PROJE 1: DOKUWIKI KOCAELİ UNİVERSİTESİ

1st Muhammet Rıdvan İNCE
Kocaeli Üniversitesi
Bilgisayar Mühendisliği Bölümü
Kocaeli, Türkiye
Öğrenci No: 210201123

2nd Ömer YENER
Kocaeli Üniversitesi
Bilgisayar Mühendisliği Bölümü
Kocaeli, Türkiye
Öğrenci No: 210201122

***Abstract*—Dokuwiki Kocaeli Üniversitesi projesi kapsamında herhangi bir veri tabanı kullanılmadan üniversite bulunan bölümler ve bu bölümlerde okutulan derslerin içeriklerinin gösterildiği bir uygulama hazırlanması amaçlanmaktadır. Projenin temel amacı; ham verilerin bulunduğu dosyalardan proje kapsamında gerekli olan verilerin elde edilmesi (dosya okuma), yeni verilerin oluşturulması (dosyaya yazma) ve var olan verilerin değiştirilmesidir (güncelleme).**

***Index Terms*—C, Dosya Uygulamaları, Algoritma**

I. GİRİŞ

Dokuwiki Kocaeli Üniversitesi projesi C dilinde yazılmış olup 2 adet sabit değişken, 15 global değişkene ek olarak 7'si yardımcı fonksiyon, 1'i çekirdek fonksiyon ve 3'ü proje fonksiyonu olmak üzere 11 adet fonksiyon kullanılmıştır. Bunların haricinde **stdio.h**, **dirent.h**, **locale.h** ve **string.h** kütüphaneleri projeye dahil edilmiştir. Kütüphaneler içerisinde dirent.h; ilgili dizinlere ulaşılabilmesi için, locale.h; ekran çıktısında Türkçe karakterlerin gösterimleri için, string.h ise dosyalardan çekilen metinlerin manipüle edilebilmesi için kullanılmıştır. Raporun devam eden kısımlarında sırasıyla yöntem ve deneysel sonuçlardan bahsedilmiştir. Yöntem kısmında proje mimarisi tanıtılmış ve algoritma yapısı incelenmiştir. Deneysel sonuçlar kısmında ise uygulamanın sonuçları ve kısıtları hakkında bilgi verilmiştir.

II. YÖNTEM

Proje kapsamında; **UNIVERSITE_DIZIN** ve **UNIVERSITE_DIZIN2** isminde iki adet sabit değişken oluşturulmuştur. Bu sabitlerden ilki ÜNİVERSİTE klasörünün dizinini, ikincisi ise ÜNİVERSİTE klasörünün dizinin sonuna '/' ifadesinin yerleştirilmiş halidir. Tüm arama işlemleri bu iki değişken ile başlamakta ve alt klasörlere ve klasörlerin içerisindeki dosyalara ait dizinler dinamik bir şekilde belirlenmektedir. dirent.h kütüphanesine parametre olarak ilk gönderilen dizin UNIVERSITE_DIZIN değişkenidir. UNIVERSITE_ DIZIN2 değişkeni ise diğer dizinlere geçiş yapmak için kullanılmaktadır.

Diğer taraftan projede 15 adet global değişkene yer verilmiştir. Bunlardan 7 adeti integer tipinde 8 adeti ise iki boyutlu char tipindedir. Integer tipinde olan global değişkenler içerisinde **int uni_klasor_sayisi**; üniversite dosyası içerisindeki alt klasörlerin sayısını, **int uni_dizin_sayisi**; üniversite dosyası içerisindeki alt klasörlere ait dizinlerin sayısını , **int txt_dosya_sayisi**; üniversite klasörü içerisinde bulunan klasörlerin içerisindeki tüm txt dosyalarının sayısını, **int txt_dizin_sayisi**; üniversite klasörü içerisinde bulunan klasörlerin içerisindeki tüm txt dosyalarına ait dizinlerin sayısını , **int etiket_sayisi**; tüm txt dosyaları içerisinde etiket formatına uyan kelime / kelime öbeklerinin sayısını, **int yetim_etiket_sayisi**; tüm txt dosyaları içerisinde etiket formatına uyan ancak aynı isimde txt dosyası bulunmayan etiketlerin sayısını, **int istenen_etiket_sayisi**; tüm txt dosyaları içerisinde etiket isimleri ile uyuşmayan dosya isimlerinin sayısını göstermektedir.

Char tipinde olan global değişkenler içerisinde; **char uni_klasor[30][100]**; üniversite klasörü içerisindeki alt klasörlerin isimlerini, **char uni_dizin[30][200]**; üniversite klasörü içerisindeki alt klasörlerin dizinlerini, **char txt_dosya[100][100]**; tüm alt klasörlerdeki txt formatına uygun olan dosyaların isimlerini, **char txt_dizin[100][200]**; tüm alt klasörlerdeki txt dosyalarına ait dizinleri, **char txt_icerik[100][5000]**; tüm alt klasörlerdeki txt dosyalarının içeriklerini, **char etiket_listesi[100][100]**; tüm txt dosyalarındaki etiket formatına uyan kelime/kelime öbeklerini, **char yetim_etiketler[100][100]**; yetim etiketleri, **char istenen_etiketler[100][100]**; istenen etiketleri depolamaktadır. txt_dosya, txt_dizin ve txt_icerik stringlerinin içerisindeki değişkenlerin sıralaması aynıdır. Örneğin *txt_dosya[1]*, Matematik_I dosya ismi iken, *txt_dizin[1]* bu dosyaya ait dizini, *txt_icerik[1]* ise bu dosyaya ait içerikleri göstermektedir. Ayrıca Dosya isimlerden, ".txt" ifadesi *strcat()* fonksiyonu kullanılarak *VerileriGuncelle()* fonksiyonu içerisinde çıkarılmıştır.

Bu değişkenler; tüm fonksiyonların dışında tanımlanmış olup her birinin değerleri *VerileriGuncelle()* fonksiyonu içerisinde belirlenmekte ve istenildiği zaman tüm fonksiyonların içerisinden ulaşılabilmektedir. İçerisinde global değişkenlerinde bulunduğu proje mimarisi Şekil 1'de gösterilmektedir.

