Öğr.Gör.Dr. Ahmet ELBİR BLM2031 Yapısal Programlama

**Son Teslim tarihi:** **02.01.2023 23:59**

Yapısal Programlama Projesi

Oğuzhan Topaloğlu

Ç19052025 – Grup 2

*Bilgisayar Mühendisliği Bölümü,*

*Elektrik-Elektronik Fakültesi,*

*Yıldız Teknik Üniversitesi*

****

Istanbul, 2023

|  |
| --- |
| **İÇERİK** |
| 1. Genel Bilgiler |
| 2. Menü Fonksiyonlarının Çalıştırılması |
| 3. Projenin Kazandırdıkları |
| 4. Kaynaklar |
| 5. Veritabanının Son Durumu |
|  |
|  |
|  |
|  |

**1. Genel Bilgiler**

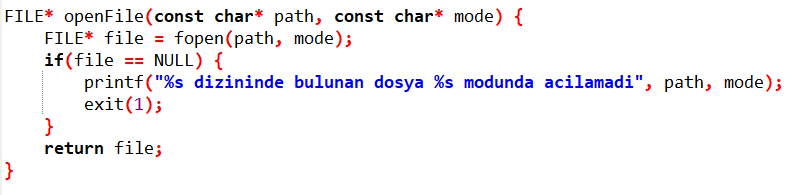
Bu projede küçük çaplı bir kütüphane otomasyonu kodlanmıştır. C dosyası sırasıyla (yukarıdan aşağıya doğru) şu bölümlere ayrılmıştır:

1. Include’lar ve makro tanımları
2. Struct tanımları
3. Fonksiyon prototipleri
4. Main fonksiyonu
5. Menu fonksiyonlarının tanımları
6. Printleme fonksiyonları
7. Dosya fonksiyonları
8. Helper fonksiyonlar

Birinci bölümde gerekli kütüphaneler eklenmiş ve kodun bir sürü yerinde kullanılacak makrolar tanımlanmıştır. Bu makrolar bir kitap örneğinin ödünç alınmamış olduğunu belirten “Rafta” const char pointer’ı ve 1024’e eşit olan MAX\_LEN sayısıdır. MAX\_LEN sayısı kod içinde dosya okurken ve string’lerle çalışılırken kullanılacaktır.

İkinci bölümde proje pdf’inde verilen struct’ların tanımları bulunmaktadır. Bu struct’larda gerekli değişiklikler soru drive’ında verilen cevaplara göre yapılmıştır. Örneğin tanımlarda typedef kullanılmıştır, struct member’larının isimleri yazılırken camelCase’e uyulmuştur ve ogrID gibi char[] değişkenlerin en sonunda ‘\0’ bulunması gerektiğinden boyut 1 arttırılmıştır.

Üçüncü bölümde kod içinde kullanılan main() dışında her fonksiyonun tanımı bulunmaktadır. Bu fonksiyonlar; menü fonksiyonları, printing/string fonksiyonları, dosya fonksiyonları ve helper (yardımcı) fonksiyonlar olarak 4’e ayrılmışlardır. Menü fonksiyonları, main fonksiyonu içinde yazdırılan menüde direkt olarak çağrılan fonksiyonlardır. Printing/string fonksiyonları isimlerinden de anlaşılacağı gibi struct nesnelerini string’e dönüştüren veya yazdıran fonksiyonlardır. Örneğin studentToString(Ogrenci\*) fonksiyonu içine verilen öğrenci parametresini bir char\* olarak geri döner. Bu string fonksiyonları malloc ile üretilen yeni bir string döndüğünden bu string’ler sonradan free edilmelidir. Dosya fonksiyonları, dosyaları okuyan ve main içinde tanımlı linkedlist ve dizilere yazan fonksiyonlardır. Bu fonksiyonlar ayrıca bu veri yapılarındaki duruma göre istenilen bir dosyayı silip yeniden de yazabilmektedir. Örneğin writeBooks(Kitap\*\*, const char\*) fonksiyonu içinde verilen path’e kitap single linkedlist’ini yazdırır. Helper fonksiyonlar da kod içinde öbür fonksiyonlar içinde kolaylık sağlamaları amacıyla tanımlanmış fonksiyonlardır. Örneğin openFile(const char\*, const char\*) fonksiyonu bir FILE\* oluşturur ve bu null ise kontrol yapar ve ardından bu FILE\* nesnesini döner. Bu fonksiyon sayesinde kodun bir sürü yerinde her dosya açtığımızda NULL kontrolü yapmaktan kaçınırız. (Bu FILE\* sonrasında fclose ile kapatılmalıdır)



