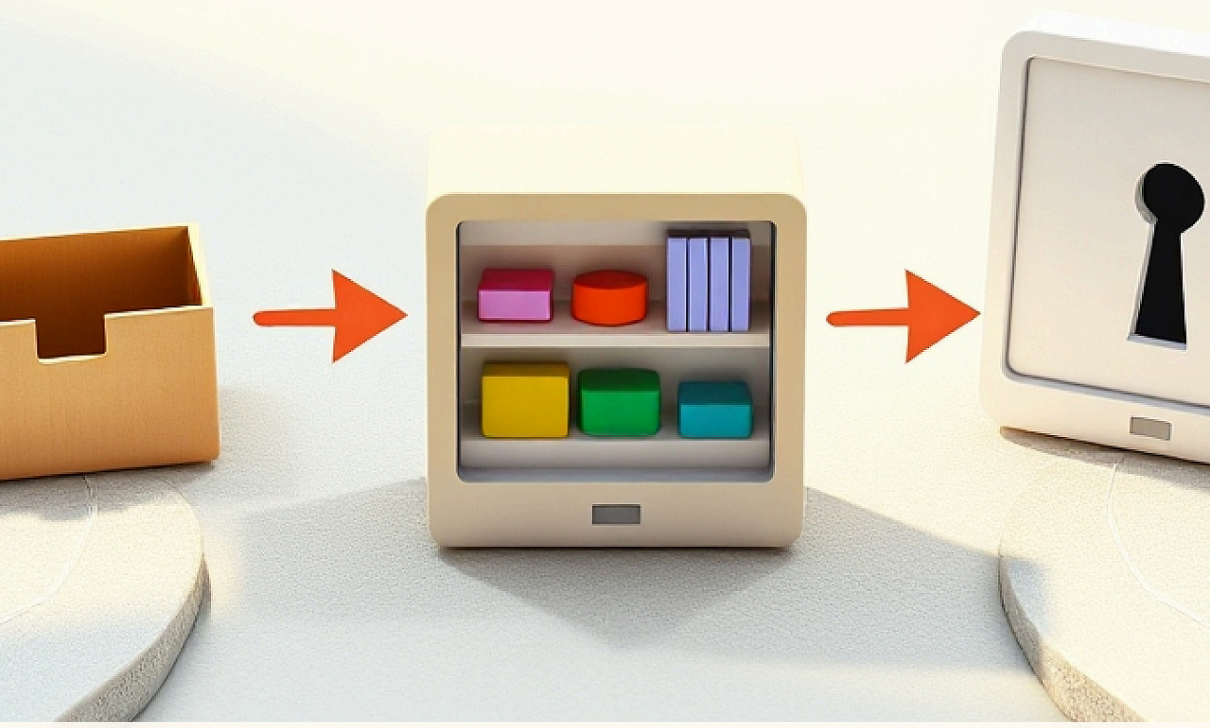
Bir bilgisayar bir listede aradığı bilgiyi nasıl bulur?



www.yzokul.com.tr

Bir bilgisayar bir listede aradığı bilgiyi nasıl bulur?