Mimariden de anlaşılabileceği üzere proje kapsamında hazırlanan fonksiyonlar Çekirdek fonksiyon, Yardımcı fonksiyonlar ve Proje Fonksiyonları olmak üzere 3 kategoriye ayrılmıştır.

Fonksiyonlar içerisinde çekirdek fonksiyon **(VeriGuncelleme())** tüm süreçlerin kontrol edilmesini sağlamakta olup uygulama çalıştırıldığında ilk çalışan fonksiyondur. Bu fonksiyonun görevi Yardımcı Fonksiyonların içerisindeki

Identify applicable funding agency here. If none, delete this.

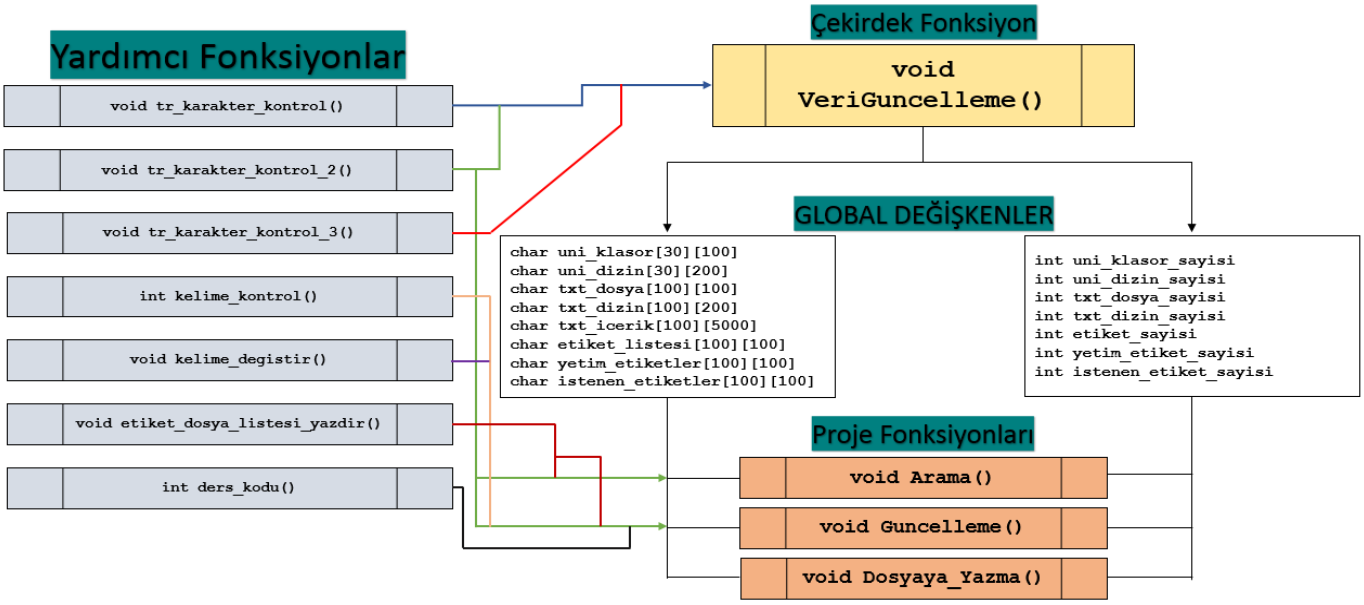

Fig. 1. Proje Mimarisi

Türkçe karakter kontrolü yapan fonksiyonları da kullanarak global değişkenlerin değerlerini belirlemektir. Proje kapsamında talep edilen yetim ve istenen etiketlerin tespit edilmesi de **(VeriGuncelleme())** fonksiyonu içerisinde yapılmaktadır. Dolayısıyla yetim ve istenen etiketlerin nasıl bulunduğuna yönelik oluşturulacak akış diyagramı da aslında **(VeriGuncelleme())** fonksiyonunun tamamına ait akış diyagramının çizilmesi gerektiği anlamına gelmektedir. Dolayısıyla talep edilen akış diyagramlarını gösteren ve rapor sonunda yer verilen **Şekil 3** tüm fonksiyonu kapsayacak şekilde hazırlanmıştır.

(VeriGuncelleme()) fonksiyonu projenin çekirdeğini oluşturmaktadır. Üniversite klasörü içerisindeki tüm veriler bu fonksiyon yardımıyla global değişkenlere aktarılmaktadır. Tüm GÖREV fonksiyonları, **(VeriGuncelleme())** fonksiyonu ile güncellenen global değişkenler üzerinden çalışmaktadır. Bu fonksiyon hem *main()* fonksiyonunun hemen başında, hem de *Guncelleme()* fonksiyonunun sonunda çalıştırılmaktadır.

Fonksiyonun hemen başında global değişkenlerin tamamı sıfırlanmıştır. Bunun yapılmasının sebebi özellikle *Guncelleme()* fonksiyonu çalıştırıldıktan sonra global değişkenler içerisinde bulunan eski verilerin üzerine yazılmasının engellenmesidir. Yani *VeriGuncelleme()* fonksiyonu her çağrıldığında global değişkenlerin hepsi sıfırlanmakta ve Üniversite klasörü içerisindeki güncel tüm veriler bu değişkenlere aktarılmaktadır. Fonksiyon içerisinde devam eden aşamada Üniversite klasörüne ulaşabilmek amacıyla *dirent.h* kütüphanesi kullanılmıştır. Bu kütüphane **DIR** ve **struct dirent** olmak üzere iki veri tipi içermektedir. **DIR** veri tipi dizin akışını, **struct dirent** veri tipi ise ilgili dizindeki klasör/leri temsil etmektedir.

Yardımcı fonksiyonlar içerisinde 3 adedi Türkçe karakter kontrolü (tr_karakter_kontrol(), tr_karakter_kontrol_2(), tr_karakter_kontrol_3()) için tanımlanmıştır. Bu fonksiyonların her biri, kendisine parametre olarak gönderilen string tipindeki değişkenin her bir karakterini kontrol ederek Türkçe karakter olanları İngilizce karaktere çevirmektedir.