Dördüncü bölümde main fonksiyonu bulunmaktadır. Main içinde öncelikle kod içinde kullanılacak dosyaların path’leri tanımlanmıştır. Ardından da readStudents(const char\*), readAuthors(const char\*) ve readBooks(const char\*) fonksiyonları ile çift linkli ogrenciler listesi, tek linkli yazarlar ve kitaplar listeleri okunmuştur. Bunlar okunduktan sonra da geç teslim edilen kitapların ISBN bilgilerini depolamak için bir char\* dizisi tanımlanmıştır. Ardından da kitapYazar ve oduncler dinamik dizileri readBookAuthor(Kitap\*\*, Yazar\*\*, const char\*, int\*) ve readBorrowed(const char\*, int\*) fonksiyonaları ile okunmuştur. Burada kod çalışmadan önce veritabanında KitapOdunc.csv dosyası olabilir diye düşünüldüğünden bu okumalar ardından processBorrowed(...) fonksiyonu ile okunan veriler öğrencilere, kitaplara ve geç teslim edilen kitap ISBN’leri dizisine geçirilmiştir.



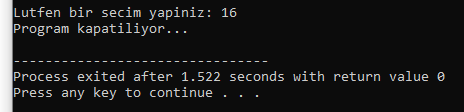
Bu fonksiyonlardan sonra uzun bir menü gerektiğinde alt menüleri ile print edilmiş ve girdiler istenerek fonksiyonlar çalıştırılmıştır. Main fonksiyonun en sonunda da ilk başta tanımladığımız linkli listeleri ve dinamik dizileri free etmekteyiz.

Beşinci, altıncı, yedinci ve sekizinci bölümlerde de üçüncü bölümün açıklamasında anlatılan fonksiyonlar kategorize edilerek tanımlanmıştır.

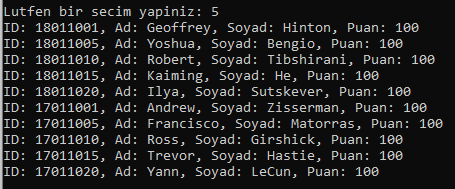
**2. Menü Fonksiyonlarının Çalıştırılması**

Bu bölümde her menü fonksiyonun çalıştırımasını göstereceğim. Menü fonksiyonları 1’den 16’ya doğru sırasıyla değil, kolaydan zora doğru gösterilecektir.

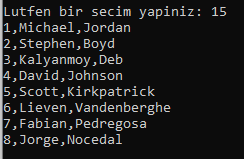
**Seçenek 16:** Bu seçenek seçildiğinde uygulama running = 0 yapılarak döngüden çıkılır ve önceden oluşturulan linkli listeler ve diziler free edilerek program kapatılır.



**Seçenek 5:** Bu seçenek seçildiğinde Ogrenciler.csv dosyasında yazan ve main’de ilk başta okunan değerler konsola print edilir.



**Seçenek 15:** Bu seçenek seçildiğinde Yazar.csv dosyasından okunan veritabanında şu anda bulunan yazarlar konsola yazdırılır. Burada yazarlar ID’lerine göre küçükten büyüğe göre sıralıdır.

