Bu projede uzun bir txt dosyasından sayıların okunarak AVL ağacı yapısına yerleştirilmesi ardından da post order okuma işlemi yapılması istenmektedir. Post order okuma yapılırken yapraklarda bulunan sayılar bir yığına sağında veya solunda çocuğu bulunan düğümlerdeki sayıların ise toplanması gerekmektedir. Okuma işlemi sonrasında yığının en üstündeki elemanlar sırayla karşılaştırılacak bir en küçük bir en büyük sayı olacak şekilde yığından çıkarılacaklardır. Eğer bu çıkarma işlemleri sonrasına herhangi bir yığın boşalırsa o yığının AVL ağacı silinecektir. Bu şekilde tüm AVL ağaçları gezilecek ve son bir AVL ağacı kalana kadar döngü devam edecektir. Son AVL ağacı kaldığı zaman düğümlerinde bulunan sayıların toplamı bir işleme sokulacak ve bu sayede bir ASCII karakter elde edilecektir. Programın sonunda son kalan AVL ağacının kaçıncı AVL ağacı olduğu ve ASCII karakteri ekrana yazdırılacaktır.

Projemde kullandığım sınıflar ve sınıflara ait özellikler şu şekilde:

AVLDugum:
- İnt veri
- AVLDugum* sol
- AVLDugum* sag

AVL:
- DugumAVL* root
- İnt AVLToplamDugumDeger
- Yigin* AVLYigin
- Char ascii

DugumYigin:
- İnt veri
- DugumYigin* sonraki

Yigin:
- DugumYigin* tepe

Projemde Veri.txt'den okunan sayılar eğer daha önce yazılmamışsa satır sayısına bağlı olarak oluşturulan işaretçi dizisinde ilgili AVL ağacına yazılıyor. Ardından AVL ağacının yapraklarındaki sayılar ilgili ağacın yığınına yazılıyor. Düğümlerdeki sayılar ise toplanıyor ve ascii karşılığı bulunuyor. Ekrana tüm ascii karakterler yazdırılıyor. Sonra sırayla tüm AVL ağaçlarının yığınları karşılaştırılıyor. Yığının en üstündeki eleman için yapılan bir küçük bir büyük şeklindeki bu karşılaştırma herhangi bir AVL ağacına ait bir yığın bitene kadar devam eder. Yığın bitince ekrandaki harfler siliniyor ve yığını biten AVL ağacının harfi es geçilerek kalan harfler yazılıyor. Ardından yığınlar temizleniyor ve yapraklardaki sayılar tekrar okunarak biten yığın hariç başlangıç haline döndürülüyor. Bu işlem bir döngü halinde geriye tek bir AVL ağacı kalana kadar devam eder.

Sona kalan AVL ağacının indeksi ve ascii karakteri ekrana basılıyor. Program bitiminde kalan tüm yığınlar, root'lar dizi elemanları ve son olarak da işaretçi dizisi yok ediliyor.

Ödevi yaparken program başlangıcında diziye eleman atama işlemini yapan fonksiyonu yazarken referansları nasıl kullanacağımı öğrendim. Bir diğer yandan system("cls") komutu kullanırken oluşan performans kaybına çözüm ararken bu işlemi ANSII karakterleri ile de yapabileceğimi öğrendim ve bu sayede programın performansında iyileşme sağladım. Pointer tutan dizilerin yönetimi ve temizlenmesi konusunda da tecrübe edinmiş oldum.

Proje yazım sürecinde beni en çok zorlayan noktalardan birisi AVL ağacını dengelemek oldu. Bu noktada öğretim elemanlarının GitHub hesaplarında bulunan kodlardan yardım aldım. Bu sorunu çözdükten sonra yığının en tepesindeki elemanları çıkarma aşamasında sağlıklı bir döngü kurarken bolca sorun yaşadım. Örnek vermek gerekirse bir yığın bitip program geri kalan yığınları tekrar okuduğunda mevcut yığına ek yapıyordu. Bu yüzden her yığını okuma yapmadan önce temizlemek zorunda kaldım. Yine döngü esnasında doğru anda programı sonlandırabilmek oldukça karmaşıktı. Bu sorunu ise break ve continue ve basit bir sayaç ekleyerek çözdüm. Özellikle ağaçları gözle görememek hata ayıklama sürecinde oldukça zorlandığım bölümlerden biri oldu. Bu sorunu ile internette bulduğum algoritma görselleştirme ara yüzleri sayesinde aştım.