ALGORİTMA ANALİZİ-ÖDEV 5

Kaba Kod

```
1. Başla
2. Bir sayı al
3. "faktoriyel" adında bir değişken oluştur ve 1 ile başlat
4. Sayının faktöriyelini bul:
   4.1.    1'den başlayarak sayıya kadar tüm sayıları çarp
5. Faktöriyeli ekrana yazdır
6. Bitir
```

ALGORİTMANIN ANALİZİ

1. Özyinelemeli (Recursive) Algoritma

```
static int FaktoriyelRecursive(int n)
{
    if (n <= 0)                                                    //  n
        return 1;                                                  // 1
    else
        return n * FaktoriyelRecursive(n - 1);                     // n-1
}                                                                  //n+1+n-1= 2n
```

- Bu çözümde bir fonksiyon kendini çağırır (recursive olarak) ve her seferinde bir önceki sayının faktöriyelini kullanarak hesaplama yapar.
- Recursive çağrılar fonksiyonun kendi çalışma yığını (stack) içinde depolanır. Bu nedenle büyük sayılar için bellek kullanımı ve performans açısından bazı dezavantajlar olabilir.
- Recursive çözüm, kodun daha temiz görünmesini sağlar ve bazı problemlerde anlaşılması daha kolay olabilir.
- Ancak, recursive çözüm, derin recursive çağrılarla karşılaşabileceği için zaman ve bellek açısından daha maliyetli olabilir.
- Zaman karmaşıklığı O(n) olacaktır, ancak bellek kullanımı ve stack derinliği daha fazla olabilir, bu da maliyeti artırır.

2. Yinelemeli (Iteratif) Algoritma

```
Console.WriteLine("Iteratif Çözüm:");

    for (int i = 1; i <= sayi; i++)                                //1+n+1 +n
    {
        faktoriyel *= i;                                          // n+1
    }
                                                                  //1++n+1+n+n+1=3n+3
```

- Bu çözümde bir döngü kullanılır. Döngü, 1'den başlayarak sayıya kadar tüm sayıları çarpar ve sonuç olan faktöriyeli hesaplar.
- Zaman ve hafıza açısından, döngüde her bir adımda sadece bir değişkenin değeri güncellenir, bu nedenle zaman ve hafıza açısından etkili bir çözümdür.
- Döngü, sayı kadar kez çalıştığı için zaman karmaşıklığı O(n) olacaktır.

ÖZET

- Her iki çözüm de bir sayının faktöriyelini hesaplamak için farklı yaklaşımlar kullanır.
- İteratif çözümde, döngü kullanılarak her bir adımda bir değişken güncellenir. Bu, zaman ve hafıza açısından etkili bir çözümdür.
- Recursive çözüm ise daha temiz bir kod sunar ve anlaşılması kolaydır. Ancak, derin recursive çağrılarla zaman ve bellek açısından maliyetli olabilir.
- Her iki yaklaşım da n faktöriyelini O(n) zaman karmaşıklığıyla hesaplar.