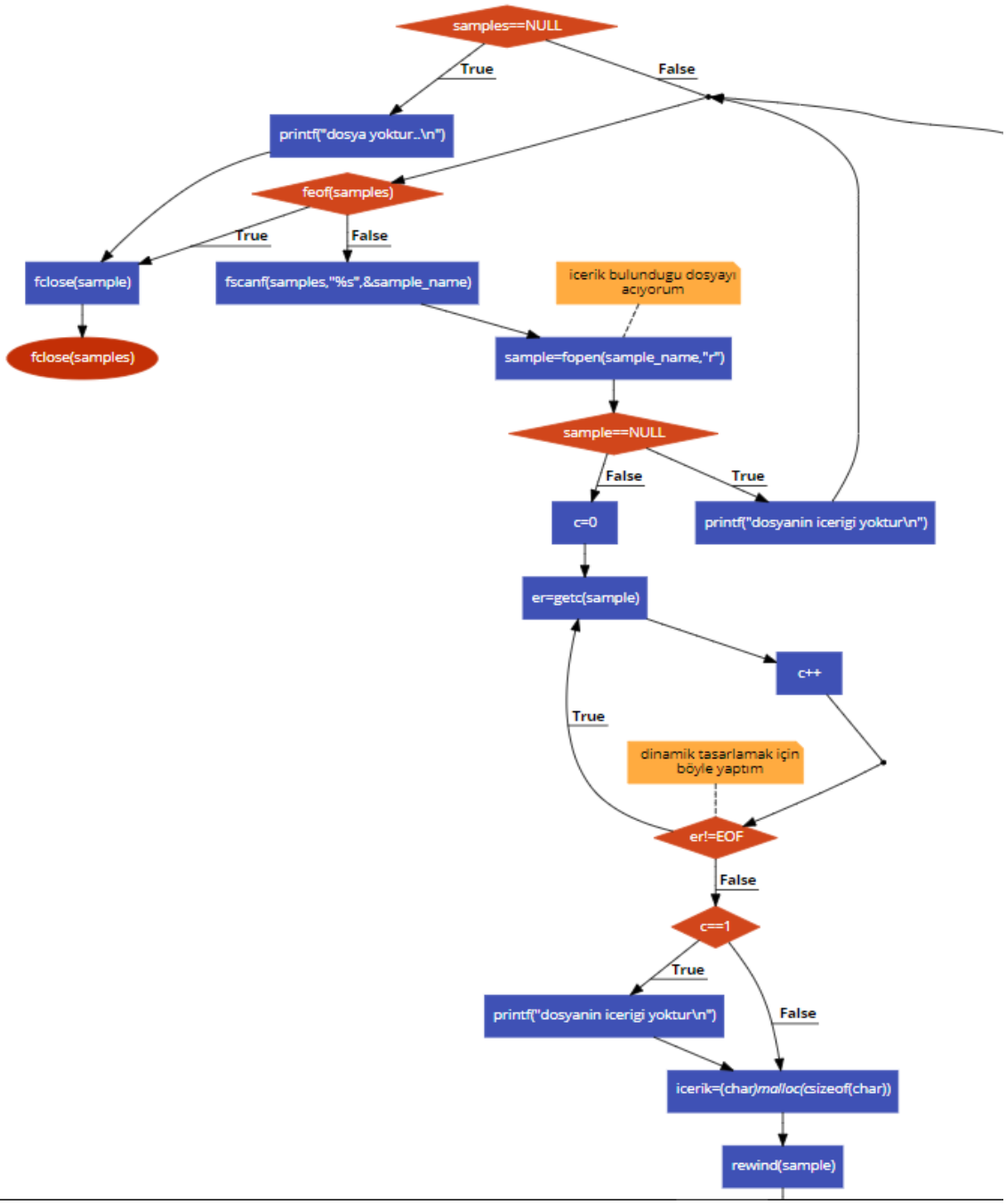
ALGORİTMA ANALİZİ
ÖDEV-2

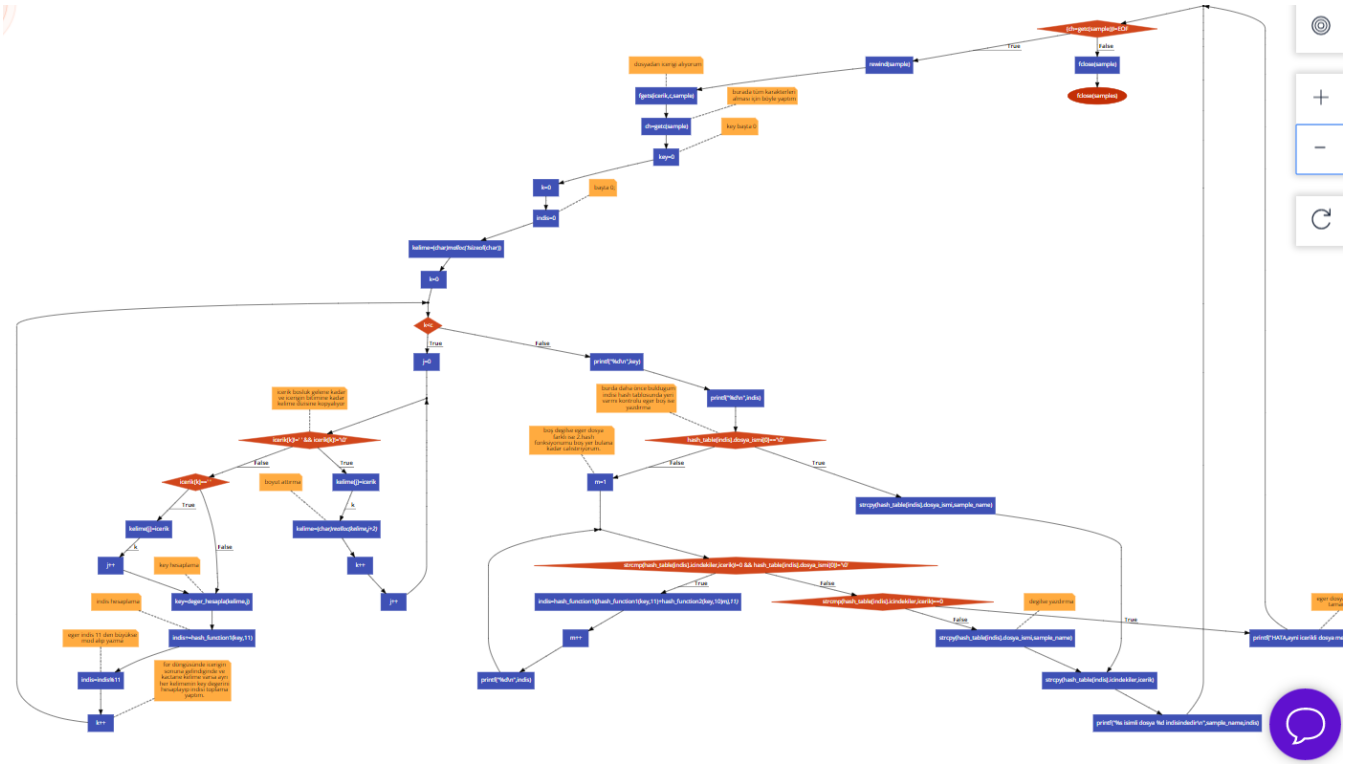
- DERSİN ADI: ALGORİTMA ANALİZİ
- AD-SOYAD: İLKER BEDİR
- NUMARA:16011036
- KONU: HASHİNG
- Tarih:27.11.2019

YÖNTEM BÖLÜMÜ:

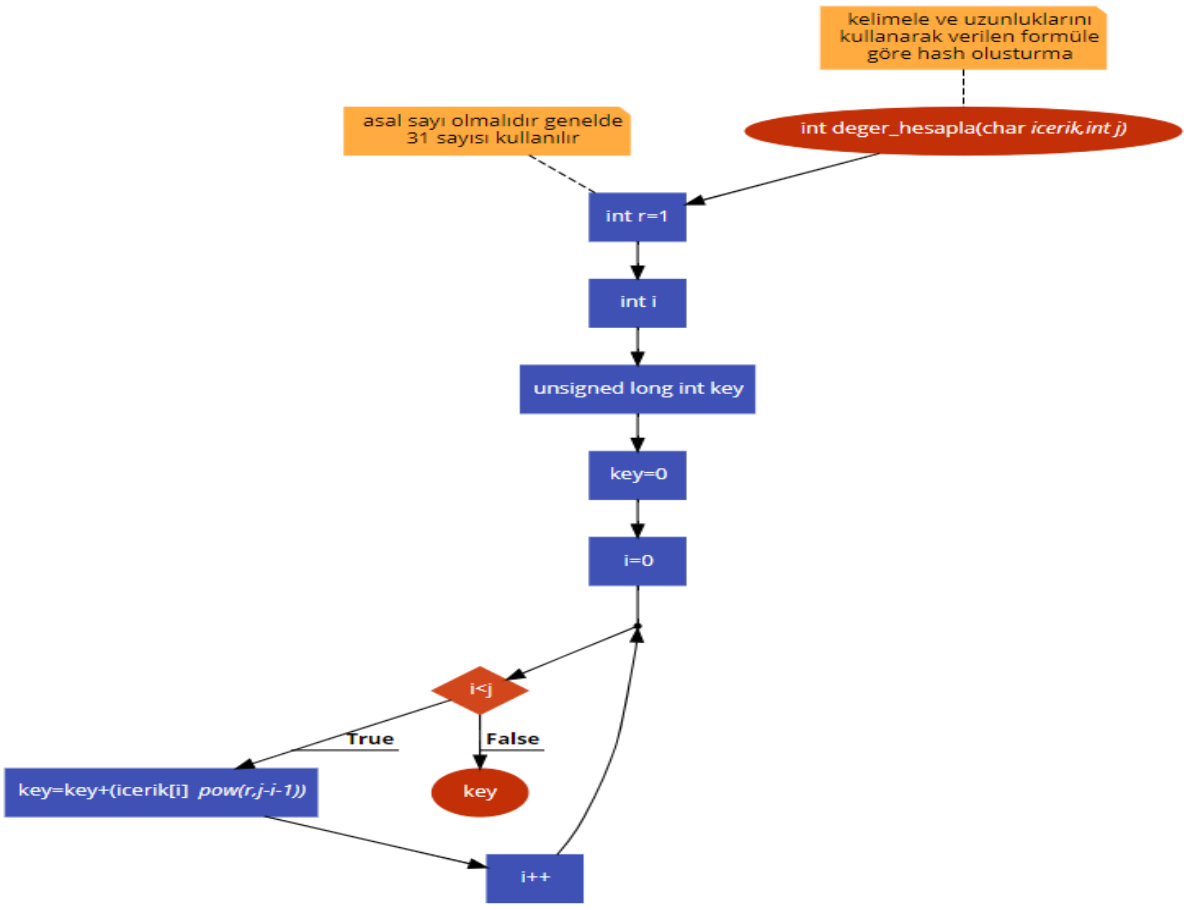
Problemimiz, eklenmek istenen yeni bir dosyanın,dosyaların isimlerinin bulunduğu dosyada olup olmadığını kontrol eden, eğer yoksa dosyaya ekleyen bir sistem yapmalıyız. Dokümanın dosyada olup olmadığının mevcut bütün dokümanların içeriklerine tek tek bakılarak yapılması zaman alıcı bir işlemdir. Bu yüzden hashing yapacağız.

