Özet

- **GİRİŞ**
- Yazılım geliştiriciler birbiri ardına yardıkları kodlarda farklı değişkenler tanımlama ihtiyaçları duyarlar. Genellikle değişkenlere tek bir değer atanmaktadır. Fakat bazı durumlarda aynı tip özelliğini taşıyan değişkenleri bir arada tutmamız gerekebilir. Bu konuda yazılımcılar aynı tipteki değişkenlerin bir arada ve tek bir isimle saklayabilecekleri bir sınıf olan *dizileri (array)* kullanmaktadırlar.
- Diziler, ortak adlarla anılan aynı tipteki verilerin topluluğudur. Diziler bir programlama dilindeki en önemli veri yapılarından biridir. Dizilerden span tipleride kullanılmaktadır. Bir dizi üzerinden iki farklı metot yazmak yerine span kullanarak tüm özellikleri bir ortak tip üzerinden gerçekleştirebilirsiniz. Aslında span hafıza üzerinde belirli bir alanı görebilen bir pencere olarak düşünülebilir.
- **DİZİLER**
- Değişkenlere sadece bir değer atandığı bilinmektedir. Fakat bazı durumlarda aynı veri tipindeki değişkenleri bir arada tutmamız gerekebilir. İşte bu durumlarda ayın türdeki değişkenleri tek bir isimle saklayabileceğimiz diziler ortaya çıkmaktadır. Diziler kodlar kısmında "*array*" olarak ifade edilmektedir. Diziler kısaca aynı türdeki verilerin ortak bir isimle anılması olarak ifade edilir (). Diziler programlama dilinde önemli bir yer sahiptir. Aynı zamanda dizileri aynı türdeki verilerin hep bir arada olduğu değişkenler olarak düşünebilirsiniz. Örneğin 8 adet meyve ismini tek bir liste içerisinde tutmak istersek bir dizi kullanabiliriz.
- **Dizi Oluşturma**
- Bir dizi oluşturulurken boş köşeli parantez ve bir değişken ismi ile takip edilen elemanların ortak tipleri tanımlatılarak oluşturulur. İki farklı yol ile dizi tanımlanması yapılabilir.
  - **1.yol:**
    degiskenturu[ ] dizi_ismi = new degiskenturu [elemansayisi]
- Yukarıdaki durumda, ilk olarak dizini değişken türü belirlenir. Değişken türü dizinin içerisinde yer alan elemanların veri türünü belirler. "degiskenturu" ifadesinden sonra köşeli parantez olduğuna dikkat edin. Köşeli parantezler burada bir boyutlu bir dizinin tanımlandığını belirtmektedir. Köşeli parantezler içerisinde yer alan "elemansayisi" ile dizi içerisinde ne kadar eleman tutulacağını belirtmektedir.
  - **2. yol:**
  - *int[ ] finalnotları;*
  - *sayilar = new int[5];*
- Eğer integer değişken türünde eleman içeren bir dizi tanımlanmak isteniyor yukarıdaki gibi yapılabilir. Yukarıda "sayilar" adlı dizi içinde 5 adet int türünde veri tutulabilir.
- **Diziye Değer Girme**
- Dizi tanımlandıktan sonra diziye değer girilmesi gerekir. Diziye değer girme işlemi dizi tanımlama aşamasında veya program akışı esnasında gerçekleştirilebilir. Dizi tanımlandıktan sonra dizi içerisinde yer alan her bir elemana *indeks* değerleriyle ulaşılıp ve değer atamaları yapılabilir.
- Bir dizi içerisinde yer alan elemanlara dizi indeksi yardımıyla tek tek erişilebilir. Dizi indeksi *array index* olarak ifade edilmektedir. Dizi indeksi, bir elemanın dizi içerisindeki konumu olarak ifade edilir. Programlama dillerinde genellikle ilk dizi indeksi sıfır (0)'dır.
- **Diziyi Yazdırma**
- Bir diziye ulaşılmak istenildiğinde dizinin *indeks numarası* ile ulaşılabilinmektedir. Diziye değer girmenin yanında dizinin ekrana yazdırılması da dizinin özellikleri arasında yer almaktadır. Değerlerin ekrana yazdırılması işlemi aşağıdaki gibi yapılmaktadır.