Türkçe karakter kontrolü için 3 ayrı fonksiyon tanımlanmasının sebebi, Türkçe karakterlerin farklı durumlarda farklı değerler almasından kaynaklanmaktadır. Örneğin 'ç' karakteri:

- txt dosyasından veri okunduğunda aynı anda (-61 ve -89)
- gets() fonksiyonu ile kullanıcıdan veri alındığında -121
- script içerisinde string bir ifade tanımlandığında -25 değerini almaktadır.

Her üç durumunda ayrı ayrı kontrol edilmesi için 3 farklı fonksiyon tanımlanmıştır. **tr_karakter_kontrol()** fonksiyonu txt dosyalarından çekilen verilerin Türkçe karakter kontrolünü yapmaktadır. Bir adet string tipinde parametre almaktadır. Fonksiyona parametre olarak verilen stringin her bir karakterini dolaşarak içerisinde bulunan Türkçe karakterleri (ç, ı, ü, ğ, ö, ş, İ, Ğ, Ü, Ö, Ş, Ç), ingilizce karakterlere çevirmektedir. **tr_karakter_kontrol_2()** fonksiyonu gets() fonksiyonu ile kullanıcıdan alınan verilerin Türkçe karakter kontrolünü yapmaktadır. Bir adet string tipinde parametre almaktadır. Fonksiyona parametre olarak verilen stringin her bir karakterini dolaşarak içerisinde bulunan Türkçe karakterleri (ç, ı, ü, ğ, ö, ş, İ, Ğ, Ü, Ö, Ş, Ç), ingilizce karakterlere çevirmektedir. Bu fonksiyon, proje fonksiyonları içerisinde yer alan *Arama()* ve *Guncelleme()* fonksiyonları çağrıldığında kullanıcıdan istenen girişlerin kontrolünü sağlamaktadır. **tr_karakter_kontrol_3()** fonksiyonu ise script içerisinde kullanıcı tarafından tanımlanan stringlerin Türkçe karakter kontrolünü yapmaktadır. Bir adet string tipinde parametre almaktadır. Fonksiyona parametre olarak verilen stringin her bir karakterini dolaşarak içerisinde bulunan Türkçe

karakterleri (ç, ı, ü, ğ, ö, ş, İ, Ğ, Ü, Ö, Ş, Ç), ingilizce karakterlere (c, i, u, g, o, s, I, G, U, O, S, C) çevirmektedir. Bu fonksiyon txt_dosya değişkeni içerisindeki txt dosya isimlerindeki Türkçe karakterlerin ingilizce karakterlere çevrilmesi için kullanılmıştır.

Yukarıdaki üç fonksiyon yardımıyla txt dosyalarındaki karakterler, kullanıcıdan alınan veriler içerisindeki karakterler ve değişkenler içerisinde tutulan verilere ait karakterlerin tamamı ingilizce karakterlere çevrilmektedir. Dolayısıyla proje fonksiyonları içerinde bulunan *Guncelleme()* fonksiyonu ile herhangi bir kelime etiket ya da dosya güncellendiğinde güncellenen isim Türkçe karakter içermeyecektir.

Yardımcı fonksiyonlar içerisindeki diğer 4 fonksiyonların görevleri ise sırasıyla şu şekildedir.

- ***int kelime_kontrol()***: Bu yardımcı fonksiyon, proje fonksiyonları içerisindeki *Guncelleme()* fonksiyonu tarafından kullanılmaktadır. Bir adet char tipinde parametre almakta olup bu parametre Guncelleme() fonksiyonu içerisinde kullanıcıdan alınan güncellenmesi istenen değerdir. kelime_kontrol() fonksiyonunun görevi ise bu değerin bir kelimemi, Yetim etiket mi yoksa normal etiket mi olduğuna karar vermektedir.
- ***void kelime_degistir()***: Bu yardımcı fonksiyon da proje fonksiyonları içerisindeki *Guncelleme()* fonksiyonu tarafından kullanılmakta olup, üniversite klasörü içerisindeki tüm txt dosyalarını dolaşarak dosyalar içerisinde kullanıcı tarafından girilen değerleri tespit etmekte ve yine kullanıcı tarafından girilen değer ile değiştirmektedir.
- ***void etiket_dosya_listesi_yazdir()***: Bu yardımcı fonksiyon proje fonksiyonları içerisindeki hem *Guncelleme()* hem de *Arama()* fonksiyonu tarafından kullanılmaktadır. Bu fonksiyonun hemen başında çekirdek fonksiyonu çağrılmakta olup, fonksiyonun temel vazifesi çağırıldığı yerde en güncel yetim etiket, istenen etiket ve normal etiketleri (hem etiket olup hem de bu etiket adında dosya olması) yazdırmasıdır.
- ***int ders_kodu()***: Bu yardımcı fonksiyon da proje fonksiyonları içerisindeki *Guncelleme()* fonksiyonu tarafından kullanılmakta olup, herhangi bir yetim etiket değiştirilmek istendiğinde eğer bu etikete ait txt dosyası oluşturulması istenirse bu yeni dosyadaki ders kodunu belirlemektedir. Bu fonksiyona ait sözde kodlardan oluşturulmuş algoritma yapısı aşağıda gösterilmiştir.

1) Global değişkenlerdeki tüm verileri güncelle
2) İnteger tipinde ***enbuyuk*** isimli bir değişken oluştur ve değerini 0'a eşitle
3) txt_dizin global değişkeni içerisinde dizinleri verilen tüm txt dosyalarına ulaş.
4) Her bir dosyanın ilk satırını oku ve ***buf*** isimli değişkene ata (Çünkü genel şablonda ders kodları ilk satırda bulunuyor).
5) ***buf*** değişkeninin her bir karakteri içerisinde dolaş eğer rakam olan bir karaktere rastlar isen bu karakterin indeksi dahil ***buf*** değişkeninin sonuna kadar her bir karakteri ***kod*** isimli değişkene yazdır.
6) ***kod*** isimli değişkeni integera çevir.
7) ***kod*** isimli değişkeni ***enbuyuk*** ile karşılaştır ve eğer ***kod*** daha büyük ise ***enbuyuk***'ün değerini ***kod***'a eşitle.
8) Fonksiyonun değeri olarak ***enbuyuk***'ü döndür.

