beradi. Dastlab bunday xususiyatga ega bo'lmagan kolleksiyalarga indeks orqali murojaat qilish imkoniyatini berishingiz mumkin. Bundan tashqari, siz indeks sifatida nafaqat int turini, balki string kabi boshqa turlarni ham indekslashingiz mumkin

Indeksator formati

Indeksator odatiy xususiyatga juda o'xshash ko'rinadi, bundan tashqari, o’zgaruvchi nomi o'rniga *this* kalit so'z va [ ] indeks operatori paydo bo'ladi:

```
class MyArray
{
   public string this[int index]
   {
     get
     {
         return array[index] ;
     }
     set
     {
```

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2024 SJIF 2024 = 5.073/Volume-3, Issue-1

```
        array[index] = value;
       }
    }
}
```

Sahna ortida s=myArray [i]; ifodasi i indeksi qiymatini unga o'tkazish orqali get kirish metodini chaqiradi. myArray[i]="string"; ifodasi i indeksi va "string" qatori qiymat sifatida uzatiladigan set kirish metodini chaqirishga olib keladi.

Indeksator bilan ishlash dasturi

Indekslar int turi bilan cheklanmaydi. Masalan, siz uy kolleksiyasini indeksatsiya qilish uchun ularning egalarining ismlari yoki manzillaridan foydalanishingiz mumkin. Bundan tashqari, indeksator xususiyati bir xil to'plamning turli elementlarini indekslash uchun bir nechta indeks turlari bilan ortiqcha yuklanishi mumkin.

Quyidagi *Indexer* dasturi oddiy massivga o'xshab ko'rinadigan va ishlaydigan *KeyedArray* virtual massiv sinfini yaratadi, bundan tashqari string tipidagi qiymat indeks sifatida qo'llaniladi.

Kerakli sinf sozlamalarini bajarish

Ushbu misol maxsus sinfga asoslangan, ya'ni buning uchun siz sinf ramkasini yaratishingiz kerak.

```
using System;

namespace Indexer
{
public class KeyedArray
{

   private string[] _keys;
   private object[] _arrayElements;

public KeyedArray(int nSize)
   {
```

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2024 SJIF 2024 = 5.073/Volume-3, Issue-1

```
    _keys = new string[nSize] ;
    _arrayElements = new object[nSize] ;
    }
  }
}
```

KeyedArray klassi ikkita oddiy qatorni o'z ichiga oladi. *_arrayElements* massivi haqiqiy KeyedArray ma'lumotlarini o'z ichiga oladi. *_keys* massivida saqlanadigan satrlar obyektlar massivining identifikatorlari sifatida ishlaydi, _keysning i-elementi _arrayelEmentsning i-yozuviga mos keladi. Bu dastur *string* tipidagi indekslar yordamida *KeyedArray* klasini indekslash imkonini beradi.

Butun sonlar bo'lmagan indekslar kalitlar deb ataladi.

public KeyArray(int size) qatori konstruktor deb ataladigan maxsus funktsiyaning boshlanishi hisoblanadi. Konstruktorlarni sinf misollarini yaratish bo'yicha ko'rsatmalar sifatida tasavvur qiling. Endi bu haqda tashvishlanishingiz shart emas, lekin aslida konstruktor _keys va _arrayElements uchun qiymatlarni belgilaydi.

Indeksatorlar bilan ishlash

Endi kodda ishlaydigan indeksatorni aniqlash kerak. Bu quyidagi qismda ko'rsatilgan. E'tibor bering, public object this[string key] indeksatori ikkita funktsiyadan foydalanishni talab qiladi, Find() va Find Empty().

```
private int Find(string targetKey){
  for(int i=O; i<keys.Length; i++)
  {
    if(String.Compare(keys[i], targetKey)==O)
    {
      return i;
    }
  }
return -1;
}

private int FindEmpty()
```

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2024 SJIF 2024 = 5.073/Volume-3, Issue-1

```
{
  for(int i=O; i<keys.Length; i++)
  {
    if(_keys[i]==null)
    {
      return i;
    }
  }
  throw new Exception("Massiv to'ldirilgan");
}

public object this[string key]
{
  set
  {
    int index=Find(key);
    if(index<O)
    {
      index=FindEmpty();
      _keys[index] = key;
    }
    _arrayElements[index] = value;
  }
  get
  {
    int index Find(key);
    if(index < O)
    {
      return null;
    }
    return _arrayElements[index];
  }
}
```

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2024 SJIF 2024 = 5.073/Volume-3, Issue-1

Set[string] indeksatori Find() metodini qo'llagan holda massivda berilgan indeks mavjudligini tekshirishdan boshlanadi. Agar u indeksni qaytarsa, set[] yangi ma'lumotlar obyektini tegishli _arrayElements elementida saqlaydi. Agar Find() kalitni topa olmasa, set[] bo'sh elementni qaytarish uchun FindEmpty() ni chaqiradi, bu yerda uzatilgan obyekt saqlanadi.