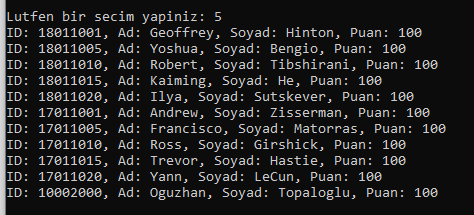
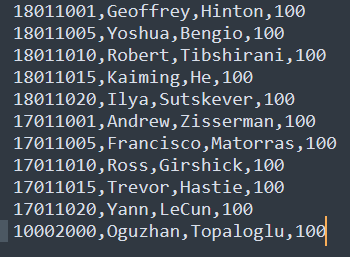
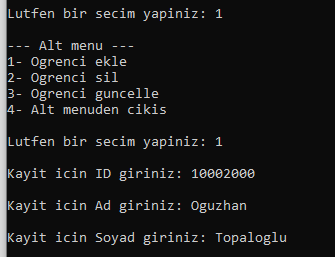


**Seçenek 1:** Bu seçenek seçildiğinde bir altmenü yazdırılır ve alt menü seçeneğine göre öğrencil eklenir/silinir/güncellenir. Bu değişimler hem çift linkli öğrenciler listesinde hem de dosyada gözükmektedir.

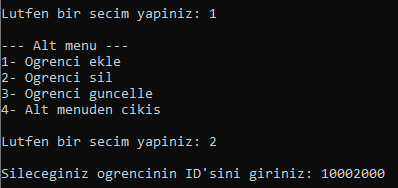
Herhangi bir işlem yapılmadan önce Ogrenciler.csv şu şekildedir:



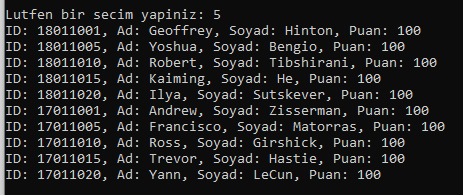
Eğer aşağıdaki gibi bir ekleme yaparsak bu ekleme hem listeye hem de dosyaya etki eder:



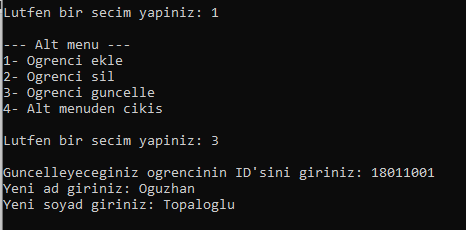
İstersek bu yeni eklediğimiz kaydı ya da başka bir kaydı silebiliriz de:

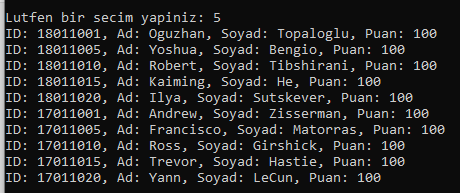






Benzer şekilde güncelleme de yapılabilir:

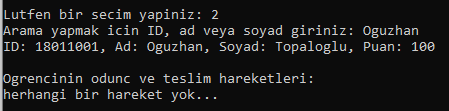






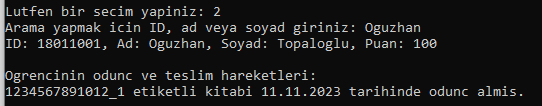
Bu işlemlerde önceden alınmış ID’lerin alınmasına izin verilmemesi gibi veri tutarlılığı için kontroller yapılmaktadır. Rapor çok uzun olmasın diye her birini teker teker göstermiyorum.

**Seçenek 2:** Bu seçenekte girilen bir ID/isim/soyisim string’ine göre bir öğrenci araması yapılır ve eğer bu öğrenci var öğrencinin bilgileri ve odunc/teslim hareketlerinin tümü yazdırılır.



