LINQ Alıştırmaları ve Ödevleri-2

Bu ödevde, bir sosyal medya uygulamasının backend'inde çalıştığınızı hayal edin. Göreviniz, kullanıcılar, gönderiler, yorumlar ve takipler arasındaki ilişkileri kullanarak platformun veri akışından anlamlı raporlar ve içgörüler çıkarmak.

Tüm Ödevlerde Kullanılacak Veri Seti

```
public class User
{
    public int Id { get; set; }
    public string Username { get; set; }
    public DateTime JoinDate { get; set; }
    public string City { get; set; }
}

public class Post
{
    public int Id { get; set; }
    public int UserId { get; set; }
    public string Content { get; set; }
    public DateTime Timestamp { get; set; }
}

public class Comment
{
    public int Id { get; set; }
    public int PostId { get; set; }
    public int UserId { get; set; } // Yorumu yapan kullanıcı
    public string Content { get; set; }
}

// Many-to-Many ilişkisi
public class Follow
{
    public int FollowerId { get; set; }  // Takip eden
    public int FollowingId { get; set; } // Takip edilen
}

// Data Lists
```

```
List<User> users = new List<User>
{
    new User { Id = 1, Username = "ahmet_y", JoinDate = new
DateTime(2025, 1, 15), City = "İstanbul" },
    new User { Id = 2, Username = "zeynep_k", JoinDate = new
DateTime(2024, 3, 20), City = "Ankara" },
    new User { Id = 3, Username = "can_d", JoinDate = new DateTime(2025,
8, 10), City = "İzmir" },
    new User { Id = 4, Username = "elif_s", JoinDate = new DateTime(2023,
11, 5), City = "Ankara" }
};

List<Post> posts = new List<Post>
{
    new Post { Id = 101, UserId = 1, Content = "Bugün hava harika!
#istanbul", Timestamp = DateTime.Now.AddDays(-2) },
    new Post { Id = 102, UserId = 2, Content = "Yeni projemiz için çok
heyecanlıyım. #gündem", Timestamp = DateTime.Now.AddDays(-5) },
    new Post { Id = 103, UserId = 1, Content = "LINQ öğrenmek çok
keyifli.", Timestamp = DateTime.Now.AddHours(-5) },
    new Post { Id = 104, UserId = 4, Content = "Ankara'da sonbahar bir
başka...", Timestamp = DateTime.Now.AddDays(-1) },
    new Post { Id = 105, UserId = 2, Content = "Backend development
dünyası sürekli gelişiyor.", Timestamp = DateTime.Now.AddDays(-10) }
};

List<Comment> comments = new List<Comment>
{
    new Comment { Id = 201, PostId = 102, UserId = 1, Content =
"Başarılar dilerim Zeynep!" },
    new Comment { Id = 202, PostId = 102, UserId = 4, Content = "Merakla
bekliyoruz!" },
    new Comment { Id = 203, PostId = 103, UserId = 2, Content =
"Kesinlikle katılıyorum Ahmet." },
    new Comment { Id = 204, PostId = 101, UserId = 3, Content =
"Gerçekten de öyle." },
    new Comment { Id = 205, PostId = 104, UserId = 2, Content =
"Ankara'yı özledim..." }
};

List<Follow> follows = new List<Follow>
{
```

```
    new Follow { FollowerId = 1, FollowingId = 2 }, // Ahmet, Zeynep'i
takip ediyor
    new Follow { FollowerId = 1, FollowingId = 4 }, // Ahmet, Elif'i
takip ediyor
    new Follow { FollowerId = 2, FollowingId = 1 }, // Zeynep, Ahmet'i
takip ediyor
    new Follow { FollowerId = 4, FollowingId = 2 }  // Elif, Zeynep'i
takip ediyor
};
```

Ödev 1: “Ana Sayfa Akışı” Simülasyonu

Senaryo: Bir kullanıcının ana sayfa akışını oluşturacaksınız. Bu akış, kullanıcının takip ettiği kişilerin en son attığı gönderileri içermelidir.
İstenenler:

1. Kullanıcı ID’si 1 (“ahmet_y”) için bir akış oluşturun.
2. `follows` listesinden bu kullanıcının takip ettiği kişilerin ID’lerini bulun.
3. `posts` listesinden, bu ID’lere sahip kullanıcıların attığı tüm gönderileri bulun.
4. Bu gönderileri en yeniden en eskiye doğru (`Timestamp`'e göre) sıralayın.
5. Sonuç olarak gönderinin içeriğini ve sahibinin `Username`'ini ekrana yazdırın.
   İpucu: `follows.Where(...).Select(...)` ile takip edilen ID’leri bir listeye alın. Sonra `posts.Where(p => followedIds.Contains(p.UserId))` gibi bir yapı kullanın.
   Beklenen Çıktı:

```
Kullanıcı: elif_s, Gönderi: Ankara'da sonbahar bir başka...
Kullanıcı: zeynep_k, Gönderi: Yeni projemiz için çok heyecanlıyım. #gündem
Kullanıcı: zeynep_k, Gönderi: Backend development dünyası sürekli gelişiyor.
```

Ödev 2: En Popüler Gönderiyi Bulma (Yoruma Göre)

Senaryo: Platformdaki en “etkileşimli” gönderiyi, yani en çok yorum alan gönderiyi bulmak istiyoruz.
İstenenler:

1. `comments` listesini `PostId`'ye göre gruplayın.
2. Her bir gönderi için toplam yorum sayısını (`Count`) hesaplayın.
3. Bu sonuçları yorum sayısına göre azalan şekilde sıralayıp en üsttekini alın.
4. En çok yorum alan gönderinin ID’sini, içeriğini ve yorum sayısını ekrana yazdırın.
   Beklenen Çıktı:

```
En Popüler Gönderi: "Yeni projemiz için çok heyecanlıyım. #gündem" (2 yorum)
```

Ödev 3: Etkileşim Raporu (Kullanıcı Bazında)

Senaryo: Her kullanıcının platformdaki aktivitesini (toplam gönderi ve toplam yorum sayısı) özetleyen bir rapor oluşturmak istiyoruz.
İstenenler:

1. Tüm `users` listesi üzerinde gezinin.
2. Her bir kullanıcı için, o kullanıcının `posts` listesindeki gönderi sayısını ve `comments` listesindeki yorum sayısını hesaplayın.
3. Sonuçları “Kullanıcı Adı, Gönderi Sayısı, Yorum Sayısı” formatında bir anonim tip listesi olarak oluşturun.
4. Bu listeyi, en çok gönderi paylaşandan en aza doğru sıralayarak ekrana yazdırın.
   Beklenen Çıktı:

```
Kullanıcı: ahmet_y, Gönderi Sayısı: 2, Yorum Sayısı: 1
Kullanıcı: zeynep_k, Gönderi Sayısı: 2, Yorum Sayısı: 2
Kullanıcı: elif_s, Gönderi Sayısı: 1, Yorum Sayısı: 1
Kullanıcı: can_d, Gönderi Sayısı: 0, Yorum Sayısı: 1
```

Ödev 4: “Influencer” Tespiti (Takipçiye Göre)

Senaryo: Platformda en çok takipçisi olan kullanıcıyı bulmak istiyoruz.
İstenenler:

1. `follows` listesini, takip edilen kişiye (`FollowingId`) göre gruplayın.
2. Her bir `FollowingId` için takipçi sayısını (`Count`) hesaplayın.
3. Bu sonuçları takipçi sayısına göre azalan şekilde sıralayıp en üsttekini alın.
4. En çok takipçisi olan kullanıcının `Username`'ini ve takipçi sayısını ekrana yazdırın.
   Beklenen Çıktı:

```
En Çok Takip Edilen Kullanıcı: zeynep_k (2 takipçi)
```

Ödev 5: Karşılıklı Takipleşme (“Arkadaşlar”)

Senaryo: Birbirini karşılıklı olarak takip eden kullanıcı çiftlerini bulmak istiyoruz.
İstenenler:

1. `follows` listesindeki her bir `(A, B)` takibi için, `(B, A)` takibinin de listede olup olmadığını kontrol edin.
2. Eğer karşılıklı takip varsa, bu çiftin `Username`'lerini ekrana yazdırın. Her çiftin yalnızca bir kez listelendiğinden emin olun.
   İpucu: `follows.Where(f1 => follows.Any(f2 => f2.FollowerId == f1.FollowingId && f2.FollowingId == f1.FollowerId))` gibi bir yapı kurabilirsiniz. Tekrarı önlemek için `f1.FollowerId < f1.FollowingId` gibi bir ek koşul ekleyebilirsiniz.
   Beklenen Çıktı:

```
Karşılıklı Takipleşenler: ahmet_y <-> zeynep_k
```

