

Algoritma

- Descartes'e göre Problem Çözme Tekniđi:
 1. Doğruluđu kesin olarak kanıtlanmadıkça, hiçbir şeyi doğru olarak kabul etmeyin; tahmin ve önyargılardan kaçının.
 2. Karşılaştığınız her güçlüđu mümkün olduđu kadar çok parçaya bölün.
 3. Düzenli bir biçimde düşünün; anlaşılması en kolay olan şeylerle başlayıp yavaş yavaş daha zor ve karmaşık olanlara doğru ilerleyin.
 4. Olaya bakışınız çok genel, hazırladığınız ayrıntılı liste ise hiçbir şeyi dışarıda bırakmayacak kadar kusursuz ve eksiksiz olsun.

- Algoritma, belirli bir problemi çözmek veya belirli bir görevi gerçekleştirmek için sistematik bir şekilde izlenmesi gereken, sonlu adımlardan oluşan talimatlar dizisidir.
- Bilgisayar bilimlerinde, algoritmalar genellikle veri işleme, sıralama yapılması, arama yapılması, bir sorunun matematiksel çözüme ulaşması, hesaplama ve diğer otomatik görevleri yerine getirmek için kullanılır. Her algoritma, giriş (input) verilerini alır, bu veriler üzerinde belirli işlemler gerçekleştirir ve bir sonuç (output) üretir.

- Bir sorunu çözebilmek için gerekli olan sıralı mantıksal adımların tümüne **algoritma** denir. Doğal dille yazılabileceği için fazlaca formal değildir. Bir algoritma için aşağıdaki ifadelerin mutlaka doğrulanması gereklidir.
 - Her adım son derece belirleyici olmalıdır. Hiç bir şey şansa bağlı olmamalıdır.
 - Belirli bir sayıda adım sonunda algoritma sonlanmalıdır.
 - Algoritmalar karşılaşılabilecek tüm ihtimalleri ele alabilecek kadar genel olmalıdır.

- **Giriş:** Algoritma, başlangıçta belirli bir veri kümesini (giriş) alır. Bu veri işlenmek üzere algoritmaya sunulur.
- **İşlem:** Giriş verileri üzerinde belirli işlemler uygulanır. Bu işlemler sıralı ve belirli kurallara dayalıdır.
- **Çıkış:** İşlemlerin sonucunda elde edilen sonuçlar (çıkış) üretilir. Algoritmanın amacı, verilen bir sorunu çözüme kavuşturmaktır.
- **Belirli Olma:** Algoritmanın her bir adımı açık ve net olmalıdır. Herhangi bir belirsizliğe yer vermemelidir.
- **Sonluluk:** Algoritma sonlu sayıda adımdan oluşmalı ve bir noktada durmalıdır. Sonsuz bir döngüye girmemesi gerekir.

- Algoritma öğrenmek, problem çözme becerilerini geliştirdiği için çok değerlidir. Algoritmalar, verimli ve etkili bir şekilde sorunların nasıl çözüleceğini anlamamıza yardımcı olur.
- Bilgisayar programları, algoritmalar üzerine kurulur; bu nedenle, algoritmaları bilmek programlama dillerini ve teknolojilerini daha iyi anlamamızı sağlar. Ayrıca, optimizasyon, veri analizi ve yapay zeka gibi pek çok alanda algoritmaların doğru seçimi ve uygulanması büyük önem taşır.