III. DENEYSEL SONUÇLAR

Uygulama hazırlanan bilgisayara bağımlı olmayıp her bilgisayarda çalıştırılabilmektedir. Mevcut kod akışı içerisinde düzenlenmesi gereken tek kısım sürece dahil edilecek olan klasörlerin dizinlerini gösteren 2 adet sabit değişkendir.

Proje kapsamında düzenlenmesi oluşturulması talep edilen 3 adet fonksiyon bulunmaktadır. Bu fonksiyonlar ve performansları ile ilgili bilgiler şu şekildedir.

Arama() Fonksiyonu: Bu fonksiyon içerinde 3 adet işlem yapılmaktadır. Bu işlemler sırasıyla: kullanıcıdan alınan değerin etiket olup olmadığının tespiti, yine kullanıcıdan alınan değerin hangi dosyada hangi satırlarda kaç defa geçtiği ve tüm yetim ve istenen etiketlerin listelenmesidir. *Arama ()* fonksiyonu kapsamında tek bir karakteri kelime ya da kelime öbeği için işlem yapılabilmektedir. Bu fonksiyon kullanılırken dikkat edilmesi gereken husus ise aranan değerin dosyalardaki hangi satırlarda bulunduğu bilgisidir. Bazı dosya içeriklerinde iki satır arasında boşluk görülmekle birlikte bu boşlukta – boşluk karakteri dahil – herhangi bir karakter bulunmadığından bu kısım uygulama tarafından satır olarak değerlendirilmemektedir. Şekil 2'de Dersin kodu, ve Dersin adı ile başlayan satırların alt satırlarında gösterilmiştir. Diğer taraftan boşluk karakteri bulunan satırlarda herhangi başka bir karakter bulunmasa dahi satır olarak kabul edilmektedir. Şekil 2'de Öğretim Elemanları ile başlayan satırın alt satırında gösterilmiştir. Bu sebeple aranan kelimenin hangi satırlarda bulunduğu bilgisi bu kapsamda değerlendirilmelidir.

Guncelleme() Fonksiyonu: Güncelleme fonksiyonu kapsamında ... adet işlem yapılabilmektedir. Bunlar; kelime değişikliği, yetim etiket değişikliği ve normal etiket değişikliğidir. Kullanıcıdan alınan ve yetim ya da normal etiket (normal etiket bir etikete ait dosyanın bulunması durumudur) olmayan herhangi bir değer tüm txt'lerde tespit edilerek yine kullanıcıdan alınan başka bir değer ile değiştirilebilmektedir. Ayrıca bu değer eğer istenen etiketler listesinde var ise yani bu değere ait bir dosya bulunuyor ise bu dosyanın adı da değiştirilmektedir. İkinci işlem ise yetim etiket değişikliğidir. Kullanıcıdan alınan değer eğer yetim etiket ise tüm dosyalardaki tüm değerler yeni değer ile değiştirilmekte ve kullanıcıya bu yeni değerin dosyasının oluşturulup oluşturulmaması yönünde soru sorulmaktadır. Eğer dosya oluşturulur ise bu etiket normal etiket statüsü kazanmaktadır. Üçüncü işlem ise Normal etiketlerin isminin değiştirilmesidir. Bu işlem neticesinde txt dosyalarındaki tüm değerler değiştirilmekte ve ilgili dosya adı da güncellenmektedir.

Dosyaya_yazma() Tüm etiketler ve bu etiketlerin tüm dosyalarda toplam kaç defa geçtiği listelenmekte ek olarak bu etiketler içerisinde hangilerinin yetim etiket olduğu bilgisi verilmektedir.

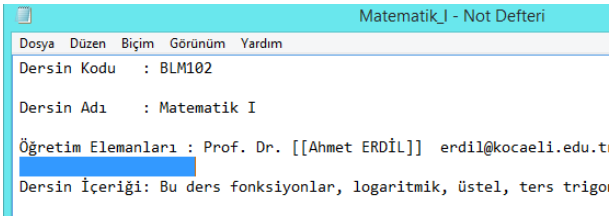

Fig. 2. Dosya İçeriği

Uygulama oluşturulduktan sonra test aşamasına tabi tutulmuş ve aşağıdaki kısıtların varlığına dikkat edilmiştir. Uygulama kullanılırken bu **kısıtlara** dikkat edilmesi gerekmektedir.

- Üniversite klasörü içerisindeki alt klasörlerin '.' sembolü taşımaması gerekmektedir. Eğer bu şekilde bir alt klasör olur ise bunun kontrolü sağlanmayacaktır. Çünkü dosya ve klasör ayrımı bu şekilde tasarlanmıştır.
- Taranacak ana klasör alt klasörleri alt klasörler ise taranacak txt dosyalarını içermelidir. Dosyalar arasında farklı bir katman bulunmamalıdır. Yani Üniversite klasörünün alt klasörlerinde farklı bir alt klasör oluşturup ilgili txt dosyalarının bu klasöre yerleştirilmesi durumunda uygulama doğru bir şekilde çalışmayacaktır.
- Alt klasörlerde bulunan txt dosyalarının ya UTF – 8 ya da ANSI kodlama türünde olması gerekmektedir. Çünkü tr_karakter_kontrol() fonksiyonu sadece bu tip dosyalardaki Türkçe karakterleri ingilizce karakterlere çevirecek şekilde tasarlanmıştır.

SONUÇ

Bu projede herahngi bir veritabanına ihtiyaç duyulmadan üniversite ve bu üniversite de bulunan bölüm ve derslere ait bilgilerin txt dosyalarından okunması, bilgilerin ve dosya isimlerinin değiştirilmesi ve mevcut bilgilerin güncellenmesine yönelik bir uygulama hazırlanmıştır. Uygulama C dilinde yazılmış olup, standart kütüphaneye ek olarak 3 adet kütüphane kullanılmıştır. Proje mimarisi kapsamında bir çekirdek fonksiyon hazırlanmış bu fonksiyon proje kapsamında talep edilen bilgilerin depolandığı global değişkenlere bağlanmıştır. Proje kapsamında talep edilen tüm işlemler ise bu global değişkenler üzerinden gerçekleştirilmiştir.