Eğer KitapOdunc.csv dosyasında şöyle bir girdi olsaydı bu otomatik olarak algılanır ve gösterilirdi:



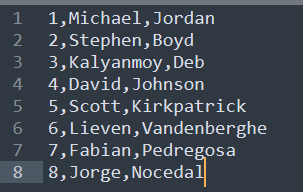


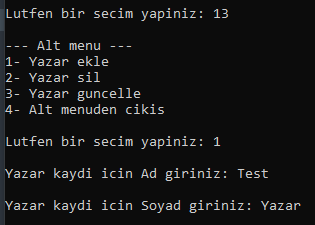
Vâr olmayan girdilerde de hiçbir işlem yapılmaz:

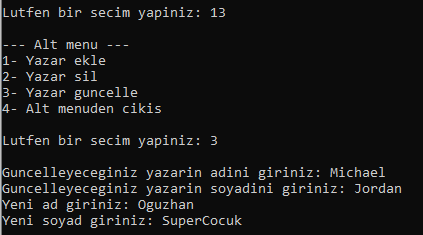


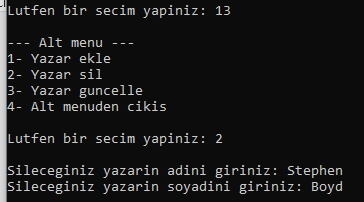
**Seçenek 13:** Bu seçenek, 1. seçenek gibi yazar eklemeye/silmeye/güncellemeye yarar. Bu işlemler sonrasında liste içinde yazarID’ye göre sıralama korunur ve dosyaya da bu işlemler yansıtılır. Ekleme yapılırken en yüksek ID’li yazarın ID’si bulunur ve bu ID’ye 1 eklenerek yeni bir ID elde edilir. Bu yüzden sıralama algoritmasına gerek olmamaktadır çünkü eklemeler zaten her zaman en yüksek ID’de bir şey ekleyeceğinden en sona eklenir. Güncellemelerde de ID güncellenmesi anlamsız olacağından böyle bir şey yapılmamaktadır. Silme sırasında da aradan bir değer çıkacağından yine sıra bozulmaz. Aşağıdaki her bir işlem için küçük bir örnek bulunmaktadır. Burada veri tutarlılığı için kontroller de vardır (lütfen bu kontroller için koda bakınız)

NOT: Burada yazar silinirken eğer bu yazarın yazdığı kitap varsa KitapYazar.csv dosyasında bu yazarın ID’sinin bulunduğu yerlere -1 yazılmaktadır. Örneği çok yer kaplayacağından koymuyorum lütfen kodu çalıştırarak kendiniz görünüz.

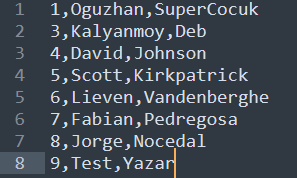




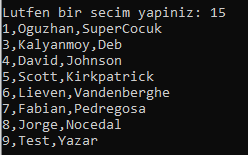




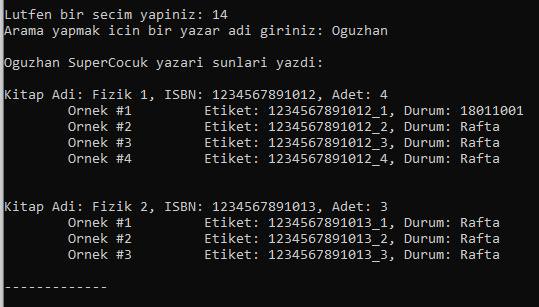
Bu işlemler sonrasında dosyanın hâli:



Aynı durum linkli listede de geçerlidir:

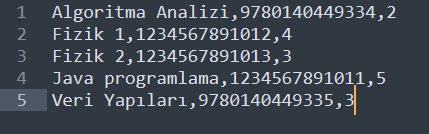


**Seçenek 14:** Bu seçenek seçildiğinde bir yazar araması yapılması için bir ad girdisi istenir. Bu girdiye göre yazar bulunursa bu yazarın yazdığı her kitabın ve bu kitapların her örneğinin bilgileri yazdırılır.