Özet (devamı)

- **Bazı Dizi Özellikleri ve Metotları**
- Dizileri, .Net Framework içinde tanımlı *Array* sınıf temsil eder. Tüm diziler Array sınıfında tanımlı özellikleri ve metotları kullanır. Bu metot ve özelliklerden en sık kullanılanları şunlardır;
- **Lenght**
- Bir dizinin saklayabileceğin toplam eleman sayısını Lenght özelliği verir. Bu değer *integer* türünde verilir. "*dizi_adi.Lenght*" olarak kullanılmaktadır.
- **Clear**
- Dizinin belirlenen indeksler arasındaki tüm değerleri temizler. Temizleme işleminde verilen değerler, dizi elemanlarını tiplerine göre değişir. Örneğin string türündeki bir dizi temizlenirse elemanlar "" (boş yazı) değerini alır eğer integer türündeki dizi temizlenirse elemanlar sıfır (0) değerini alır.
- **Reverse**
- Dizinin eleman sırasını tersine çeviren bir özelliktir. Diziye tanımlı tüm elemanların veya belirli indeks aralığındaki elemanların sırasını tersine çevirir.
- **SPAN**
- .*Net Core 2.1* ile beraber gelen ve özellikle .*Net Core 3.0* ve sonrası frameworkler tarafından oldukça fazla kullanılan bir tiptir. Span tipi bellekte bulunan bir bölgeye bir tür ve hafıza olarak erişmemizi sağlayan bir değer türüdür. Basit haliyle span bellek üzerinde belirli bir alanı gören bir pencere olarak düşünülebilir. Bu pencere yardımıyla tüm diziyi de görebilir veya dizini belirli bir kısmını (indeks) da görebilirsiniz.
- **Stackalloc ile Span**
- Stackalloc ile stack üzerinde üretilen dizilere de Span ile erişebiliriz. .Net Core 2.1'den önce stackalloc kullandığımızda unsafe kod yazmak gerekmekteydi. Fakat şuan unsafe kod yazmadan span ile bu alana erişebilmemiz mümkündür.
- **Span Kısıtlamaları**
- Span kavramı yapı gereği bir *ref struct* olduğu için bazı kısaltmalara sahiptir. Aynı zamanda ref struct olması nedeniyle boxingi önlemek amacıyla object, dynamic veya interface variablelarına atanmamaktadır. Bu yüzden spanı kullanamıyorsanız *Memory<T>* veya *ReadOnlyMemory<T>* tiplerini kullanabilirsiniz.

DEĞERLENDİRME SORULARI

1. Dizilerin kapasiteleri tanımlanırken kullanılan karakter aşağıdakilerden hangisidir?
   a) { }
   b) < >
   c) [ ]
   d) ( )
   e) | |

2. int[ ] sayilar= new int[5] şeklinde tanımlanan bir dizi için aşağıda verilenlerden hangisi kesinlikle yanlıştır?
   a) Sayilar dizisinin son elemanı 5. indekse sahiptir.
   b) Sayilar dizisinin ilk elemanı 0. indekse sahiptir
   c) Sayilar dizisinin son elemanı 4. indekse sahiptir.
   d) Sayilar dizisi maksimum 5 eleman barındırabilir.
   e) Sayilar dizisi 3 eleman barındırabilir.

3. Dizi içerisindeki elemanları silmeye yarayan metot aşağıdakilerden hangisidir?
   a) IndexOf
   b) Reverse
   c) Sort
   d) Clear
   e) Lenght

4. Dizi içerisinde elemanları tersten sıralamaya yarayan metot aşağıdakilerden hangisidir?
   a) IndexOf
   b) Sort
   c) Reverse
   d) Clear
   e) Lenght

5. Dizide yer alan eleman sayılarını gösteren metot aşağıdakilerden hangisidir?
   a) IndexOf
   b) Sort
   c) Clear
   d) Lenght
   e) Reverse

6. Dizi indeksi aşağıdakilerden hangisi olarak ifade edilmektedir?
   a) Array index
   b) index
   c) array
   d) index array
   e) in array

7. Bellekte bulunan bir bölgeye bir tür veya hafıza olarak erişmemizi sağlayan tip aşağıdakilerden hangisidir.
   a) Array
   b) Span
   c) Index
   d) Reverse
   e) Clear

8. Aynı türdeki değişkenlerin ortak bir isimle anılması aşağıdakilerden hangisidir?
   a) Diziler
   b) Reverse
   c) Clear
   d) Lenght
   e) Span

9. Span tipi hangi Core sürümünden sonra daha fazla kullanılmaya başlanılmıştır?
   a) .Net Core 3.0
   b) .Net Core 1.9
   c) .Net 3.09
   d) .Net Core 2.2
   e) .Net Core 1.8

10. .Net Core 2.1 den önce stackalloc kullandığımızda hangi kod bloğu gerekliydi ?
    a) Array
    b) Unsafe
    c) Reverse
    d) Sort
    e) Clear

Cevap Anahtarı
1.c, 2.a, 3.d, 4.c, 5.d, 6.a, 7.b, 8.a, 9.a, 10.b