Projenin hazırlanması aşamasında en fazla zorluğun karşılaşıldığı durum txt dosyalarından Türkçe karakterlerin okunması ve okunan bu değerlerin kullanıcıdan alınan değerlerle karşılaştırılması süreçlerinde yaşanmıştır. Bu zorluğun üstesinden gelinebilmesi amacıyla 3 farklı fonksiyon projeye dahil edilmiştir. Bu fonksiyonlar ise UTF-8 ve ANSI kodlaması ile oluşturulan txt dosyalarındaki Türkçe karakterleri İngilizce karakterlere çevirmektedir. Dolayısıyla uygulama çalıştırılmadan önce verilerin çekileceği txt dosyalarının bu kodlama türleri ile oluşturulduğundan emin olunmalıdır.

Projenin tamamlanması yaklaşık olarak 3 hafta sürmüştür. Bu süreçte ilk hafta proje kapsamında talep edilenlerin tam olarak anlaşılması ve bu kapsamda çeşitli araştırılmaların yapılmasında ayrılmıştır. İkinci haftada proje mimarisinin oluşturulması ve bu mimariye uygun olacak şekilde kodların hazırlanmasına ayrılmıştır. Son hafta da ise hazırlanan kodların testi yapılmış ve talep edilen akış diyagramları hazırlanmıştır.

REFERENCES

[1] T. Karaçay, "C Programlamanın Temelleri," Abaküs Kitap Yayın Eğitim Ltd. Şti., Ağustos 2015
[2] stackoverflow.com
[3] geeksforgeeks.org
[4] Kocaeli Üniversitesi Bilgisayar Mühendisliği Bilgisayar Labaratuvarı Ders Notları