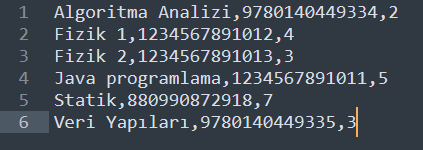


**Seçenek 7:** Bu seçenek, 1 ve 13. seçeneklere benzemektedir. Amacı kitap ekleme/silme/güncellemedir. Yine veri tutarlılığı göz önünde bulundurulmuştur. Lütfen kodu inceleyin bu tutarlılık kontrolleri için. Aşağıda bir örnek bulunmaktadır:

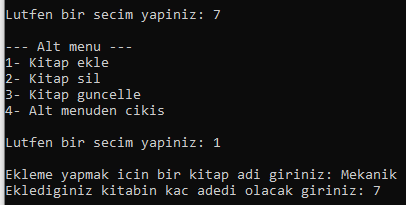
Önce:

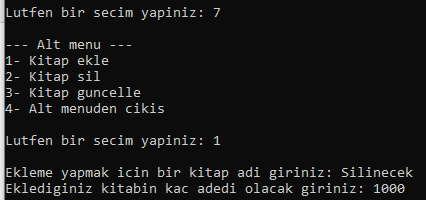


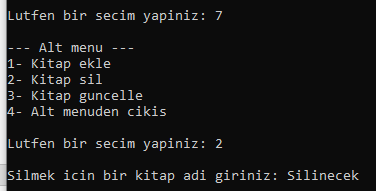
Sonra:

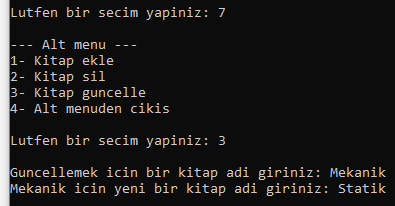


İşlemler:



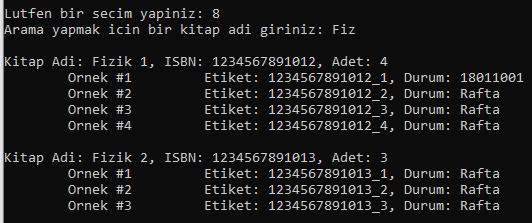




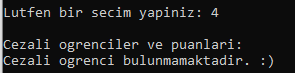


NOT: burada fark edebileceğiniz gibi program açıldığında dosya okunurken de, bir kitap eklemesi yapıldığında da kitaplar her zaman önce kitap adına göre sonra da ISBN numarasına göre sıralı olmaktadır. Bu sıralı durum sağlandıktan sonra dosya da güncellenmektedir. Yani unsorted bir Kitaplar.csv ile program açılırsa program bu dosyayı otomatik olarak sıralar.

**Seçenek 8:** Bu seçenek ile herhangi bir kitap için arama yapılabilmektedir. Kodlanırken strcmp yerine strstr fonksiyonu kullanıldığından dolayı substring araması yapılmaktadır ve her kitabı bulabilmektedir. Aşağıda bir örnek bulunmaktadır:



**Seçenek 4:** Bu seçenek cezalı öğrencileri konsola yazdırmaya yaramaktadır. Cezalı öğrenciler 100 puandan az puana sahip olacağından arkaplanda çalışan fonksiyon yalnızca “student->puan < 100” diye bir kontrol yapmaktadır.



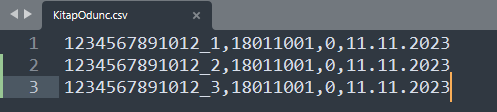
Eğer KitapOdunc.csv’ye şunlar manüel olarak eklenirse:

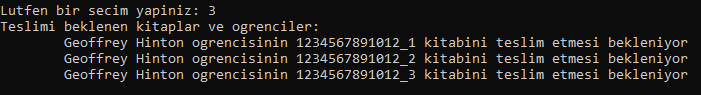


Bu kez şöyle bir sonuç çıkar:



**Seçenek 3:** Bu seçenek, geçmişte ödünç alınmış kitapları ve teslim edecek kişileri yazar.

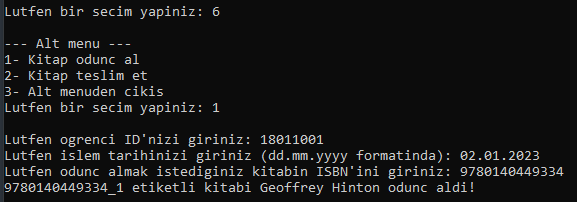


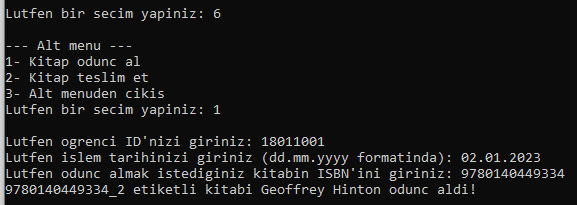


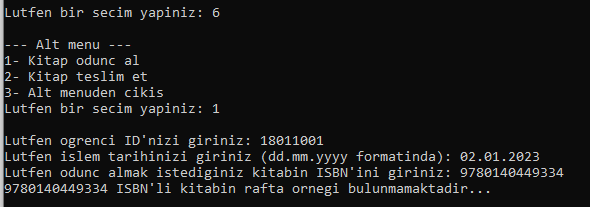
**Seçenek 6:** Bu seçenek, herhangi bir öğrencinin herhangi bir geçerli tarihte önceden ödünç aldığı bir kitabın örneğini teslim etmesini veya örneği rafta olan bir kitabın herhangi bir örneğini ödünç almasını sağlar. Bu fonksiyon ardından değişiklikler hem programda hem de dosyada uygulanmakta ve fonksiyon sırasında bir sürü kontrol yapılmaktadır.

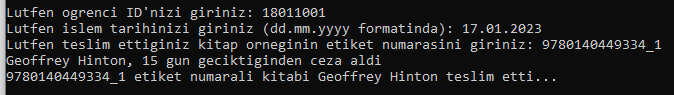
Bu kontroller teslim sırasında ISBN ve geçerli adet (diyelim ki 1234\_20 isteniyor eğer 1234 ISBN’li kitabın 20’den az örneği varsa böyle bir örnek yoktur) kontrol edilir, kitabı o kişinin cidden geçmişte ödünç aldığı (ödünç alınan kitabı yalnızca ödünç alan teslim edebilir) kontrol edilir, teslim eden kişinin öğrenci kaydı olması gereklidir, teslim edilen kitap rafta olmamalıdır.

Örnekler (KitapOdunc.csv boş olarak başlayalım):

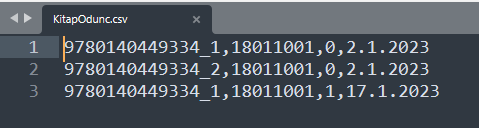




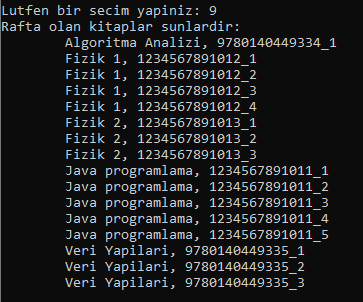




Sonuç:



**Seçenek 9:** Bu seçenek durumu “Rafta” olan her kitap örneğini yazdırır. (Örnekte Alg. Analizinin 2. Örneği yok çünkü bir önceki örnekte 18011001 numaralı öğrenci bunu ödünç aldı)

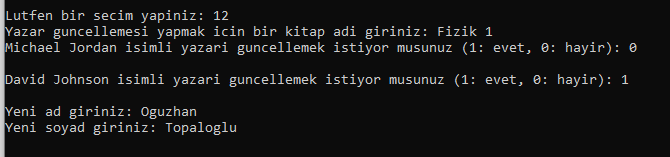


**Seçenek 10:** Bu seçenek, geç teslim edilen ve birisine ceza verilmesine neden olan her teslim işlemindeki kitabın bilgilerini yazar. Bir önceki örnekte Alg. Analizinin 1. örneği geç teslim edildiğinden çıktı şöyledir:



**Seçenek 12:** Bu seçenek, girilen bir kitap adı girdisine göre kitap araması yapar. Eğer böyle bir kitap varsa o kitabın her yazarı için kullanıcıya o yazarın adını soyadını güncelleme için sorar. Eğer güncelleme yapılmak isteniyorsa yeni ad ve soyad alınır, listede ve dosyada değişiklikler yapılır.

