**Algoritma tasarlama yöntemleri**

**1.kelimelerle ifade etme**

2.sözde kodla yazma.

Akış şemasını zor problemlerde kullanmak zordur. O yüzden ileri seviyede kullanılmaz.

Sözde kod, sade şekilde kelimeleri yazmaktır.

Algoritma tasarlanığında sınırlı bir süre içerisinde doğruluğunu hesaplamanız gerekir

İspat yöntemlerini **ayrık işlemselde görmüş olmalısınız.** Bu yöntemleri kullanarak algoritma-ları ispat edeceksiniz

Yaklaşım algoritması için doğruluk kavramı kesin algoritmadan daha karmaşıktır. Önceden tanımlanmış bir limiti aşmadığını göstermemiz gerekir.

Algoritma analizi:

1.doğruluk

2.verimlilik

2.1 zaman

2.2 ne kadar alan harcıyor

Algoritma kurarken basitlik istenir.

Bir algoritmanın başka bir olması gereken şey genelliktir. Kabul ettiği şeyler ve girdiler kümesi önemlidir.

İyi bir algoritma tekrarlanan çalışmayla mümkündür

Mükemmellikten feragat etmemiz gerekir.

**Algoritmadan beklentimiz sonucun doğru olup olmadığını söylemesi.**

Kararlı algoritma: i ve j ilemeanlarını koruyaibiliyorsa kararlı bir algodur

Arama algoritmaları:

Bigdata veritabanından bilgi çekebilmek için arama algoritmaları kullanırız.

Graph: kenalarla köşelerin birleşmesi ile yapılan modelleme algosu

Şehirler arasaında gezen bir gezgin satıcı her şehrie bir kere giderek en kısa yolu nasıl bulur(ışın izleme vb bir çok dalı var)

Kombiratoryonel ?

Hesaplamalı algoları:

1.En yakın çit problemi:düzlemde en yakın çifti bulur

2.Dışbükey problemi:kümenin en küçük dışbükey çökkeni bulur

Sayısal alogar:?

Temel veri yapıları:

Bir veri yapısı verilerin organize etmesine yardımcı olur

Hızlı bir anlatım olacak:

Doğrusal veri yapıları:

Bitişik olarak depolanan ve indexli olan diziler, aynı türde olmalıdır.

Dizi ve bağlantılı liste:

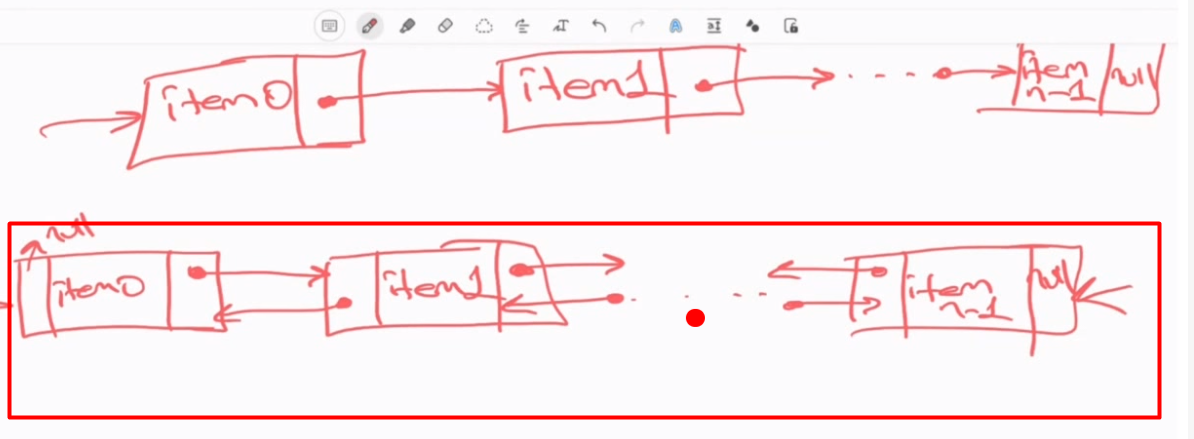
1 ile n arasında gösterilen bir tam sayıdır.