Ödev 6: “Pasif” Kullanıcılar

Senaryo: Platforma üye olmuş ancak hiç gönderi paylaşmamış kullanıcıları tespit etmek istiyoruz.
İstenenler:

1. `users` listesi üzerinde, `posts` listesinde kendisine ait (`Id == UserId`) hiçbir kaydın (`Any`) olmadığı kullanıcıları filtreleyin.
2. Bulunan kullanıcıların `Username`'ini ve üyelik tarihlerini ekrana yazdırın.
   Beklenen Çıktı:

```
Hiç Gönderi Paylaşmamış Kullanıcı: can_d, Üyelik Tarihi: 10.08.2025
```

Ödev 7: Şehirlere Göre Aktivite

Senaryo: Hangi şehrin platformda daha aktif olduğunu, şehirlerdeki toplam gönderi sayısına göre analiz etmek istiyoruz.
İstenenler:

1. `posts` ve `users` listelerini birleştirerek her gönderinin hangi şehirden atıldığını bulun.
2. Bu birleştirilmiş sonucu şehirlere göre gruplayın.
3. Her şehir için toplam gönderi sayısını (`Count`) hesaplayın.
4. Sonuçları en çok gönderi atılan şehirden en aza doğru sıralayarak ekrana yazdırın.
   Beklenen Çıktı:

```
Şehir: İstanbul, Toplam Gönderi: 3
Şehir: Ankara, Toplam Gönderi: 2
```

Ödev 8: Gönderi ve Yorumları Birleştirme

Senaryo: Bir gönderiye tıklandığında, gönderinin içeriğini ve ona yapılmış tüm yorumları (yorum yapanın adıyla birlikte) göstermek istiyoruz.
İstenenler:

1. ID’si 102 olan gönderiyi bulun.
2. Bu gönderiye ait tüm yorumları `comments` listesinden filtreleyin.
3. Bu yorumları `users` listesi ile birleştirerek her yorumu yapan kullanıcının `Username`'ini bulun.
4. Önce gönderinin içeriğini, ardından alt alta “Yorum Yapan: [Username], Yorum: [Content]” formatında yorumları yazdırın.
   Beklenen Çıktı:

```
Gönderi: Yeni projemiz için çok heyecanlıyım. #gündem
--- Yorumlar ---
Yorum Yapan: ahmet_y, Yorum: Başarılar dilerim Zeynep!
Yorum Yapan: elif_s, Yorum: Merakla bekliyoruz!
```

Ödev 9: Hashtag Analizi

Senaryo: “#gündem” hashtag’ini içeren tüm gönderileri ve bu gönderilere yapılan tüm yorumları bulmak istiyoruz.
İstenenler:

1. `posts` listesinde `Content`'i “#gündem” içeren gönderileri bulun.
2. Bu gönderilerin ID’lerini bir listeye alın.
3. `comments` listesinden `PostId`'si bu listede bulunan tüm yorumları bulun.
4. Bulunan yorumların içeriklerini ekrana yazdırın.
   Beklenen Çıktı:

```
"#gündem" Konulu Gönderilere Yapılan Yorumlar:
- Başarılar dilerim Zeynep!
- Merakla bekliyoruz!
```

Ödev 10: En Son Etkileşim

Senaryo: Platformdaki en son etkileşimin ne olduğunu (en son atılan gönderi mi, yoksa en son yapılan yorum mu) ve ne zaman olduğunu bulmak istiyoruz.
İstenenler:

1. `posts` listesindeki en son `Timestamp` değerini bulun.
2. `comments` listesi için de en son etkileşim zamanını bulmanız gerekiyor. Ancak `Comment` sınıfında `Timestamp` yok. Bu bir tuzak! Öğrenciden `Comment` sınıfına bir `Timestamp` özelliği eklemesini ve veri setini güncellemesini isteyin. Bu, problemin tanımını anlama ve modeli geliştirme yeteneğini test eder.
3. *Güncellenmiş modelle:* `comments` listesindeki en son `Timestamp` değerini bulun.
4. Bu iki zaman değerinden daha yeni olanı bularak “Platformdaki son aktivite [tarih] tarihinde bir [gönderi/yorum] idi.” şeklinde bir mesaj yazdırın.
   Beklenen Çıktı (Model güncellendikten sonra, veriye göre):

```
Platformdaki son aktivite 16.10.2025 13:50:00 tarihinde bir gönderi idi.
```