Algoritma Bilmenin Faydaları

- **Verimli Problem Çözme:** Algoritmalar, karmaşık sorunların nasıl çözüleceğine dair sistematik bir yaklaşım sunar. Bu da daha hızlı, daha doğru ve daha verimli çözümler elde edilmesini sağlar. Özellikle büyük veri setleri veya karmaşık hesaplamalar içeren sorunlar için uygun algoritmalar seçildiğinde zaman ve kaynak tasarrufu sağlanır.
- **İyi Tasarlanmış Yazılım:** Bir yazılımın performansı büyük ölçüde kullanılan algoritmalara bağlıdır. Kötü bir algoritma, yazılımın yavaş çalışmasına veya gereksiz kaynak tüketmesine neden olabilir. Doğru algoritmalar, yazılımın hızını ve verimliliğini artırır.
- **Teknolojik Gelişmelere Uyum Sağlama:** Günümüzde yapay zeka, makine öğrenimi, büyük veri analitiği gibi alanlar hızla gelişiyor ve bu alanlardaki gelişmelerin çoğu algoritmaların iyileştirilmesi sayesinde gerçekleşiyor. Algoritmaların nasıl çalıştığını anlamak, bu teknolojileri daha etkili kullanmanızı ve geliştirmenizi sağlar.
- **Kariyer Gelişimi:** Yazılım mühendisleri, veri bilimciler ve diğer teknoloji profesyonelleri için algoritma bilgisi kritik bir beceridir. Birçok iş görüşmesi, algoritmalar ve veri yapıları üzerine yoğunlaşan teknik sorularla yapılır. İyi bir algoritma bilgisi, profesyonel gelişim için önemli bir araçtır.

- Algoritmalar, bilgisayarlarda kullanılan hemen hemen her teknolojinin temelini oluşturur. İşletim sistemleri, veri tabanları, ağ protokolleri ve yapay zeka gibi teknolojiler, algoritmalar olmadan var olamaz. Örneğin, bir arama motoru algoritması, milyonlarca web sayfası arasından en uygun sonuçları saniyeler içinde bulur. Ağ protokollerinde ise, veri paketlerinin verimli bir şekilde iletilmesi için algoritmalar kullanılır. Yapay zeka ve makine öğreniminde ise algoritmalar, büyük veri kümelerinden anlamlı sonuçlar çıkarılmasına yardımcı olur.

Bilgisayarlarda Kullanılan Diğer Teknolojilerle İlgili Algoritmaların Rolü

- **Veri Tabanları:** Bir veritabanında veri sorgulama, sıralama ve arama işlemleri algoritmalar sayesinde yapılır. Örneğin, SQL veritabanında bir sorgu çalıştırdığınızda, bu sorgunun verimli bir şekilde işlenmesi ve en kısa sürede sonuçların döndürülmesi için indeksleme ve arama algoritmaları kullanılır. Bu algoritmalar, büyük veri kümeleri üzerinde çalışırken verilerin hızlıca bulunmasını sağlar.
- **İşletim Sistemleri:** İşletim sistemleri, bilgisayarda çalışan her işlemi yönetmek için algoritmalar kullanır. Bu algoritmalar, işlem zamanlaması, bellek yönetimi ve kaynak paylaşımı gibi görevlerde kritik rol oynar. Örneğin, işletim sistemleri birden fazla işlemi verimli bir şekilde yönetmek için “zamanlama algoritmaları” kullanır. Bu algoritmalar, CPU’nun işlemleri nasıl sıralayacağını ve hangi işlemin ne zaman çalışacağını belirler.
- **Ağ Protokolleri:** Bilgisayar ağlarında veri iletimi algoritmalarla yönetilir. Örneğin, veri paketlerinin en hızlı ve en güvenilir şekilde iletilmesini sağlamak için yönlendirme algoritmaları kullanılır. Bu algoritmalar, veri paketlerinin ağ üzerinde en kısa ve en verimli yolu bulmasını sağlar.

- **Yapay Zeka ve Makine Öğrenimi:** Yapay zeka, makine öğrenimi ve veri madenciliği gibi alanlarda algoritmalar en kritik bileşenlerdir. Yapay zeka algoritmaları, büyük veri kümelerinden öğrenme yaparak yeni bilgilere ulaşır ve kararlar verir. Örneğin, makine öğrenimi algoritmaları, bir kullanıcının tercihlerini öğrenerek ona en uygun önerileri sunabilir. Bir film izleme platformu, kullanıcıların izleme alışkanlıklarına göre yeni film önerileri sunmak için makine öğrenimi algoritmaları kullanır.
- **Görüntü İşleme:** Bilgisayarla görme ve görüntü işleme teknolojilerinde de algoritmalar önemli rol oynar. Görüntülerin tanınması, sınıflandırılması ve işlenmesi için kullanılan algoritmalar, kameralar veya yapay zeka sistemleri aracılığıyla nesneleri ve yüzleri tanıyabilir. Örneğin, bir güvenlik sistemi, kamera görüntülerini işleyerek yüz tanıma algoritması ile bir kişinin kimliğini doğrulayabilir.

