VERİ YAPILARI 2019-2020 2. ÖDEV RAPORU

Yazdığım program bir C++ konsol uygulamasıdır. Program çalıştığında AVL sınıfından bir nesne oluşturulur. Okuma vb. birtakım işlemler AVL'nin kurucu fonksiyonunda yapılmıştır. AVL'nin kurucu fonksiyonunda, içinde İSİM#DOĞUMYILI#KİLO formatında kişi bilgilerinin yer aldığı Kisiler.txt text dosyasını ifstream ile okur. getline() metodu ile okuma yapılır. # simgesini görünce ilk değeri isim , ikinci değeri doğum yılı son değeri ise kilo değeri olarak tutar. Bu değerler tasarlanan Kişi sınıfına aittir. Doğum yılı değişkeni okunup ,

Ekle(2020- ks->dogumyili) fonksiyonunun çağrılması suretiyle yaş hesaplaması yapılarak AVL ağacına ekleme yapılır.

Ekle fonksiyonu , private olarak tanımlanmış bir başka ekleme fonksiyonuna, aldığı integer değeri gönderir. Burada ise AVL ağacının ekleme , dengeleme işlemlerini yapan algoritmalar bulunur. Yeni bir düğüm oluşturularak başlanır. Eklenecek değerin öncekilerden büyük – küçük – eşit olma durumları koşul ifadeleri ile kontrol edilir. Küçük veya eşit olması durumunda sola , büyük olması durumunda ise sağa ekleme yapılır.

İlk küçük büyük olma durumları *else if (yeni <= dugumParam->veri)* ve *else if (yeni >dugumParam->veri)* ile kontrol edildikten sonra AVL ağacının ana felsefesini oluşturan dengeli bir ikili arama ağacı elde etmek için dengeleme işlemleri yapılır.

Bunun yapılması için gerek koşul,

Yukseklik(dugumParam->sol) - Yukseklik(dugumParam->sag) == 2 durumunun sağlanmasıdır. Yükseklik farkı 0 veya 1 ise dengeleme yapılmazken 2 olduğunda dengeleme işlemleri yapılır.

Bu hususta 4 adet işlemden söz edilebilir. Bunlar: SOL-SOL , SOL-SAĞ , SAĞ-SAĞ , SAĞ-SOL durumlarıdır. Gerekli düğümler için döndürme işlemleri yapılır:

SOL-SOL durumunda sağa dönüş , SOL-SAĞ durumunda sola dönüş + sağa dönüş ,

SAĞ-SAĞ durumunda sola dönüş , SOL-SAĞ durumunda sağa dönüş + sola dönüş işlemleri yapılır.

Bu döndürme işlemlerini yapmak için tanımladığımız fonksiyonlar : *SolCocukIleDegistir(parametre) , SagCocukIleDegistir(parametre)* 'dir.

AVL sınıfında yazılı tüm fonksiyonları sıralamak gerekirse:

void Ekle(const int& yeni) ; void DerinlikGuncelle(Dugum node, int depth);*

void Postorder(Dugum dugumParam) const; int Yukseklik(Dugum* dugumParam) const;*

bool DugumSil(Dugum& dugumParam) ; void Temizle();*

şeklindedir.

Yükseklik dengeleme işlemlerinde kullanılmıştır. DugumSil metodu Temizle metodunda çağrılıp düğümlerin silinmesini sağlayan fonksiyondur. Postorder fonksiyonun görevi ise AVL ağacını dolaşarak kendisine verilen sıra doğrultusunda AVL ağacının üyelerini basmaktır. Basma sırası SOL-SAĞ-KÖK şeklindedir. Bu sıra ile düğümlerin isim , doğum yılı , kilo değerlerini ve Stackte ekli olan elemanlarını basar.

Ödevi hazırlarken kullanılması gereken bir diğer veri yapısı da Stack veri yapısı idi.

Tasarlanan Stack veri yapısı şu fonksiyonları içerir:

bool IsEmpty() const;

void Push(const string& obj);

void Pop();

const string& Top() const;

void StackYazdir(Stack s1);*

void Clear();

IsEmpty() boş – dolu olması durumlarını kontrol ederek true veya false döndürür.,

void Push(const string& obj) Stack'e ekleme yapmamızı sağlayan fonksiyondur. Bu ödev için eklenmesi gereken elemanlar A , Y , D , O harfleri olduğu için parametre olarak string veri tipi verilmiştir.

void Pop() Stack'ten eleman silmemizi sağlayan fonksiyondur. Silme kuralı ise Stack veri yapısına göredir. Yani en son eklenen elemanı siler

string& Top() Stack'in en üstündeki yani en son eklenen elemanı döndürür. Ödev hazırlanırken stack elemanları Top() ve Pop() fonksiyonları yardımıya bastırılmıştır.

StackYazdir(Stack* s1); Yazdırma fonksiyonumuz Top()'ı bastırarak başlar ve tüm elemanlar bitene kadar Pop(); cout<<Top(); döngüsü şeklinde ilerler.

Program sonunda elemanların silinmesini sağlayan Temizle() fonksiyonu çağrıldı. Program dökümanda istendiği gibi MinGW'de derlenecek şekilde hazırlandı. Ekran çıktısı için de yine ödev dökümanında istenen düzen sağlandı. Programın çalışması başta Sabis'te yayınlanan örnek olmak üzere birçok farklı Kisiler.txt dosyasında denendi ve denenen örneklerde progamın hatasız çalıştığı test edildi.

Ödev dosyasında istendiği üzere tüm sınıflar .cpp kaynak dosyası ve .hpp başlık dosyası şeklinde ayrıldı. .hpp dosyalarında yalnızca değişken tanımlamaları ve metot imzaları yer aldı, metot gövdelerine .hpp'de yer verilmedi.

Ödevi hazırlarken en çok zorlandığım konu Stack'lere harf değerlerini Push() etmek için yapılması gereken seviye kontrolünü sağlamak, ekleme öncesi ve sonrasını kıyaslayarak harfleri doğru şekilde eklemek oldu. Bu ödev stack veri yapısı ,ağaç veri yapısı ve çeşitlerinin neler olduğunu, nasıl ve ne amaçla kullanıldıklarını ödevi hazırlamam süresince yaptığım araştırmalarla daha iyi öğrenmemi ve kavramamı sağladı.

Ödevi hazırlarken yararlandığım kaynaklar sanal ders konu anlatımı, ve C/C++ İle Veri Yapıları ve Çözümlü Uygulamalar kitabı oldu. Kitap , Stack ve Ağaç veri yapılarının mantığı, Stack işlemleri, Ağaç işlemleri , ikili ağaçlarda dolaşımın rekürsif işletimi, ağaç ve stack veri yapılarının hem dizi hem de liste gerçekleştirimlerini , konu anlatımı ve örnekleri ile kavramamı sağlayarak ödevimi hazırlamamda yardımcı oldu.

GÜRKAN KAYA G181210102