BINARY SEARCH TREE SIRALAMA PROJESİ

Veri Yapıları Dersi

Ad Soyad: İlayda Meryem Coşkun

Numara: 241201019

Bölüm: Bilgisayar Mühendisliği

PROJE AÇIKLAMASI

Bu projede, İkili Arama Ağacı (Binary Search Tree - BST) kullanarak bir sıralama algoritması uygulanmıştır. Projenin temel mantığı, sıralanmamış verileri bir BST yapısına ekleyerek, daha sonra in-order (sıralı) dolaşım ile sıralanmış şekilde elde etmektir.

Projenin Temel Özellikleri:

- **BST Düğüm Yapısı**: Her düğüm bir değer, sol çocuk ve sağ çocuk referansı içerir.

- **Ekleme Algoritması**: Küçük değerler sol tarafa, büyük veya eşit değerler sağ tarafa eklenir.

- **Dolaşma Fonksiyonları**: In-order, Pre-order ve Post-order dolaşım fonksiyonları uygulanmıştır.

- **Ek Özellikler**: Düğüm sayısı hesaplama ve ağaç yüksekliği bulma fonksiyonları eklenmiştir.

PYTHON KODU:

```python
class Dugum:
    def __init__(self, veri):
        self.deger = veri
        self.sol = None
        self.sag = None

def ekle(kok, veri):
    if kok is None:
        return Dugum(veri)
    gecici = kok
    while True:
        if veri < gecici.deger:
            if gecici.sol is None:
                gecici.sol = Dugum(veri)
                break
            gecici = gecici.sol
        else:
            if gecici.sag is None:
                gecici.sag = Dugum(veri)
                break
            gecici = gecici.sag
    return kok

def in_order(kok):
    if kok:
        in_order(kok.sol)
        print(kok.deger, end=" ")
        in_order(kok.sag)

def pre_order(kok):
    if kok:
        print(kok.deger, end=" ")
        pre_order(kok.sol)
        pre_order(kok.sag)
```

```python
def post_order(kok):
    if kok:
        post_order(kok.sol)
        post_order(kok.sag)
        print(kok.deger, end=" ")

def dugum_sayisi(kok):
    if kok is None: return 0
    return 1 + dugum_sayisi(kok.sol) + dugum_sayisi(kok.sag)

def yukseklik(kok):
    if kok is None: return -1
    return 1 + max(yukseklik(kok.sol), yukseklik(kok.sag))

girdi = input("Lutfen sayilari aralarinda bosluk birakarak yazin: ")
liste = [int(s) for s in girdi.split()]

root = None
for x in liste:
    root = ekle(root, x)

print("\nSONUCLAR:")
print("Sirali Liste (In-Order):")
in_order(root)
print("\nPost-Order:")
post_order(root)
print("\nPre-Order:")
pre_order(root)

print(f"\n\nToplam Dugum Sayisi: {dugum_sayisi(root)}")
print(f"Agac Yukseklig: {yukseklik(root)}")
```

KODUN ÇIKTISI:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                zsh

Lutfen sayilari aralarinda bosluk birakarak yazin: 8 3 10 1 6 14 4 7 13

SONUCLAR:
Sirali Liste (In-Order):
1 3 4 6 7 8 10 13 14
Post-Order:
1 4 7 6 3 13 14 10 8
Pre-Order:
8 3 1 6 4 7 10 14 13

Toplam Dugum Sayisi: 9
Agac Yukseklig: 3
ilaydameryemcoskun@Ilayda-MacBook-Air Binary Search Tree %
```

KOD AÇIKLAMASI

1. Dugum Sınıfı

Dugum sınıfı, BST'nin temel yapı taşıdır. Her düğüm üç özellik içerir:

- deger: Düğümde saklanan sayı
- sol: Sol çocuk düğümün referansı
- sag: Sağ çocuk düğümün referansı

2. Ekle Fonksiyonu

ekle(kok, veri) fonksiyonu, BST'ye yeni bir değer ekler. Fonksiyon iteratif (döngü tabanlı) bir yaklaşım kullanır:

- Eklenecek değer mevcut düğümden küçükse sol tarafa gidilir
- Eklenecek değer mevcut düğümden büyük veya eşitse sağ tarafa gidilir
- Uygun boş konum bulunduğunda yeni düğüm eklenir

3. In-Order Dolaşım

in_order(kok) fonksiyonu, ağacı özyinelemeli (recursive) olarak dolaşır:

- Önce sol alt ağaç ziyaret edilir
- Sonra kök düğüm işlenir (yazdırılır)
- En son sağ alt ağaç ziyaret edilir
- Bu sıralama BST'de değerlerin küçükten büyüğe sıralanmasını sağlar

4. Pre-Order Dolaşım

pre_order(kok) fonksiyonu, ağacı Kök-Sol-Sağ sırasıyla dolaşır:

- Önce kök düğüm işlenir
- Sonra sol alt ağaç ziyaret edilir
- En son sağ alt ağaç ziyaret edilir
- Bu yöntem ağacı kopyalamak için kullanılır

5. Post-Order Dolaşım

post_order(kok) fonksiyonu, ağacı Sol-Sağ-Kök sırasıyla dolaşır:

- Önce sol alt ağaç ziyaret edilir
- Sonra sağ alt ağaç ziyaret edilir
- En son kök düğüm işlenir
- Bu yöntem ağacı silmek için kullanılır

6. Düğüm Sayısı Fonksiyonu

dugum_sayisi(kok) fonksiyonu, özyinelemeli olarak tüm düğümleri sayar:

- Boş düğüm için 0 döner
- Her düğüm için 1 + sol alt ağacın düğümleri + sağ alt ağacın düğümleri

7. Yükseklik Fonksiyonu

yukseklik(kok) fonksiyonu, ağacın en uzun dalının uzunluğunu hesaplar:

- Boş düğüm için -1 döner
- Her düğüm için 1 + max(sol yükseklik, sağ yükseklik)