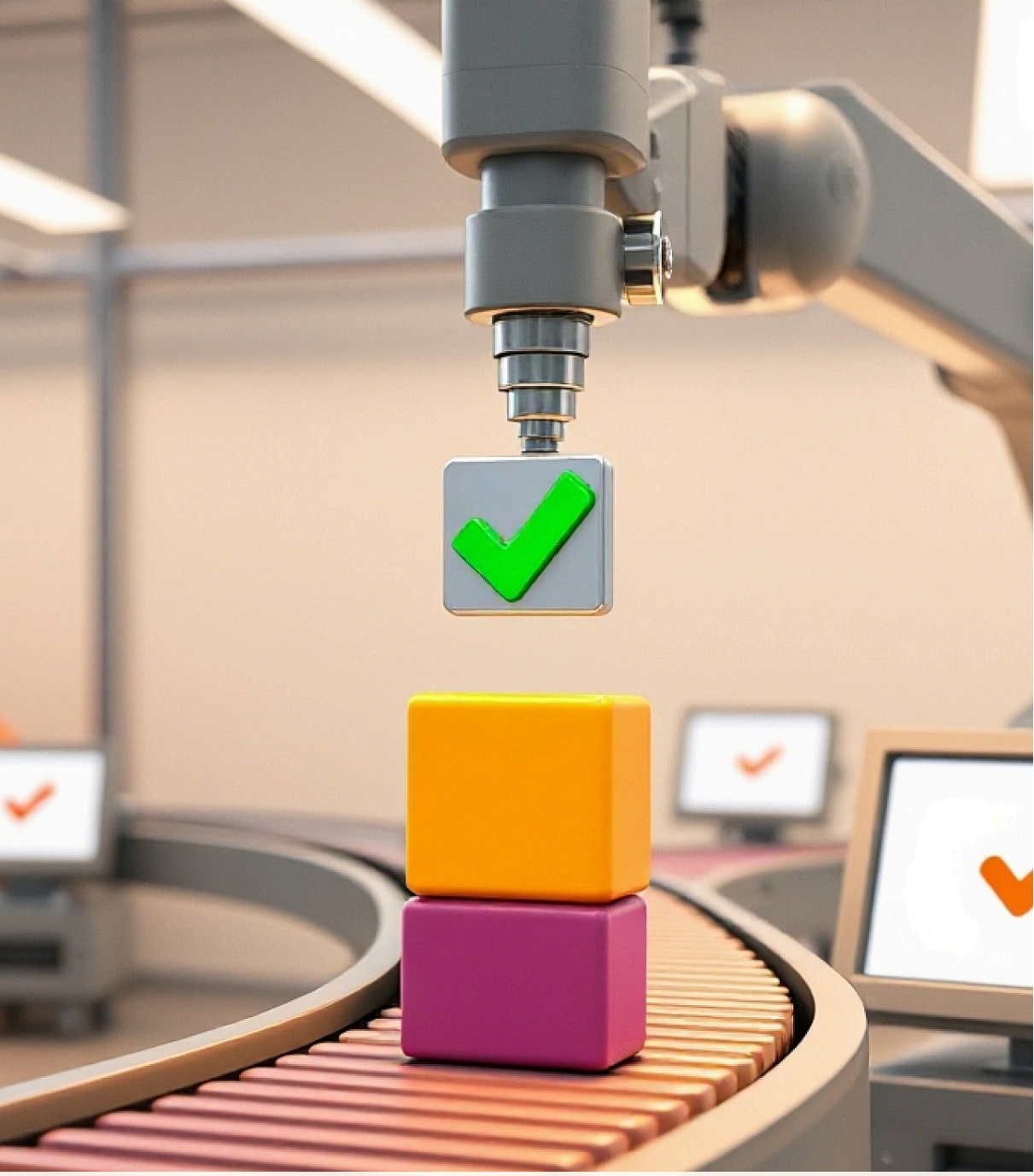
Bir bilgisayarın bir listede bir şeyi nasıl bulduğunu bilmek istiyorsun. Kısa cevap: öğeleri bir yöntemle kontrol eder ve en iyi yöntem listenin nasıl düzenlendiğine bağlıdır. Örneğin, kısa bir listede bir ismi bulmak, doğru olanı görene kadar her ismi kontrol etmek anlamına gelebilir. Ancak çok uzun ve sıralı bir liste daha hızlı bir yönteme olanak tanır. Verinin düzenlenişi seçimi yönlendirir.



Liste, bir kâğıt üzerindeki isimler gibi belirli bir sırada tutulan öğelerdir. Bir bilgisayar da bir listeyi aynı şekilde, bellekte birbiri ardına yerleştirilmiş öğeler olarak saklar. Örneğin, bir meyve listesi elma, muz, kiraz olabilir ve bu sırayla yerleştirilir. Her öğenin bir konumu vardır ve bu, ilkinden başlar. Bu basit düzenleme, tüm aramanın temelidir.



Bir listede bir ismi bulmak için birinci ismi, sonra ikinciyi, sonra üçüncüyü okuyabilir ve istediğini bulana kadar devam edebilirsin. Öğeleri sırayla kontrol etme alışkanlığı, aramanın başlangıç noktasıdır. Örneğin, bir isim listesinde 'Mia'yı arıyorsan, her ismi 'Mia' ile karşılaştırırsın. Bir eşleşme bulduğunda durursun; eşleşme olmadan sona ulaşırsan, isim orada yoktur.



Öğeleri baştan tek tek kontrol etmeye doğrusal arama denir. Hiç sırası olmayan bir liste de dahil olmak üzere her listede çalışır. Örneğin, [3, 8, 7, 1] listesinde 7 sayısını bulmak için doğrusal arama 3'ü, sonra 8'i, sonra 7'yi karşılaştırır ve durur. Sıraya dayanmadığı için basittir ve her zaman doğrudur, ancak uzun listelerde yavaş olabilir.