Dizilerin 10 ile 10000. Elemanına aynı anda erişilebileceğinden diziler bağlantılı listeden bu yönden daha iyidir. Diziler üzerinde gerçekleştirilen işlemler diğerlerinden farklıdır

Bağlantılı liste : işaretçi adında bir bağlantısı var

Bir düğümün halefinin yokluğunu belirtmek için null adında işaretçiler kllanılır.

Çift yönlü bağlantılı liste var: gerisine ve ilerisine gidebilen bilgileri tutabiliyo



1Dizi:sonu var: bir öğeyi aramak eklemek ve silmek amaçtır

1.1Yığın:ekleme ve silmenin yanlızca sondan yapılabildiği bir listedir.lifo yani(recursivee de kullanılır)

1.2kuyruk-ilk giren ilk çıkar. Banka sırası gibi.

1.2.1 öncelik kuyruğu : Öncelikli müşteri daha öne gelir

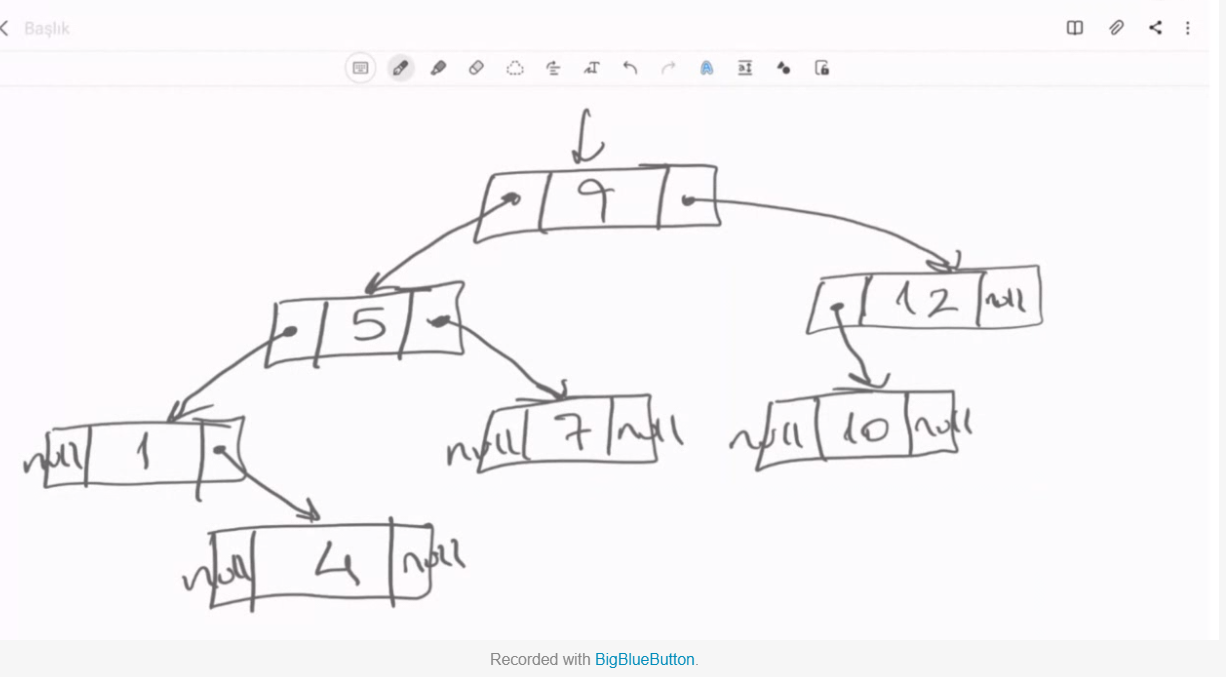
Ağaçar

Root olmayan ağaçlarda vardır ama bilgisayar bilimlerinde rootlu ağaç kullanılır

Özyenilemeli algoların çözümünde yardımcı olur.

Ağaç, düğüm, köşe vb açıklandı

Binary searc tree



Ne kadar karışık olduğu,yüksekliği ile ilgilidir

first child next sibling tree

