**YZM 2124**

Veri Yapıları

2022-2023 Bahar Dönemi

**ÖDEV # RAPORU**

Teslim Tarihi: 02.06.2023

Öğrenci Adı: İlayda DİNÇBİLEK

Öğrenci Numarası: 212802040

1. Giriş

Ödevin amacı, bir ikili ağaç yapısı kullanarak plaka kodları ve şehir bilgileriyle ilişkili düğümleri depolamak ve çeşitli işlemler yapmaktır.

Giriş Verileri: "sehir\_plaka\_kodlari.txt" adında bir dosya kullanılır. Bu dosyada her bir satırda bir plaka kodu ve şehir adı çifti bulunur. Bu dosya program tarafından okunur ve plaka kodları ağaca eklenir.

Çıkış Verileri: Ağacın görsel temsili, düğümlerin yerleşimini ve hiyerarşisini gösteren bir çıktı. Ağacın derinliği (depth) ve dengeli olup olmadığına dair bir çıktı. Kullanıcı tarafından belirtilen iki plaka koduna sahip şehirlerin ağaçtaki düzey farkını gösteren bir çıktı. Preorder, inorder ve postorder sıralamasına göre düğümlerin gezilmesi sonucu oluşan çıktılar.

Kullanılan Dosyalar:

- "tree.h": İkili ağaç yapısının tanımlandığı ve Tree sınıfının yer aldığı başlık dosyası.

- "sehir\_plaka\_kodlari.txt": Plaka kodu ve şehir bilgilerinin bulunduğu giriş dosyası.

Kullanılan Metotlar:

- `Insert`: Ağaca yeni bir düğüm ekler.

- `GetTreeWidth`: Ağacın genişliğini hesaplar.

- `GetDepth`: Ağacın derinliğini hesaplar.

- `IsBalanced`: Ağacın dengeli olup olmadığını kontrol eder.

- `FindNode`: Belirli bir plaka koduna sahip düğümü bulur.

- `GetLevelDifference`: İki düğüm arasındaki seviye farkını hesaplar.

- `PreorderTraversal`: Ağacı önceden sıralı olarak dolaşır.

- `InorderTraversal`: Ağacı sırayla dolaşır.

- `PostorderTraversal`: Ağacı sonradan sıralı olarak dolaşır.

- `readFromFile`: Bir dosyadan plaka kodu ve şehir bilgilerini okur ve ağaca ekler.

- `printTree` ve `TreePrinter`: Ağacın görsel temsilini ekrana basar.

- `start`: Programın çalışmasını başlatır.

Program, "sehir\_plaka\_kodlari.txt" dosyasından okunan plaka kodları ve şehir bilgilerini kullanarak bir ikili ağaç oluşturur. Oluşturulan ağaç üzerinde derinlik, dengelilik, gezinme, düzey farkı gibi işlemler yapılır ve sonuçlar ekrana yazdırılır.

1. Geliştirme ve Çalıştırma Ortamları

Ödevi geliştirmek, test etmek ve çalıştırmak için Visual Studio Code ve C++ kullanıldı.

1. Veri Yapıları ve Değişkenler

Ödevde kullanılan veri yapısı bir ikili arama ağacıdır (binary search tree). İkili arama ağacı, elemanların düzenli bir şekilde depolandığı ve hızlı erişim sağladığı bir veri yapısıdır. İkili arama ağacı, ağaç düğümleriyle temsil edilir ve her düğüm, sol ve sağ alt ağaçlara işaret eden iki bağlantıya sahiptir. İkili arama ağacı, sıralı depolama, etkili arama, dengeli bir yapıya sahip olması ve düzey farkı hesaplama gibi özellikleriyle ödevin gereksinimlerini karşılamak için uygun bir veri yapısıdır.

1. Program Akışı

1. `main` fonksiyonu programın başlangıç nokasıdır. Bu noktada `Tree` sınıfından bir nesne oluşturulur.

2. Oluşturulan `Tree` nesnesi `start` metodu çağrılarak programın ana işlevi başlatılır.

3. `start` metodu içinde `readFromFile` metodu çağrılarak "sehir\_plaka\_kodlari.txt" dosyasından veriler okunur ve ikili arama ağacına eklenir.

4. Ardından, `TreePrinter` metodu çağrılarak oluşturulan ikili arama ağacı görsel olarak ekrana yazdırılır.

5. `GetDepth` ve `IsBalanced` metotları kullanılarak ikili arama ağacının derinliği ve dengelilik durumu hesaplanır ve ekrana yazdırılır.

6. Kullanıcıdan 1. ve 2. şehir plaka kodları istenir ve `FindNode` ve `GetLevelDifference` metotları kullanılarak bu plaka kodlarına sahip düğümler bulunur ve düzey farkları hesaplanır. Bulunan sonuçlar ekrana yazdırılır.

7. `PreorderTraversal`, `InorderTraversal` ve `PostorderTraversal` metotları kullanılarak ikili arama ağacının ön sıralama, iç sıralama ve son sıralama gezinmeleri yapılır ve sonuçlar ekrana yazdırılır.

1. Sonuç

Ödevi geliştirirken ikili arama ağacı veri yapısını daha iyi anlama fırsatı buldum. Programda kullanılan ağaç görselleştirme yöntemi sayesinde, bir ağacı metinsel bir şekilde ekrana çizdirmeyi öğrendim. Bu, veri yapısının yapısını daha iyi görselleştirmeme ve anlamama yardımcı oldu. Program, bir dosyadan veri okuma ve bu verileri işleme gerektiriyor ve bu sayede, dosya işleme konusundaki becerilerimi geliştirdim. Program sayesinde ağacın derinliğini ve dengelilik durumunu hesaplama, ağaçta düğüm arama, gezinme işlemleri gibi farklı algoritma ve veri yapısı konularına pratik uygulamalar yapabildim. Programda kullanıcıdan plaka kodu girişi alınırken hata kontrolü yapılması gerekiyor. Bu sayede, kullanıcı girişlerini doğrulama ve hatalı girişleri yönetme becerilerimi de geliştirdim.