Get[] metodi shunga o'xshash mantiq yordamida indeks bilan ishlaydi. Birinchidan, Find() metodi yordamida ma'lum bir kalitni qidiradi. Agar Find() manfiy bo'lmagan indeksni qaytarsa, get[] so'ralgan ma'lumotlarni saqlaydigan tegishli _arrayElements a'zosini qaytaradi. Agar Find() -1 ni qaytarsa, get[] metodi *null* qiymatini qaytaradi, bu ro'yxatda berilgan kalit yo'qligini ko'rsatadi.

Find() metodi _ keys massivining barcha elementlari bo'ylab aylanib, string turidagi targetKey qiymati bilan bir xil qiymatga ega elementni qidiradi. Find() metodi topilgan elementning indeksini qaytaradi(yoki element topilmasa -1 qiymati). FindEmpty() metodi bog'langan kalit elementga ega bo'lmagan birinchi element indeksini qaytaradi.

Yangi sinfni sinovdan o'tkazish

Indexer dasturining bir qismi bo'lgan, lekin sinfning bir qismi bo'lmagan Main() funksiyasi KeyedArray sinfining ahamiyatsiz qo'llanilishini ko'rsatadi:

```
public class Program
{
  public static void Main(string[] args)
  {

    KeyedArray family = new KeyedArray(100);

   family ["Bart"]=8;
   family ["Lisa"]=10;
   family ["Maggie"]=2;

    Console.WriteLine("Lisani qidiramiz:");
    int age=(int)family["Lisa"];
```

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2024 SJIF 2024 = 5.073/Volume-3, Issue-1

```
        Console .WriteLine("Lisaning yoshi - {10}", age);


        Console.WriteLine("Dasturni tugatish uchun <Enter> ni bosing!");
        Console.ReadLine();
    }
}
```

Birinchidan, dastur 100 uzunlikdagi (ya'ni yuzta erkin element bilan) KeyedArray tipidagi family obyektini yaratadi. Bundan tashqari, ushbu obyekt bolalarning yoshini indeks sifatida ismlardan foydalangan holda saqlaydi. Va nihoyat, dastur family["Lisa"] iborasi yordamida Lisaning yoshini oladi va uni ekranga chiqaradi.

E'tibor bering, dastur family[] dan qaytarilgan qiymat uchun turni o'zgartirishni amalga oshirishi kerak, chunki KeyedArray har qanday turdagi obyektlarni saqlashi mumkin bo'lgan tarzda yozilgan. Agar indeksator faqat int tipidagi qiymatlar bilan ishlashi uchun yozilgan bo'lsa yoki KeyedArray umumlashtirilgan sinf bo'lsa, bunday turdagi transformatsiyasiz amalga oshirish mumkin. Dastur natijasi quyidagicha:
Lisani qidiramiz:
Lisaning Yoshi – 10
Dasturni tugatish uchun <Enter> ni bosing!

Xulosa: Maqolada indeksatorlar orqali kolleksiyalarga oson va intuitiv tarzda murojaat qilish imkoniyati yaratish usullari ko‘rib chiqildi. Indeksatorlar yordamida massiv elementlariga xos bo'lgan container[n] sintaksisi boshqa turdagi kolleksiyalar uchun ham qo'llanishi mumkinligi aniqlandi. Maqolada ishlab chiqilgan **KeyedArray** sinfi string turidagi indekslar orqali ma'lumotlarga kirish imkonini beradi. Ushbu yondashuv nafaqat int turidagi indekslar, balki boshqa turlardan ham foydalanishga yo‘l ochadi. Misollar asosida indeksatorlar bilan ishlashda kerakli funksiyalar, xususan Find va FindEmpty metodlari taqdim etildi. Ushbu yondashuv katta hajmdagi ma'lumotlarni boshqarishda qulaylik yaratadi va dasturchilarga samaradorlikni oshirish imkonini beradi.

ADABIYOTLAR:

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2024 SJIF 2024 = 5.073/Volume-3, Issue-1

1. Albahari, J., & Albahari, B. (2022). C# 11 in a Nutshell: The Definitive Reference (11th ed.). O'Reilly Media.
2. Troelsen, A., & Japikse, P. (2021). Pro C# 9 with .NET 5: Foundational Principles and Practices in Programming (10th ed.). Apress.
3. Microsoft. (n.d.). Indexers (C# Programming Guide). Microsoft Learn. Retrieved January 23, 2025, from https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/indexers/
4. Balena, F. (2023). Using Indexers in C#: A Practical Guide. CodeProject. Retrieved January 23, 2025, from https://www.codeproject.com/Articles/5293917/Using-Indexers-in-Csharp
5. GeekforGeeks. (2023). Indexers in C#. Retrieved January 23, 2025, from https://www.geeksforgeeks.org/indexers-in-c-sharp/
6. Smith, J. (2024). Advanced Data Structures in C#: Indexers and Collections. International Journal of Computer Science and Applications, 36(2), 145–156.