- Bir algoritma eğer bir girdiyi doğru çıktı ile sonlandırıyorrsa **doğru** olarak isimlendirilir. Doğru algoritma verilere ne yapması gerektiğini kesin olarak tanımlar.
- **Yanlış** bir algoritma bazı girdi örneklerinde her zaman sonlanmayabilir ya da yanlış bir cevabı ile sonlanabilir.
- Beklenilenin aksine, eğer hata oranını kontrol edebiliyorsak yanlış algoritmalar bazen yararlı olabilirler.

- **Polinomsal Zamanlı (Polynomial Time) algoritmalar** ile en iyi çözüm bulunabilir. Bir problemin çözümü veya çözümünün doğrulanması, giriş verisinin boyutuna göre polinom büyüklüğünde bir zamanda yapılabiliriyorsa, bu probleme polinomsal zamanda çözülebilir veya doğrulanabilir denir. n boyutlu girdilerde en kötü çalışma süreleri bir k sabit sayısı için $O(n^k)$ olur. (Sıralama algoritmaları)
- **NP-Tam (Nondeterministic Polynomial Time) problemler** için iyi ama muhtemelen en iyi olmayan çözümler önerilebilir. Çözümü bulmak genellikle belirlenimsiz bir süreçtir ve mevcut algoritmalarla polinomsal zamanda çözülüp çözülemeyeceği bilinmemektedir. Bir NP-tam problem için henüz hiç bir polinom-zamanlı algoritma bulunmamıştır; ayrıca henüz hiç kimse herhangi biri için hiç bir polinom-zamanlı algoritmanın var olamayacağını da kanıtlayamamıştır. (Gezgin Satıcı problemi NP-tam bir problemdir ve çözüm yaklaşım algoritmaları ile çözülebilir)

Verimlilik

- Aynı problemi çözmek için verimlilikleri bakımından önemli ölçüde farklı olan çeşitli algoritmalar vardır. Bu farklılıklar donanım ve yazılımla ilgili farklardan daha önemli olabilmektedir.
- Örneğin, araya sokma sıralaması yapan daha yavaş bir bilgisayara (bilgisayar A) karşı birleştirerek sıralama yapan daha hızlı bir bilgisayarı (bilgisayar B) düşünelim. Her biri, 10 milyon sayıyı bir dizide sıralamak zorunda olsun. Varsayalım, A bilgisayarı saniyede 10 milyar komut ve B bilgisayarı saniyede sadece 5 milyon komut gerçekleştirsin, o halde hızı bakımından A bilgisayarı B bilgisayarından 1000 kat hızlıdır. A bilgisayarı için makine dilinde araya sokma sıralaması kodladığını ve sonuçta n girdi için sıralaması için $2n^2$ komut gerektiğini varsayalım. Ayrıca ortalama bir programcının birleştirerek sıralamayı uyguladığını, bir derleyicide yüksek seviyeli bir dil kullandığını, sonuçlanan kodun $50n \log n$ komut sürdüğünü varsayalım. 10 milyon sayıyı sıralamak için B bilgisayarında

$$\frac{50 \cdot 10^7 \log 10^7 \text{ komut}}{10^7 \text{ saniye bana komut}} \approx 1163 \text{ saniye (20 dakikadan daha az)}$$

- A bilgisayarı

$$\frac{2 \cdot (10^7)^2 \text{ komut}}{10^{10} \text{ saniye bana komut}} = 20.000 \text{ saniye (5, 5 saatten daha fazla)}$$

- 100 milyon sayıyı sıraladığımız zaman Araya Sokma Algoritması yaklaşık 23 günden fazla çalışırken, Birleştirerek Sıralama Algoritması 4 saatten az sürmektedir.
- Modern programlama teknoloji ile, algoritma hakkında fazla bilgi sahibi olmadan bazı görevler başarılabilir, fakat iyi bir algoritma bilgisi ile çok daha fazlası başarılabilir.