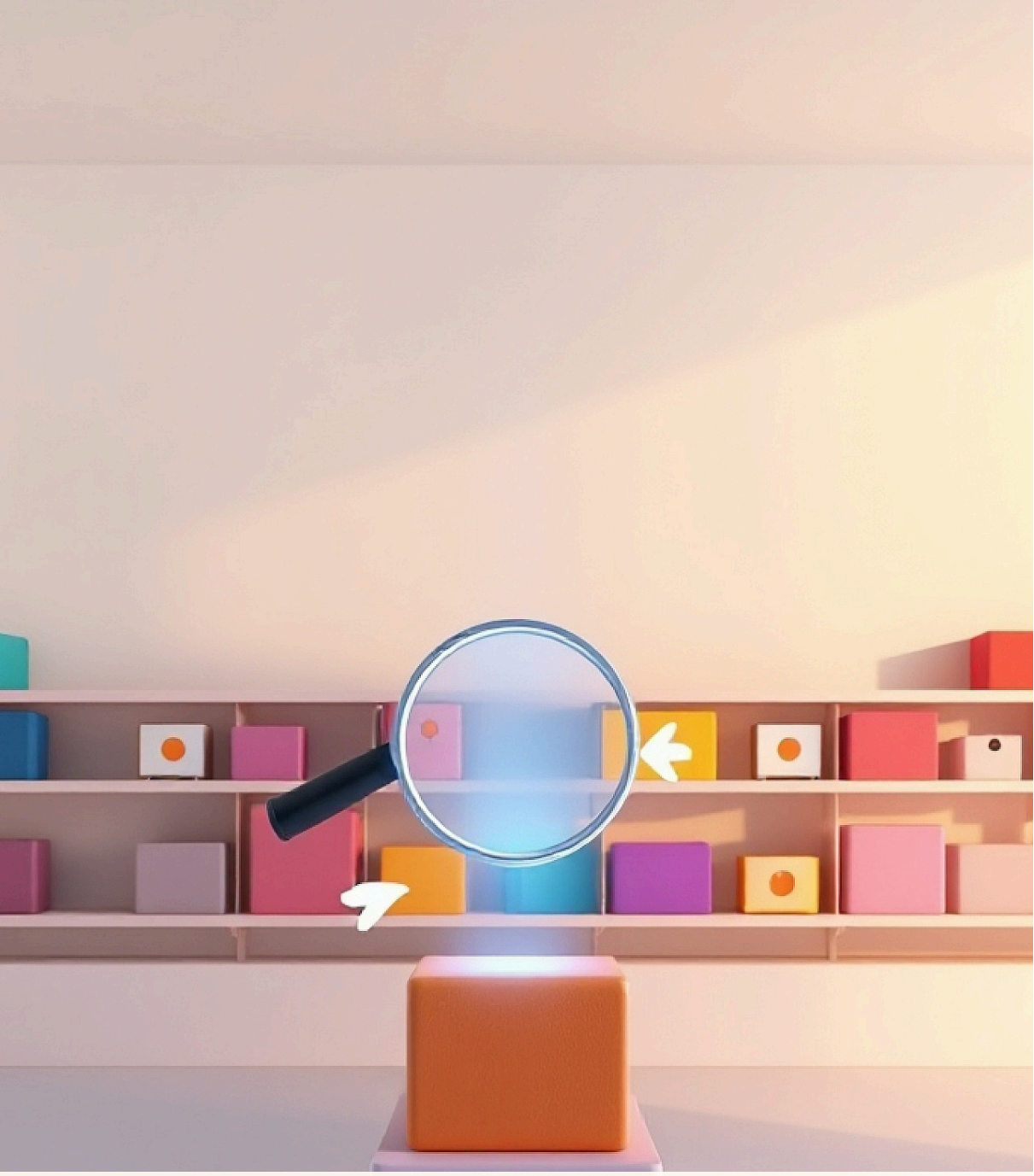
Çok uzun bir listede her öğeyi kontrol etmek uzun zaman alabilir. İstediğin öğe sonuncuysa veya yoksa, bilgisayar her bir öğeye bakmak zorundadır. Örneğin, on bin isimlik bir listede bir ismi aramak on bin karşılaştırma anlamına gelebilir. Her karşılaştırma bir an sürer, bu yüzden toplam süre birikir. Bu yüzden doğrusal arama büyük listeler için yavaş olarak adlandırılır.



