**YZM 2124**

VERİ YAPILARI

**2022-2023 BAHAR DÖNEMİ**

**2.ÖDEV RAPORU**

Teslim Tarihi: 02.06.2023

Öğrenci Adı: Ramazan YILDIRIM

Öğrenci Numarası:

1. **Giriş**

Bu ödevin amacı “sehir\_plaka\_kodlari.txt” dosyasında bulunan plaka kodlarıyla bir binary tree oluşturup bu binary tree’nin görüntüsüne, dengeli olup olmadığına, derinliğine ve preorder, inorder ve postorder olarak sıralamak ve tüm bunları ekrana yazdırmaktır.

Aynı zamanda console ekranı üzerinden kullanıcıdan alınan iki plakaya göre bu plakaların şehir olarak karşılıklarını ve ağaç üzerindeki düzey farkını yazdırmak da bu ödevin amacıdır.

1. **Geliştirme ve Çalıştırma Ortamı**

Bu ödev, MacOS işletim sisteminin 13.4 Ventura sürümünde, Visual Studio Code IDE’si ile geliştirilmiştir.

1. **Veri Yapıları, Değişkenler ve Metotlar**

**struct Dugum:** Binary tree veri yapısı için kullanılan düğüm yapısını tanımlar.

**Dugum\* add\_To\_Tree(Dugum\* node, int plaka, string sehir):** Ağaca yeni bir düğüm eklemek için kullanılır.

**Dugum\* find(Dugum\* node, int plaka):** İstenilen plaka koduna sahip bir düğümü bulmak için kullanılır. Arama işlemi ağacın her iki alt ağacında da gerçekleştirilir ve düğüm bulunduğunda veya sona erdiğinde sonuç döndürülür.

**int Derinlik\_Bul(Dugum\* node):** Ağacın derinliğini hesaplamak için kullanılır.

**bool isDengeli(Dugum\* node):** Ağacın dengeli olup olmadığını kontrol etmek için kullanılır.

**void Agac\_Yazdir(Dugum\* node, int j = 0):** Ağacı ekrana düzey düzey yazdırmak için kullanılır.

**int Duzey\_Farki(Dugum\* node, int plaka1, int plaka2):** İki plaka koduna sahip düğümün ağaçtaki düzey farkını hesaplamak için kullanılır.