Hash Tablosu Oluşturma Fonksiyonu:,

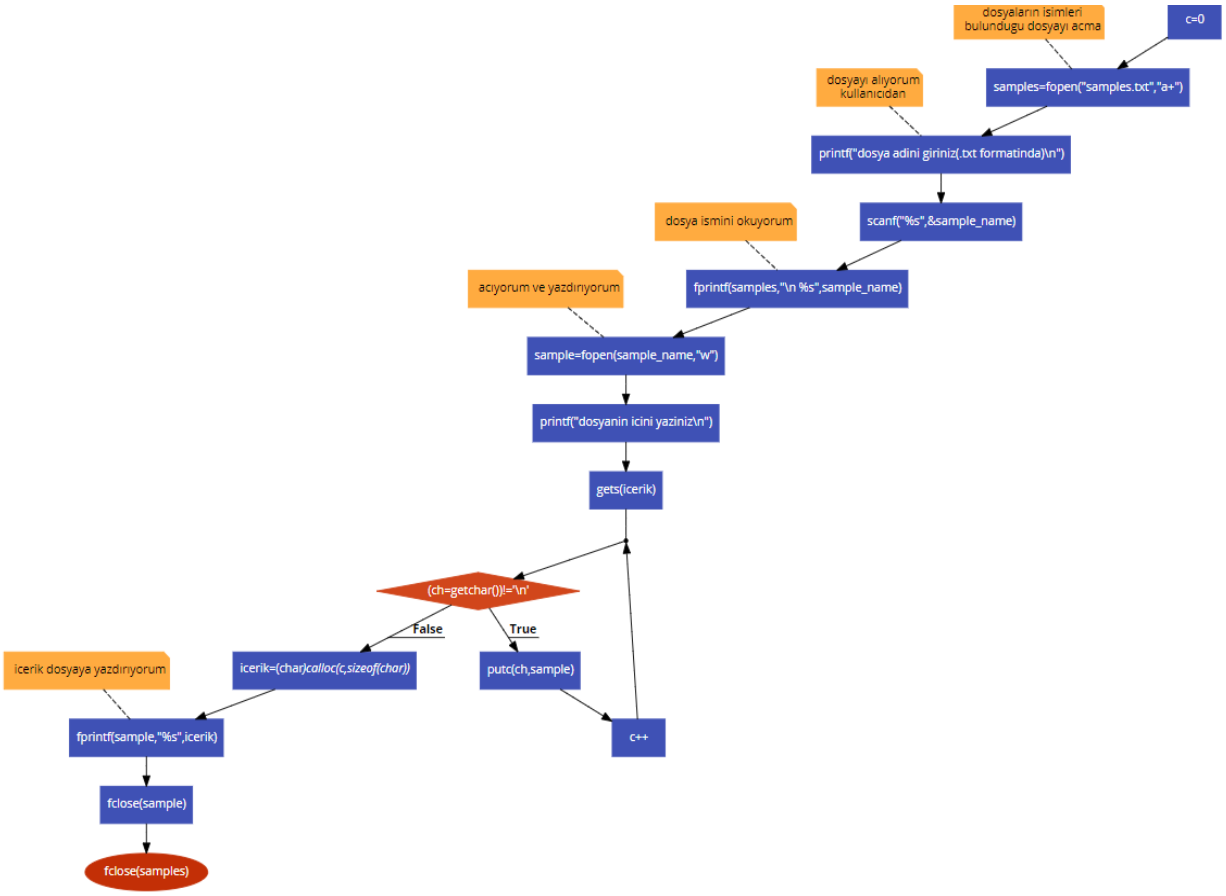




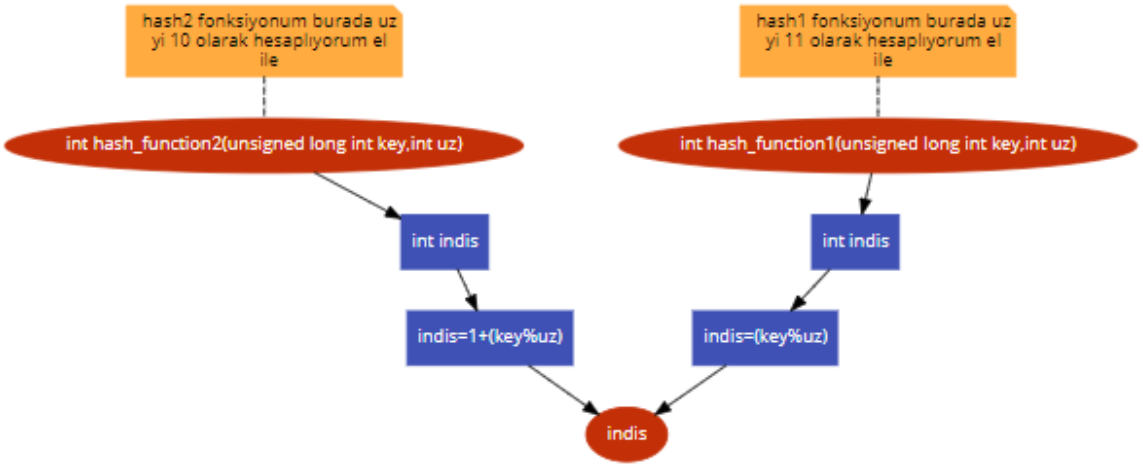
KEY Hesaplama:



Dosya Oluşturma Fonksiyonu:



Hash Fonksiyonlarım:



UYGULAMA BÖLÜMÜ:

Dosyalar;

Dosya1=abc

Dosya2=Sad

Dosya3=Ads

Dosya4 = Asd

Dosya5=aSD

KEY hesaplama; key = str[0] * R^n-1 + str[1] * R^n-2 + ... + str[n-1]

R=1 için;

Dosya1= a+b+c için key değeri 294; indis değeri 8'dir.Hash tablosunda 8. İndis boş olduğundan dolayı 8. İndise atacaktır.

Dosya2=S+a+d için key değeri 280;indis değeri 5'dir.Hash tablosunda 5.indis boş olduğundan dolayı 5.indise atacaktır.

Dosya3=A+d+s için key değeri 280;indis ilk değeri 5'dir ama indis dolu olduğundan dolayı diğer hash fonksiyonumuzu çağıracaktır ve yeni indisi 6 bulacaktır.6 da boş olduğu için yerleştirecektir.

Dosya4=A+s+d için key değeri 280 ; indisin ilk değeri 5 dir ama dolu olduğu için 2 kere diğer hash fonksiyonumuzu çağıracağız ve 7 indise yerleştireceğiz.

Dosya5=a+S+D için key değeri 248 ; indisin ilk değeri 6 'dır ama dolu olduğu için diğer hash fonksiyonunu çağırıyoruz. Bu seferde indis 4 buluyor ve yerleştiriyor.

SONUÇ BÖLÜMÜ:

Algoritmamız önce dosya adını fscanf fonksiyonu ile alıyor.N tane dosya varsa dosya okuma işleminin karmaşıklığı N'dir. Bu dosyaların içeriğini getchar ile okuduğum için dosyada M tane karakter var ise bu işlemin karşılığı N*M olacaktır.Her bir dosyanın her bir kelimesi ayrı ayrı key değerlerini hesapladığım için K kelime için K kadar key hesaplama yapacaktır.Eklemek için ama sonuç olarak hash fonksiyonumuz en iyi durumunda (yani ilk bulunan indis boş ise) sadece 1 adım yaparak yerleştirecektir (En iyi durum O(1)).Ancak en kötü durumda karmaşı biz ancak git gide ekleneceği için doluluk oranı artacaktır ve en sonunda yaklaşık değeri 1 yakın olacak (en kötü durum) bunun için karmaşıklığı ise O(N) olacaktır.

Tablo oluşturmak için N tane dosya ekleyeceksen her birinde N-1 tane boşluk kontrolü yaptığımız için şu karmaşıklık olur;

N için N-1 tane

N-1 için N-2 tane  $\left((n-1)*\frac{n-2}{2}\right)-(n-1)=\sum_{i=1}^{n-1} i-1$

…

…

2 için 1 tane

1 için 0 tane

Tablo oluşturmak için yukarıdaki denklem oluşacaktır.Bu denklemin karmaşıklığı O(N^2) (En kötü durum).En iyi durum ise O(N).

Genel olarak hashing işlemlerinde en kötü durum O(N^2) , En iyi ise O(N) karmaşıklarına sahiptir.