T.C.
SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

VERİ YAPILARI DERSİ

AVL AĞACINDA EKLEME VE GÜNCELLEME

Ad.....: Berat

Soyad....: Öner

Şube.....: 1B

No.....: B191210066

SAKARYA

ARALIK, 2020

Veri Yapıları Dersi

1. PROGRAM

Programımız bir adet main sınıfı, h ve cpp dosyaları ile birlikte firma ve eleman dosyalarından oluşmaktadır. Firma sınıfımın header dosyasında firmadan pointer ile tanımlı sol, sağ, firmaAd, Eleman sınıfından pointer ile tanımlı eleman nesnesi ve integer türünde derinlik ve calisanSayisi nesneleri bulunmaktadır. Firma header dosyamda AVL ağacı için ağaca ekleme, yeni düğüm ekleme, firma dengeleme, sola ve sağa döndürme, post order sıralama, denge kat sayısı ve yükseklik için fonksiyonlar bulunmaktadır. Eleman sınıfı için bahsedecek olursak farklılık olarak yıl ve adSoyad değişkenlerini içinde tutuyor ve firma sınıfı gibi ekstra dengeleme yapmıyor. Oluşturduğum AVL ağacının yapısından bahsedecek olursam öncelikle çoğu fonksiyonumu sınıf türünde oluşturdum ve ağacı sol-sağ bütünen dolaşabilmek için ağırlıklı olarak recursive fonksiyon kullandım. Ekle metoduma gönderilen düğüm için ağaç dolaşılıyor, gerekli karşılaştırmalar yapıldıktan sonra duruma göre sağa veya sola NULL olan bölgeye ekleniyor. Yeni düğüm ekleme esnasında yeniDugumEkle fonksiyonu çalıştırılıyor, derinliğe ve calışan sayısına 1 değeri atanıyor. Firmada ekle metodu çalışırken ağacın derinliğine ve dengesine göre ağaç sola veya sağa döndürülebiliyor. Eleman sınıfında ağacın derinliğine göre değil, yıla göre döndürme işlemleri yapılıyor. Buradaki denge dengeKatSayisi fonksiyonu içerisinde sağ ve sol köklerinin yüksekliğinin farkından bulunuyor. Main sınıfımdan bahsedecek olursam eğer, dosya okumayı, ağaca veri eklemeyi, ve ekrana yazdırmayı buradan yapıyorum. Öncelikle klasörümde bulunan txt dosyasını ifstream kütüphanesi ile satır satır okuyarak, istediğim değerleri find ve substring fonksiyonları ile alarak dizilere atıyorum. Ekstra olarak satır sayımı tutuyorum. Tek fonksiyon ile dosya okuma işlemimi bitiriyorum. Dosya okurken yıl değeri için string ifadeyi integer ifadeye dönüştürürken oluşturduğum donusturmeFonksiyonu()' nunu kullanıyorum. Bu fonksiyon parametre olarak verdiğim string ifadeyi bana integer olarak geri döndürüyor. Main fonksiyonunun içerisinde dosyaOku() metodu ile dosyam okunup dizilere atanıyor. Ardından satır sayısı kadar for döngüsü içerisinde her bir veri için AVL ağacına ekleme işlemleri gerçekleşiyor. Bu for döngüsünün içerisnde firmaları bir dizide tutuluyor ve her seferinde gelen firmadan var mı diye kontrol ediliyor. Eğer yok ise AVL ağacına firma ekleniyor, ardından ağaca eleman ekleniyor. Her eleman eklenmesinin ardından firmaların dengesi AVL ağacında tekrar dengeleniyor. Bu dengeleme şu şekilde oluyor. Eğer kök düğüm sağ düğümden büyük ise yer değiştiriyor, veya kökün sol düğümü sağ düğümünden büyük ise yer değiştiriyor, veyahut kökün sol düğümü kök düğümden büyük ise yer değiştiriyor. Buradaki büyüklüğü çalışan sayısı ile kontrol ediyorum. Her eleman eklendiğinde o firmanın çalışan sayısı bir arttırılıyor. Tüm bu işlemler yapıldıktan sonra main içerisinde çağrılan yazdir() fonksiyonu ile ağaç postOrder olarak ekrana çıkartılıyor.

2. SONUÇ

Bu projeyi yaparak recursive fonksiyonları kulanmakta kendimi geliştirdim ve recursive fonksiyonların mantığını AVL ağaçları yapısında kavradım. Ayrıca AVL ağaç yapısına eleman eklemeyi ve postOrder veya inOrder okumayı mantıken kavrayarak yaptım.