Doğrusal arama, öğeyi bulur bulmaz durur. Öğe başa yakınsa arama hızlıdır; sona yakınsa yavaştır. Örneğin, ilk öğeyi bulmak bir karşılaştırma alır, ancak 100 öğelik bir listede son öğeyi bulmak 100 karşılaştırma alır. Ortalama olarak listenin yaklaşık yarısını kontrol edersin. Bu değişkenlik, doğrusal aramanın hızını öngörülemez kılar.



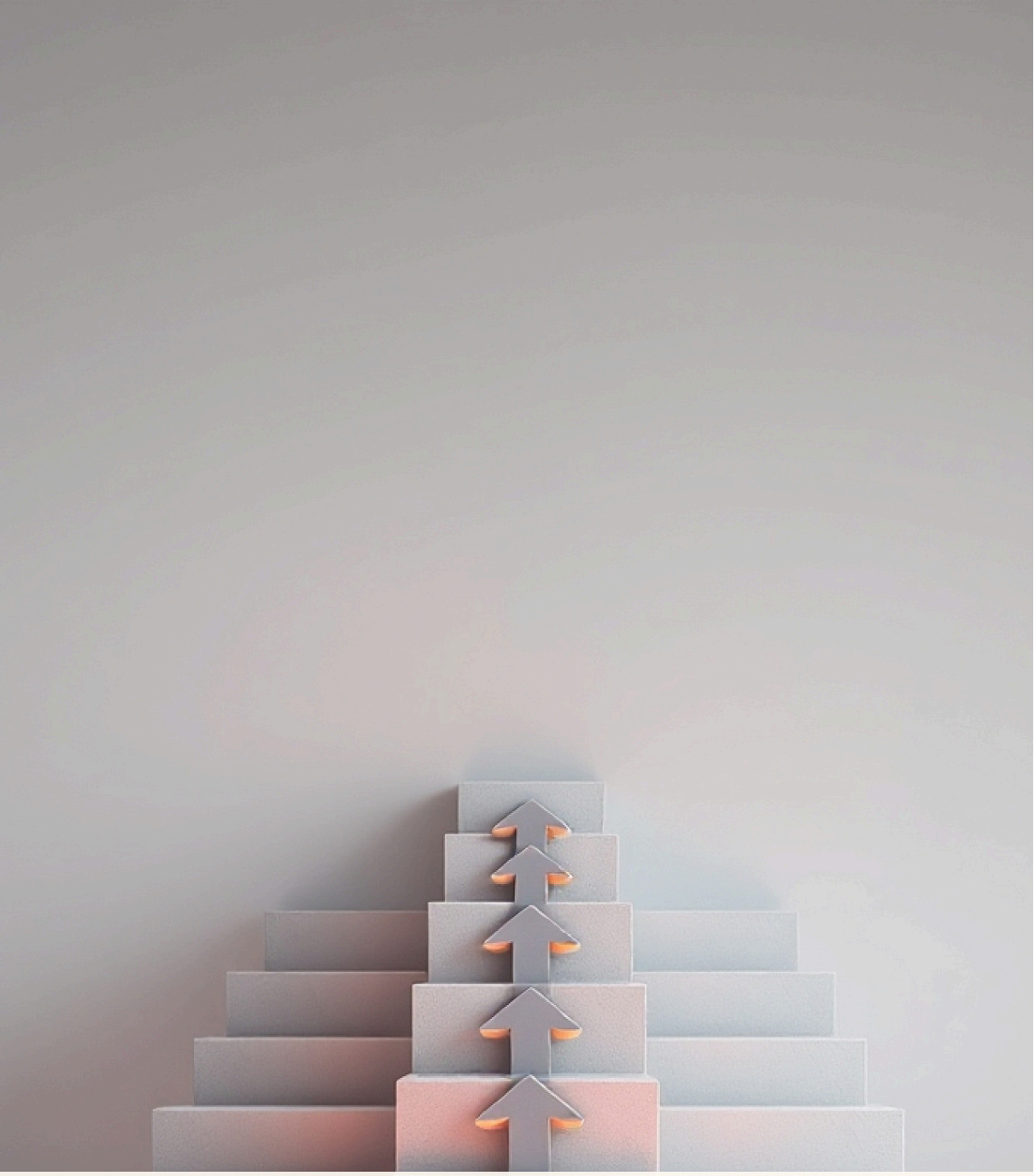
Bir listeyi sıraya koymak, örneğin isimleri alfabetik sıraya dizmek, sıralama olarak adlandırılır. Sıralı bir liste, daha hızlı bir arama yolunun kapısını açar. Örneğin, [1, 3, 5, 7, 9] gibi sıralı bir sayı listesi, sıradan yararlanmanı sağlar. Sıralama çaba gerektirir, ancak bir kez yapıldığında aramalar çok daha hızlı hale gelir. Bu, verimli arama için kilit bir fikirdir.



Daha hızlı bir arama, sıralı bir listenin başından değil ortasından başlar. Orta öğeyi karşılaştırmak, istediğinin hangi yarıda olabileceğini söyler. Örneğin, sıralı [1, 3, 5, 7, 9] listesinde orta 5'tir. 7'yi arıyorsan, 7'nin 5'ten büyük olduğu için sağ yarıda olması gerektiğini bilirsin. Böylece sol yarıyı tamamen yok sayabilirsin.



Sıralı bir listede bir karşılaştırmadan sonra listenin yarısı yok sayılabilir. Arama, kalan yarıda, yine onun ortasını kontrol ederek devam eder. Örneğin, 7'nin 5'ten büyük olduğunu bulduktan sonra sağ yarıya [7, 9] bakarsın. Onun ortası 7'dir ve hedefinle eşleşir, bu yüzden durursun. Her adım olasılıkları yarıya indirir ve aramayı çok hızlı kılar.



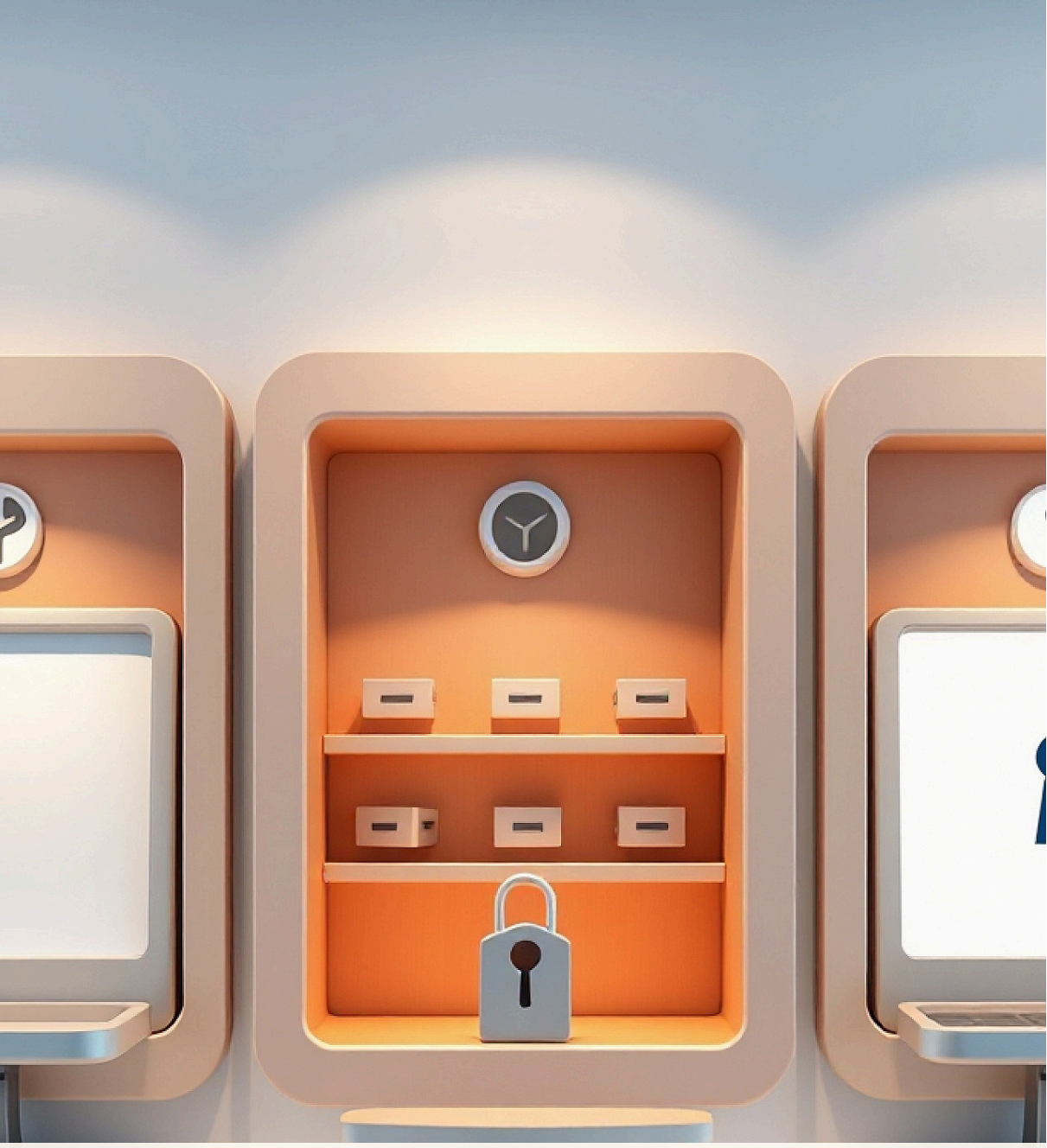
Sıralı bir listeyi tekrar tekrar yarıya indirmeye ikili arama denir. Öğeleri doğrusal aramadan çok daha hızlı bulur çünkü her adım listenin yarısını ortadan kaldırır. Örneğin, 1000 öğelik bir liste ikili aramayla en fazla yaklaşık 10 karşılaştırma alır, doğrusal arama ise 1000 alabilir. Bu hız, listenin sırasını kullanarak büyük bölümleri bir anda elemesinden gelir.