**void Preorder\_Yazdir(Dugum\* node):**Ağacı preorder yazdırır.

**void Inorder\_Yazdir(Dugum\* node):**Ağacı inorder yazdırır.

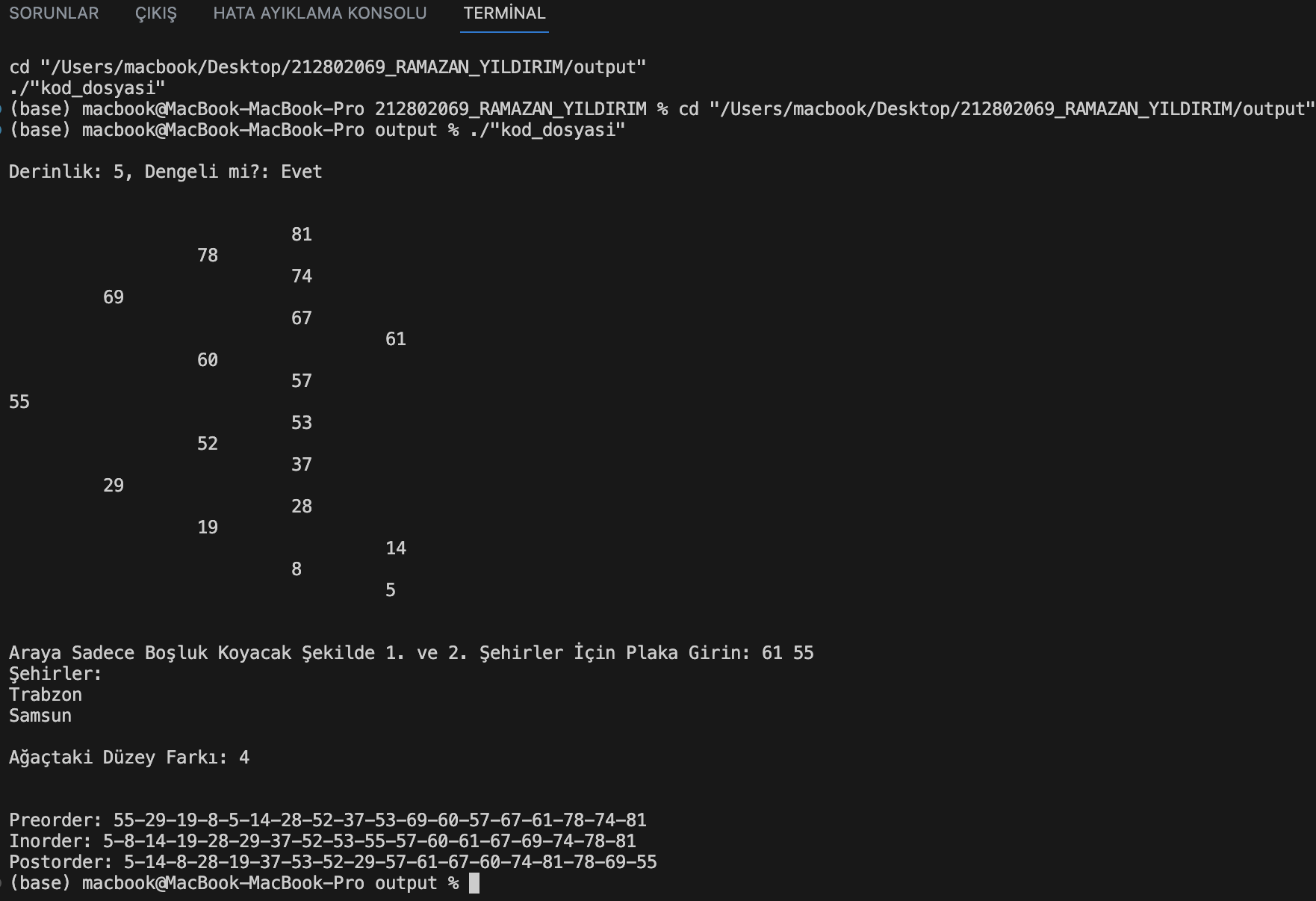
**void Postorder\_Yazdir(Dugum\* node):**Ağacı postorder yazdırır.

1. **Program Akışı**

* İlk olarak, root adında bir düğüm işaretçisi tanımlanır ve NULL olarak atanır.
* “sehir\_plaka\_kodlari.txt” dosyası okunur. Bu dosyadaki plaka ve şehir bilgileri kullanılarak yukarıda görevi tanımlanan add\_To\_Tree() fonksiyonuyla yeni düğümler ağaca eklenir.
* Yukarıda görevleri tanımlanan Derinlik\_Bul( ), isDengeli( ), Agac\_Yazdir( ) fonksiyonları çağrılıp bu fonksiyonların cevapları ekrana yazdırılır.
* Kullanıcıdan iki tane plaka istenir.
* Bu plakalar kullanılarak find( ) fonksiyonuyla ilgili düğümler bulunur ve şehir isimleri ekrana yazdırılır.
* Yukarıda görevleri tanımlanan Duzey\_Farki(), Preorder\_Yazdir(), Inorder\_Yazdir() ve Postorder\_Yazdir() fonksiyonları çağrılır ve bu fonksiyonların cevapları ekrana yazdırılır.
* Program sonlanır.

1. **Sonuç**

Elde edilen kazanımları belirtmeden önce hazırlanan programın bir örnek üzerinden çalıştırılmasında fayda var.



Yazmış olduğum programa göre klavyeden ağaç üzerinde bulunan 61 ve 55 plakalarını girdiğim taktirde program bana bu şehirlerin isimlerini ve ağaca göre aralarındaki düzey farkını yazdırdı.

Aynı zamanda program ve kod çıktısında da anlaşılabileceği üzere bir ağacın dengeli olup olmadığının, derinliğinin, preorder, inorder, postorder yazdırılmasının nasıl olduğunu öğrendim.

Son olarak rekülsif fonksiyon ve dosya işlemleri diyebilirim. Yazmış olduğum bazı fonksiyonlar - “add\_To\_Tree( )”, “find( )”, “Derinlik\_Bul( )”, “isDengeli( )” gibi - rekülsiftir.

“sehir\_plaka\_kodlari.txt“ dosyası, şehir ve plaka bilgilerini okumak için kullanılır. Dosya okuma işlemi, dosyayı açmak, satır satır okumak ve bilgileri ağaca eklemek gibi adımlardan oluşur.