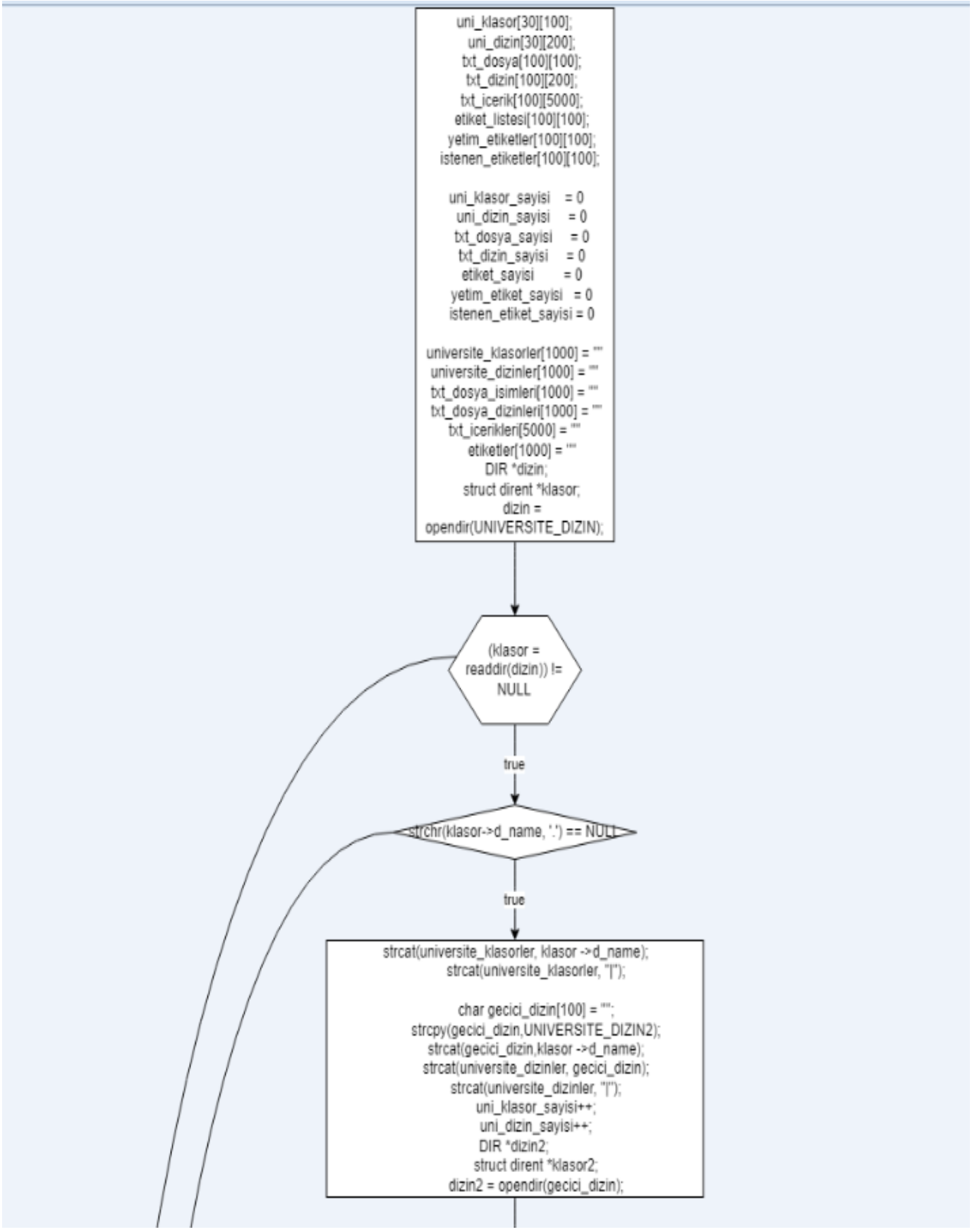

Fig. 3. Akış Diyagramı - 1

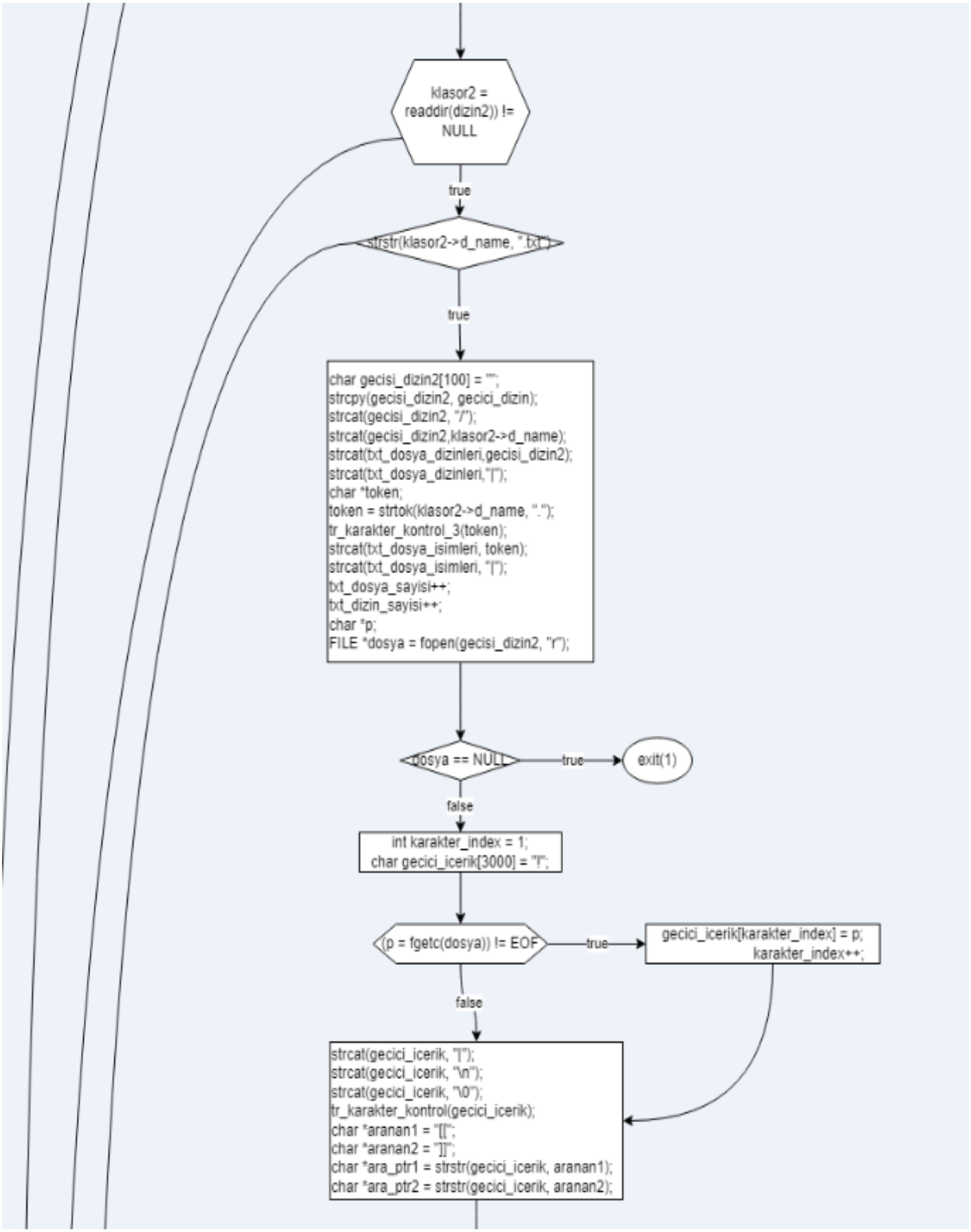

Fig. 4. Akış Diyagramı - 2

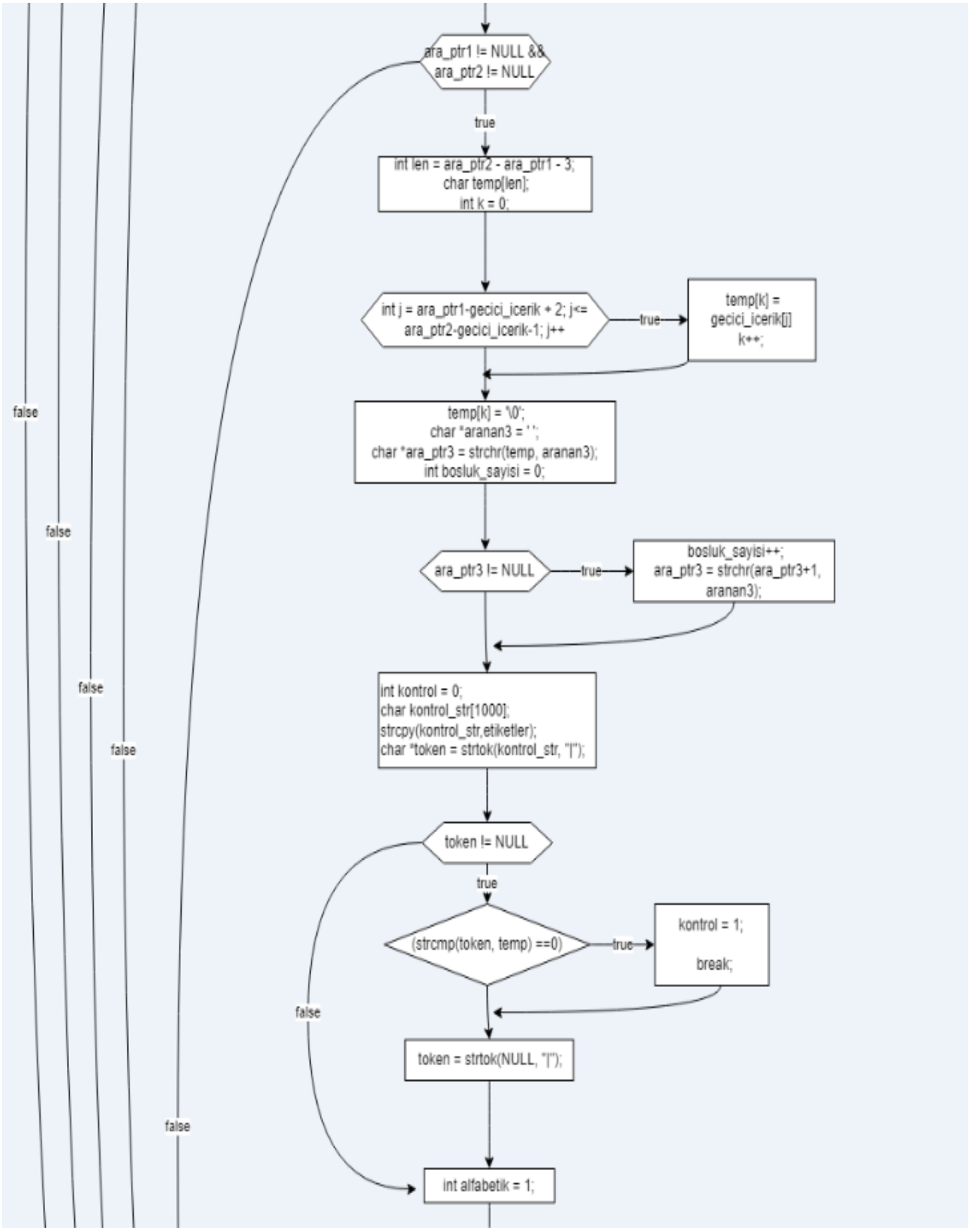

Fig. 5. Akış Diyagramı - 3

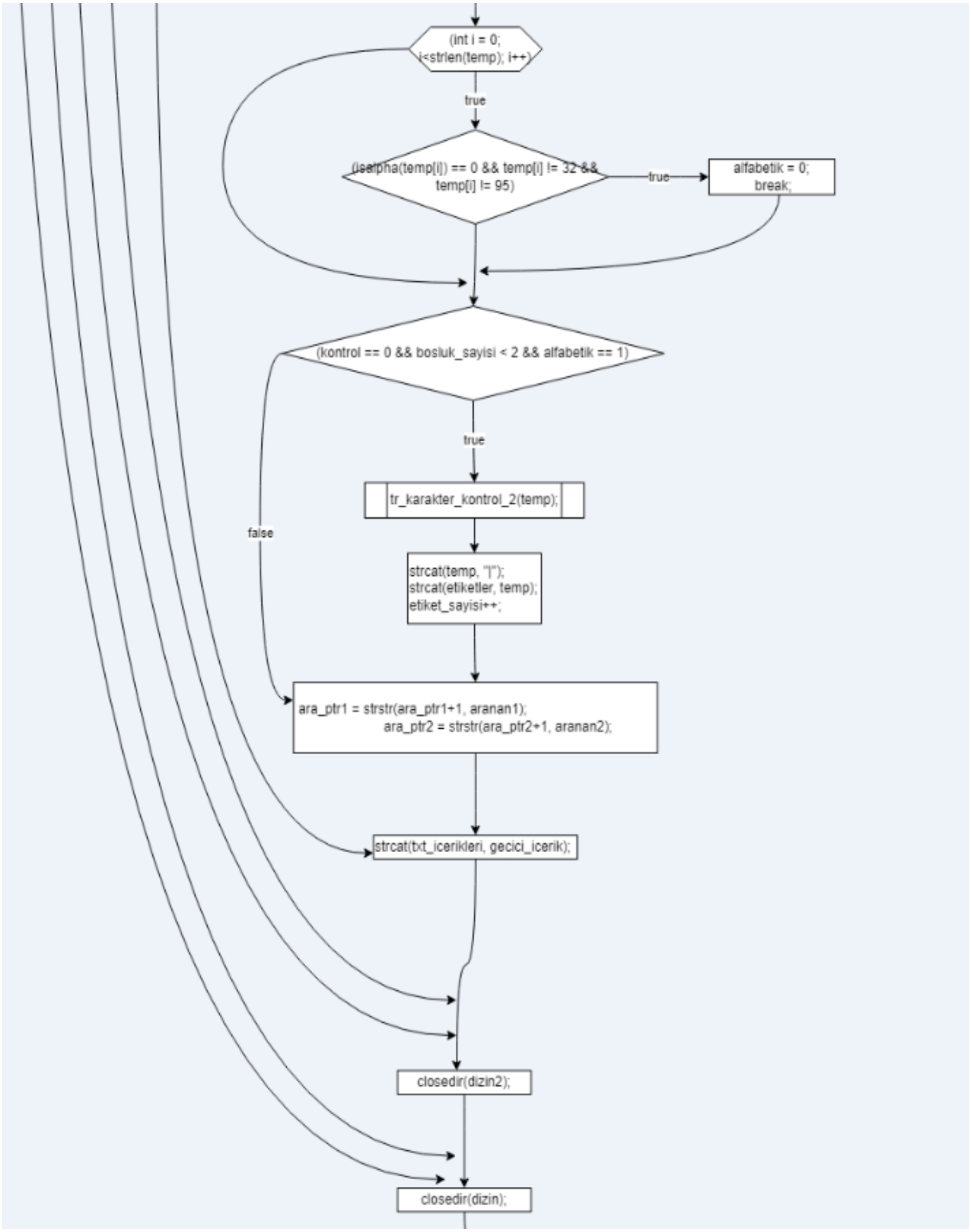

Fig. 6. Akış Diyagramı - 4