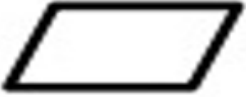
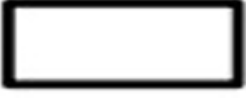
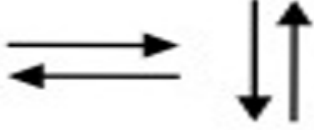
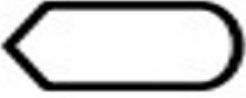
Algoritmalar da;

- Her adım son derece belirleyici olmalıdır. Hiç bir şey şansa bağlı olmamalıdır.
- Belirli bir sayıda adım sonunda algoritma sonlanmalıdır.
- Algoritmalar karşılaşılabilecek tüm ihtimalleri ele alabilecek kadar genel olmalıdır.
- Algoritmada algoritmanın genel işleyişini etkileyebilecek hiç bir belirsizlik olmamalıdır.
- Algoritmada bazı adımlar yer değiştirebilir . Ancak bir çok adım kesinlikle yer değiştiremeyeceğini bilmeliyiz. Yanlış sıradaki adımlar algoritmanın yanlış çalışmasına neden olacaktır.

.

Akış Diyagramı

- Bir algoritmanın daha görsel gösterimidir. Çizgiler, Dörtgen, daire vb. geometrik şekillerle algoritmanın gösterilmesini sağlar. Doğal dille yazılmadığı için daha formal olduğu düşünülebilir.

Simge	İşlev	Simge	İşlev
	Başla/Bitir		Döngü
	Genel Girdi/Çıktı		Manyetik Disk
	Genel İşlem		Yordam Çağırma
	Denetim (Karar)		Akış Yönü
	Yazıcı Çıktısı		Bağlaç
	Görüntü Çıktısı		Sayfa Bağlacı
	Ele ile Girdi (Klavye)		

Pseudo Code (Kaba Kod veya Yalancı Kod veya Söзде Kod)

- Bir algoritmanın yarı programlama dili kuralı, yarı konuşma diline dönük olarak ortaya koyulması/tanımlanmasıdır. Bu şekilde gösterim algoritmayı genel hatlarıyla yansıtır.

Girilen sıcaklığa göre suyun durumunu bildiren programın pseudo kodu;

1. Program açıklama mesajı yaz.
2. Kullanıcın sıcaklığı girmesi için bir uyarı mesajı yaz.
3. Girilen Sıcaklığı Oku.
5. Eğer Sıcaklık < 0 ise Durum="Buz"
6. Eğer Sıcaklık ≥ 100 ise Durum="Buhar"
7. Değilse Durum ="Su"
8. Sonucu Yaz.

Programlama Dili

- Bir Problemin Algoritmik çözümünün Bilgisayara anlatılmasını sağlayan, son derece sıkı-sıkıya kuralları bulunan kurallar dizisidir.

Derleyici

- Bir programlama dili ile bilgisayara aktarılan programın bilgisayarın anlayabileceği Makine Diline çevirmeyi sağlayan ve yazılan programda söz dizim hatalarının olup olmadığını bulan olup olmadığını bulan yazılımlardır.
- Her Programlama dili için bir derleyici olması gerekmektedir.

*Klavyeden girilecek iki sayıdan büyük olanından
küçük olanını çıkarıp sonucu ekrana yazacak
program için bir algoritma?*

Klavyeden girilecek X değerinden N değerine kadar tüm doğal sayıları listeleyen algoritmayı geliştiriniz.

Klavyeden girilen A ve B gibi iki deęerin yerlerini deęiřtirerek ekrana yazan algoritmayı geliřtiriniz

Algoritma Çalıştırma

- Bazen Bir takım algoritmaların ne işe yaradığını anlamak veya algoritmanın doğru çalışıp çalışmadığını test etmek için algoritmayı çalıştırmak gereklidir. Algoritmayı çalıştırmak demek algoritmanın adımlarını sıra ile uygulamak, oluşan değişken değerlerini bir tablo üzerinde göstermek demektir.

.

- Hangi düzey programlama dili olursa olsun tamamı algoritma mantığını aynen kullanır. Değişen sadece yazım kurallarıdır.
- Algoritmalar göz önüne alındığında bir programlama dilinin bir diğerine üstünlüğü yoktur.

Kaynakça

- Ian Sommerville - Yazılım Mühendisliğine Giriş
- Fatih Yücalar, Emin Borandağ - Yazılım Mühendisliğinde Modern Yaklaşımlar
- Thomas H. Cormen vd. - Algoritmalara Giriş