İkili arama yalnızca liste sıralı olduğunda çalışır. Sıralanmamış bir listede yarıyı atmak hata olur çünkü öğe her yerde olabilir. Örneğin, sıralanmamış [3, 8, 7, 1] listesinde orta 8'dir. 7'yi arıyorsan, 8 ile karşılaştırmak sana 7'nin hangi yarıda olduğu hakkında hiçbir şey söylemez. Bu yüzden ikili aramanın doğru olması için sıralı bir liste gerekir.



Sözlük yapısı, öğeleri anahtarlarla saklar, tıpkı bir kelimenin anlamıyla birlikte saklanması gibi. Bilgisayar, eşleşen öğeye doğrudan atlamak için anahtarı kullanır. Örneğin, bir telefon rehberi isimleri (anahtarları) numaralara eşler. Bir isim verildiğinde, her girdiyi kontrol etmeden numarayı hızlıca bulabilirsin. Bu anahtar tabanlı erişim çok hızlıdır çünkü anahtar doğrudan öğeye götürür.



Üç yöntemi karşılaştır: doğrusal arama her listede çalışır, ikili arama sıralı listelerde çalışır ve anahtar tabanlı erişim sözlük yapılarında çalışır. Doğru seçim, verinin nasıl düzenlendiğine ve hangi aramalara ihtiyaç duyduğuna bağlıdır. Örneğin, listen sıralanmamışsa ve nadiren arama yapıyorsan, doğrusal arama uygundur. Sık sık arama yapıyorsan, sıralama ve ikili arama kullanmak daha iyi olabilir.



Yaygın bir hata, sıralanmamış bir listede ikili arama kullanmaktır. Bu yöntemi seçmeden önce verinin sıralı olup olmadığını her zaman kontrol et. Örneğin, [3, 8, 7, 1] üzerinde ikili arama denersen, yarıya indirme mantığı sırayı varsaydığı için hedefi kaçırabilirsin. Bunu önlemek için listenin sıralı olduğunu doğrula veya önce sırala. Bu basit kontrol yanlış sonuçları önler.



Bir başka hata, uygun anahtarlar olmadan anahtar tabanlı erişim beklemektir. Her yöntemin önce karşılanması gereken gereksinimleri vardır. Örneğin, bir sözlük yapısı her öğe için benzersiz anahtarlar gerektirir; anahtarlar eksikse veya yineleniyorsa, aramalar başarısız olabilir. Bu yüzden anahtar tabanlı erişimi kullanmadan önce her öğenin benzersiz bir anahtarı olduğundan emin ol. Gereksinimleri karşılamak, yöntemin doğru çalışmasını sağlar.



Bilgisayarlar listelerde öğeleri karşılaştırarak arama yapar ve yöntem verinin düzenlenişine bağlıdır. Her liste için doğrusal aramayı, sıralı listeler için ikili aramayı ve sözlükler için anahtar tabanlı erişimi unutma. Örneğin, küçük sıralanmamış bir liste için doğrusal arama, büyük sıralı bir liste için ikili arama ve bir telefon rehberi için anahtar tabanlı erişim kullan. Verine ve ihtiyaçlarına göre seç.