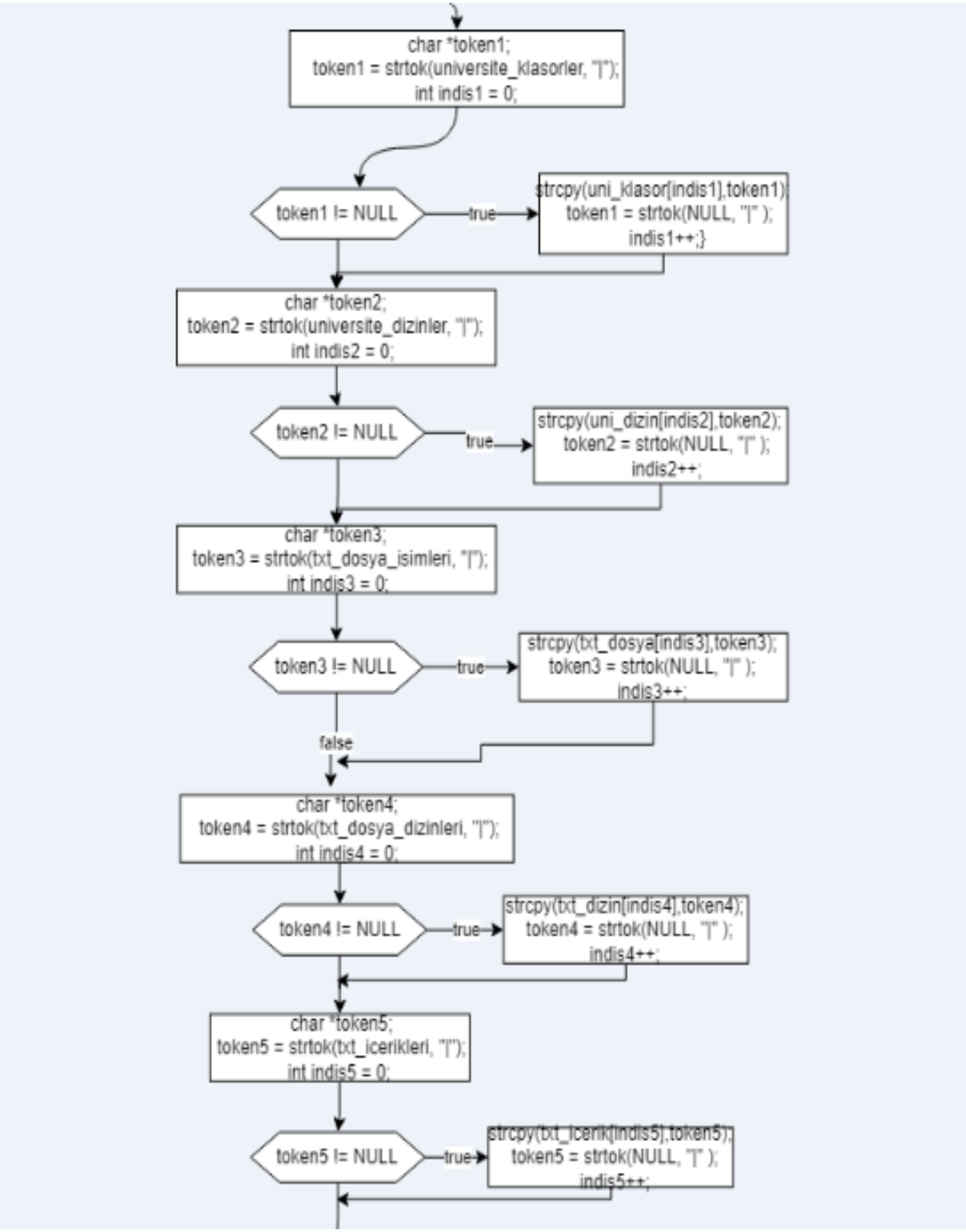

Fig. 7. Akış Diyagramı - 5

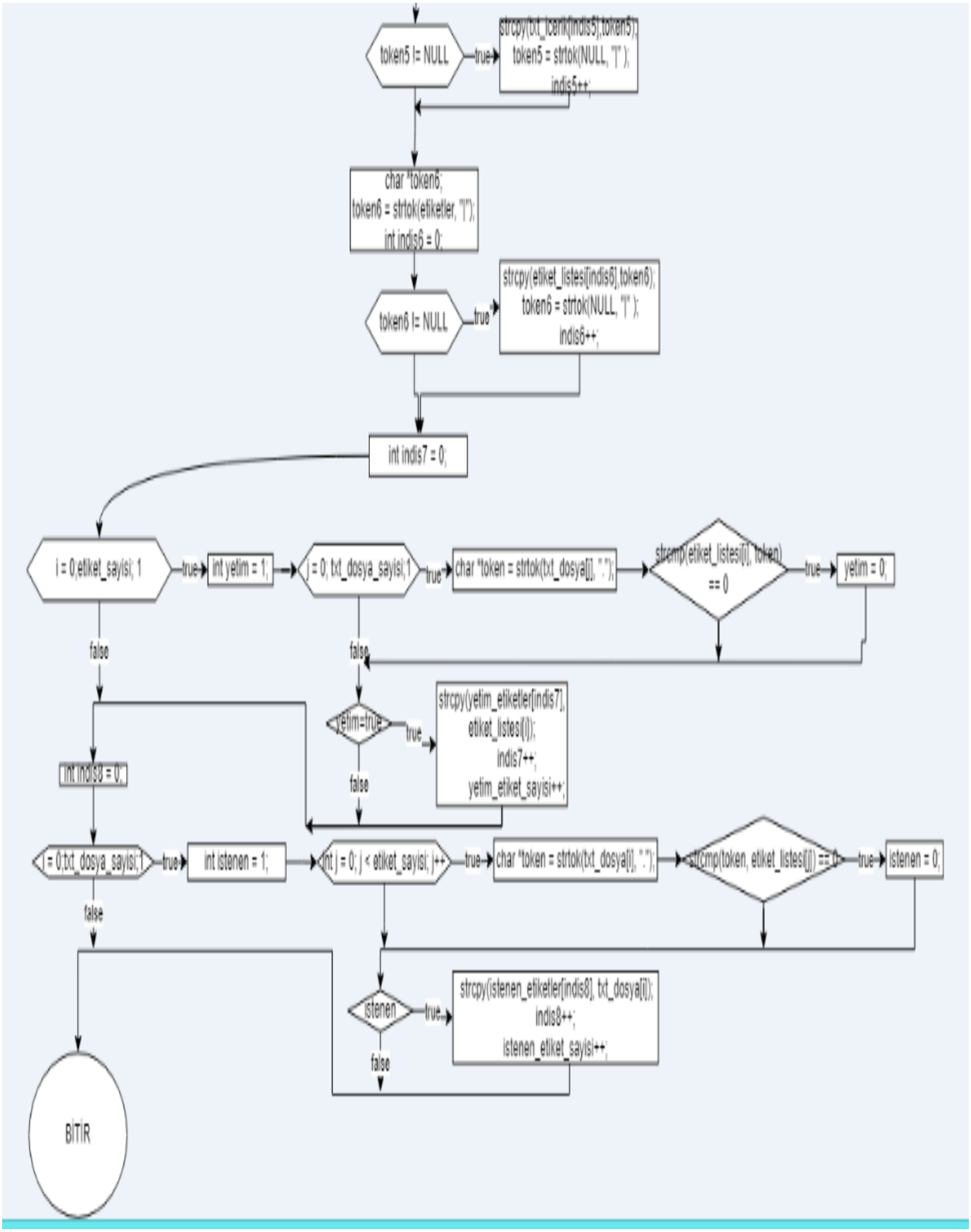

Fig. 8. Akış Diyagramı - 6