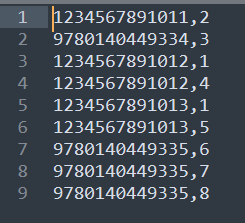
Örnek:

****

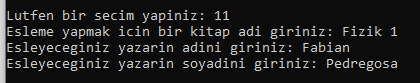
**Seçenek 11:** Bu seçenek bir kitap ve yazar seçimi yapılmasını sağlayarak bir kitabın yazarları listesine bir yazar ekler. Bu seçenek yeni bir kitap tanımlandığında o kitabın yazarlarını tanımlamak için kullanılabilir. Ya da vâr olan bir kitaba yeni bir yazar katıldığında kullanılabilir.

Örneği aşağıdadır:

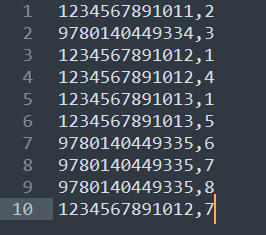
Önceden KitapYazar.csv dosyası:



İşlem:



Sonradan KitapYazar.csv dosyası:



**3. Projenin Kazandırdıkları**

Proje sırasında neredeyse her yerde pointer’ları kullanmamız gerektiğinden dolayı pointer’ları güzelce kavradığımı söyleyebilirim.

Bunun dışında daha önceden C’de dosya işlemlerini bu kadar derinlemesine öğrenmemiştim. Projede csv dosyalarını elimizle parse etmemiz, kullanmamız ve gerektiğinde yeniden yazmamız bana dosya işlemleri ile ilgili bir sürü şey öğretti. Hatta projeye harcadığım sürenin çoğunu dosya işlemlerinin çalışmasını sağlamak için harcadığımı söyleyebilirim. :)

Bunların dışında bir sürü yeni C string’leri ile çalışırken kullanabileceğim fonksiyonları öğrenmiş oldum. Geçmişte C++’da kod yazarken bazı kütüphaneler doğrudan C kütüphanesi olduğundan dolayı C-style string’lerle çalışmam gerekmişti ve çok iyi kod yazamamıştım. Bu projenin bana kattıklarından sonra yine böyle bir C++ projesinde çalışırsam daha az sorun çekeceğime eminim.

**4. Kaynaklar**

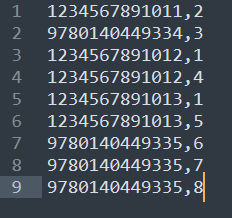
Proje sırasında çok kaynak kullanmadım. Takıldığım yerlerde genelde ders drive’ında bulunan örneklere bakmaya çalıştım.

Derste görmediğimiz strstr, atoi, itoa gibi fonksiyonları öğrenirken google’a fonksiyonların ismini yazıp önüme çıkan ilk siteyi okudum.

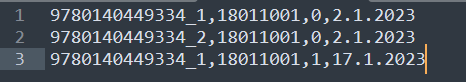
**5. Veritabanının Son Durumu**

Proje sırasında dosyaların bazılarını silip yeniden oluşturduğumdan dolayı rapordaki her işlem teker teker yapılırsa aynı sonuç elde edilmeyebilir. Yine de dosyaların son durumlarını aşağıya koyuyorum. Program KitapOdunc.csv dosyası dışında her dosya ile başlamak zorundadır, yoksa hata verir ve kapanır. KitapOdunc.csv dosyası olabilir de olmayabilir de. Varsa KitapOdunc.csv otomatik olarak okunup işlemler hatırlanır ve sonradan kod devam eder.

KitapYazar.csv:



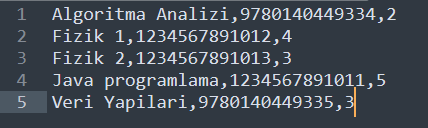
KitapOdunc.csv:



Ogrenciler.csv:



Kitaplar.csv:



Yazarlar.csv:

