ERCİYES ÜNİVERSİTESİ MÜHENDİSLİK FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ PROGRAMMING LABORATORY NOTLARI

6. HAFTA NOTU (C# Programlama)

Bu not dersin 6. haftasında yapacağınız uygulamaya hazırlanmanızı sağlayacak olup temel kavramları anlatan bilgileri içermektedir.

DERSİN ÖĞRETİM ÜYESİ

Prof. Dr. Bahriye AKAY

DENEYİ YAPTIRAN ÖĞRETİM ELEMANI

Arş. Gör. Hayri ÖZTÜRK

2026
KAYSERİ

1-) Deneyde Kullanılacak Kütüphaneler Hakkında Kısa Bilgi

1.1 System

System kütüphanesi C# dilinde en temel kütüphanelerden biridir. Bu kütüphane içerisinde programın çalışması için gerekli olan temel sınıflar ve veri türleri bulunmaktadır. Konsol uygulamalarında kullanıcı ile etkileşim kurmak, matematiksel işlemler yapmak, tarih ve zaman bilgilerini elde etmek ve çeşitli temel işlemleri gerçekleştirmek için bu kütüphaneden yararlanılır.

System kütüphanesi içerisinde bulunan Console sınıfı, program ile kullanıcı arasında metin tabanlı iletişim kurulmasını sağlar. Console sınıfı sayesinde ekrana bilgi yazdırılabilir ve kullanıcıdan veri alınabilir. Örneğin Console.WriteLine() metodu ekrana metin veya sonuç yazdırmak için kullanılırken, Console.ReadLine() metodu kullanıcıdan metin biçiminde veri almak için kullanılır.

System kütüphanesi ayrıca matematiksel işlemler için kullanılan Math sınıfını da içerir. Math sınıfı mutlak değer alma, yuvarlama işlemleri gibi birçok matematiksel işlemin kolayca yapılmasını sağlar. Bunun yanında tarih ve zaman işlemleri için kullanılan DateTime yapısı da bu kütüphane içerisinde yer almaktadır.

1.2 System.Collections.Generic

System.Collections.Generic kütüphanesi, generic koleksiyon veri yapılarının bulunduğu kütüphanedir. Generic koleksiyonlar aynı türden çok sayıda verinin düzenli ve güvenli bir şekilde saklanmasını sağlar. Generic yapıların en önemli özelliği veri türünün önceden belirlenmesidir. Bu sayede koleksiyon içerisine yanlış veri türü eklenmesi engellenir.

Bu kütüphane içerisinde en yaygın kullanılan veri yapılarından biri List<T> koleksiyonudur. List veri yapısı dinamik bir veri yapısıdır ve eleman sayısı çalışma zamanında değiştirilebilir. Liste içerisine yeni elemanlar eklenebilir, elemanlar silinebilir ve liste elemanları üzerinde işlem yapılabilir. Bu özellikler sayesinde List veri yapısı veri kümeleri ile çalışırken oldukça kullanışlı bir yapı sunar.

1.3 System.Linq

System.Linq kütüphanesi, koleksiyonlar üzerinde sorgulama yapılmasını sağlayan bir kütüphanedir. LINQ ifadesi “Language Integrated Query” kavramının kısaltmasıdır ve sorgulama mantığının doğrudan programlama diline entegre edilmesini ifade eder.

LINQ kullanılarak koleksiyonlar üzerinde filtreleme, sıralama, gruplama, veri dönüştürme ve ortalama alma gibi işlemler yapılabilir. Bu işlemler klasik döngüler kullanılarak da

gerçekleştirilebilir; ancak LINQ kullanımı kodun daha kısa, okunabilir ve anlaşılır olmasını sağlar.

LINQ özellikle koleksiyon veri yapıları ile birlikte kullanıldığında veri üzerinde güçlü sorgulama işlemlerinin yapılmasına olanak sağlar.

2-) Deneyde Kullanılacak Programlama Yapıları Hakkında Kısa Bilgi

2.1 Namespace Yapısı

C# dilinde namespace, sınıfları ve diğer program bileşenlerini mantıksal gruplar altında toplamak için kullanılan bir yapıdır. Namespace kullanımı büyük projelerde kod düzeninin korunmasına yardımcı olur ve aynı isimde sınıfların çakışmasını önler.

Namespace yapısı sayesinde bir projede bulunan sınıflar belirli bir mantıksal yapı içerisinde organize edilir. Bu durum kodun okunabilirliğini artırır ve proje yönetimini kolaylaştırır.

2.2 Sınıf (Class)

Sınıf, nesne yönelimli programlamanın temel yapı taşıdır. Bir sınıf, gerçek dünyadaki bir varlığın program içerisindeki soyut temsilidir. Sınıflar veri ve davranışları bir arada tutar. Veri, sınıf içerisindeki özellikler ile ifade edilirken davranışlar metotlar aracılığıyla gerçekleştirilir.

Sınıflar programın modüler bir yapıda geliştirilmesini sağlar. Bu sayede program daha okunabilir ve yönetilebilir bir hale gelir.

2.3 Nesne (Object)

Nesne, bir sınıfın bellekte oluşturulmuş örneğidir. Aynı sınıftan birden fazla nesne üretilebilir ve bu nesnelerin her biri farklı değerlere sahip olabilir. Nesneler sınıf içerisinde tanımlanmış özellikleri taşır ve sınıfta bulunan metotları kullanabilir.

2.4 Özellikler (Properties)

C# dilinde sınıf içerisindeki verilere erişmek için property yapıları kullanılır. Property'ler genellikle get ve set anahtar kelimeleri ile tanımlanır.

get anahtar kelimesi ilgili özelliğin değerinin okunmasını sağlar.
set anahtar kelimesi ise özelliğin değerinin değiştirilmesine olanak tanır.

Property kullanımı veri kapsülleme (encapsulation) ilkesine katkı sağlar. Bu sayede sınıf içerisindeki verilere erişim kontrollü bir şekilde gerçekleştirilebilir.

2.5 Erişim Belirleyiciler

C# dilinde sınıf ve üyelerin erişim seviyelerini belirlemek için erişim belirleyiciler kullanılır.

public erişim belirleyicisi ilgili yapının programın diğer bölümlerinden erişilebilir olduğunu ifade eder. private erişim belirleyicisi ise ilgili üyenin yalnızca tanımlandığı sınıf içerisinde kullanılabileceğini belirtir.

Bu yaklaşım veri gizleme ilkesini destekler ve yazılım güvenliğini artırır.

2.6 Kurucu Metot (Constructor)

Kurucu metotlar, bir sınıftan nesne oluşturulduğu anda otomatik olarak çalışan özel metotlardır. Kurucu metotlar sınıf ile aynı isme sahiptir ve geri dönüş değeri bulunmaz.

Kurucu metotların temel amacı nesnenin başlangıç durumunu hazırlamaktır. Bu sayede bir nesne oluşturulduğu anda gerekli başlangıç değerleri atanabilir.

3-) Temel Veri Türleri

3.1 string

string veri türü metinsel ifadeleri saklamak için kullanılır. İsim, başlık veya kategori gibi metinsel bilgiler string veri türü ile tutulur.

3.2 int

int veri türü tam sayı değerlerini temsil eder. Sayısal sayaçlar, yıl bilgileri ve menü seçimleri gibi değerler int veri türü ile ifade edilir.

3.3 double

double veri türü ondalıklı sayıları saklamak için kullanılır. Puan, ortalama ve oran gibi hassas sayısal değerler double veri türü ile tutulur.

3.4 bool

bool veri türü yalnızca true veya false değerlerini alabilir. Program akışının devam edip etmeyeceğini kontrol etmek için kullanılır.

3.5 object

object veri türü C# dilindeki tüm veri türlerinin temel üst türüdür. Farklı veri türlerinin genel bir yapı içerisinde tutulması gerektiğinde kullanılabilir.

3.6 dynamic

dynamic veri türü değişkenin veri türünün derleme zamanında değil çalışma zamanında belirlenmesini sağlar. Bu yapı özellikle anonim nesneler ile çalışırken esneklik sağlar.

4-) Koleksiyon Yapıları

4.1 List<T>

List veri yapısı aynı türden çok sayıda elemanı saklamak için kullanılan dinamik bir koleksiyon yapısıdır. Liste içerisine yeni elemanlar eklenebilir ve liste elemanları üzerinde dolaşılabilir.

5-) LINQ Yapıları

5.1 Lambda İfadeleri

LINQ sorgularında kullanılan kısa fonksiyon ifadelerine lambda ifadeleri denir. Lambda ifadeleri genellikle => operatörü kullanılarak yazılır ve bir koleksiyon elemanının hangi özelliği üzerinde işlem yapılacağını belirtir.

5.2 Where()

Where metodu bir koleksiyon içerisinden belirli bir koşulu sağlayan elemanları seçmek için kullanılır.

5.3 OrderBy()

OrderBy metodu koleksiyon elemanlarını artan sırada sıralamak için kullanılır.

5.4 OrderByDescending()

OrderByDescending metodu koleksiyon elemanlarını azalan sırada sıralamak için kullanılır.

5.5 Select()

Select metodu koleksiyon elemanlarını farklı bir veri yapısına dönüştürmek için kullanılır.

5.6 GroupBy()

GroupBy metodu koleksiyon elemanlarını belirli bir özelliğe göre gruplamak için kullanılır.

5.7 Average()

Average metodu sayısal değerlerin ortalamasını hesaplamak için kullanılır.

5.8 Take()

Take metodu koleksiyonun ilk belirli sayıda elemanını almak için kullanılır.

5.9 Any()

Any metodu koleksiyon içerisinde eleman olup olmadığını kontrol etmek için kullanılır.

5.10 ToList()

ToList metodu LINQ sorgularının sonucunu liste yapısına dönüştürmek için kullanılır.

6-) Matematiksel İşlemler

6.1 Math.Abs()

Math.Abs metodu bir sayının mutlak değerini hesaplamak için kullanılır.

6.2 Math.Round()

Math.Round metodu ondalıklı sayıları belirli basamaklara göre yuvarlamak için kullanılır.

7-) Metin İşlemleri

7.1 ToLower()

ToLower metodu bir metindeki tüm karakterleri küçük harfe dönüştürür.

7.2 Contains()

Contains metodu bir metin içerisinde belirli bir alt ifadenin bulunup bulunmadığını kontrol eder.

7.3 Equals()

Equals metodu iki değerin eşit olup olmadığını kontrol etmek için kullanılır.

7.4StringComparison.OrdinalIgnoreCase

Bu yapı metinlerin büyük-küçük harf farkı dikkate alınmadan karşılaştırılmasını sağlar.

8-) Veri Dönüştürme

8.1 int.TryParse()

int.TryParse metodu metinsel bir değeri tam sayıya dönüştürmek için kullanılır.

8.2 double.TryParse()

double.TryParse metodu metinsel bir değeri ondalıklı sayıya dönüştürmek için kullanılır.

9-) Tarih ve Zaman

9.1 DateTime.Now.Year

DateTime yapısı sistem tarih ve saat bilgilerine erişmeyi sağlar. DateTime.Now.Year ifadesi sistemin güncel yıl bilgisini verir.

10-) LINQ Gruplama Yapıları

10.1 IEnumerable

IEnumerable koleksiyonlar üzerinde dolaşmayı sağlayan temel arayüzlerden biridir.

10.2 IGrouping

IGrouping yapısı GroupBy işlemi sonucunda oluşan veri gruplarını temsil eder.

Ders Katkıları

Bu uygulama kapsamında kullanılan yapılar, öğrencinin aşağıdaki teknik kazanımları edinmesini sağlamaktadır:

1. Nesne yönelimli programlama mantığını kavrama

- Sınıf (class) yapılarının gerçek dünyadaki varlıkları temsil etmek için nasıl kullanıldığını öğrenme
- Nesnelerin özellikler (properties) aracılığıyla veri taşıma mantığını anlama
- Nesne yönelimli tasarım yaklaşımının kod okunabilirliği ve sürdürülebilirliği üzerindeki etkilerini gözlemleme

2. Koleksiyon veri yapıları ile çalışma deneyimi kazanma

- Aynı türden birden fazla veriyi koleksiyon yapıları içinde tutma mantığını öğrenme
- Dinamik veri yapılarının sabit dizilere göre sağladığı avantajları gözlemleme
- Koleksiyonlar üzerinde veri ekleme, veri saklama ve veri yönetme işlemlerini anlama

3. Veri üzerinde sorgulama ve filtreleme işlemleri yapabilme

- Belirli koşullara göre veri seçme ve veri kümesini daraltma yöntemlerini öğrenme
- Veri kümesi içerisinden yalnızca belirli özelliklere sahip kayıtları elde etme becerisi kazanma
- Veri filtreleme işlemlerinin performans ve veri analizi açısından önemini kavrama
- Koşul temelli veri seçimi ile daha anlamlı sonuçlar üretmeyi öğrenme

4. Veri sıralama ve düzenleme tekniklerini öğrenme

- Veri kümesini belirli ölçütlere göre sıralama yöntemlerini kavrama
- Büyük veri kümelerinde sıralama işlemlerinin kullanıcı deneyimine etkisini anlama
- Verilerin artan veya azalan düzende düzenlenmesinin analiz süreçlerindeki rolünü öğrenme
- Sıralama işlemlerinin veri sunumu ve raporlama süreçlerindeki önemini fark etme

5. Algoritmik düşünme becerisini geliştirme

- Veri üzerinde gerçekleştirilen işlemleri belirli bir mantık sırasına göre planlamayı öğrenme
- Problemleri küçük adımlara bölerek çözme yaklaşımını kavrama
- Veri seçme, sıralama, dönüştürme ve gruplama işlemlerini bir arada kullanabilme becerisi kazanma
- Yazılım geliştirme sürecinde sistematik problem çözme yeteneğini geliştirme