SAKARYA ÜNİVERSİTESİ

Veri yapıları

2024-2025 güz dönemi

Ödev Raporu

SELİM ALTIN	**G231210558 2.C**
selim.altin@ogr.sakarya.edu.tr	

2024

BİLGİSAYAR MÜHENDİSLİĞİ

Ödev Raporu: İkili Arama Ağacı ve Bağlı Liste Uygulaması

Proje Konusu ve İstenilenler

Bu proje kapsamında, dosyadan okunan verilerle ikili arama ağaçları oluşturulması ve bu ağaçların bağlı bir liste yapısında organize edilmesi istenmiştir. Projenin temel hedefleri:

1. Dosyadan okuma ve her satır için ayrı bir ikili arama ağacı oluşturma.
2. Ağaçları bir bağlı listeyle birbirine bağlama.
3. Kullanıcı etkileşimli bir arayüz sağlayarak liste gezinme, ağaç silme ve aynalama işlemleri yapma.
4. Her ağacın toplam değerini hesaplama ve görselleştirme.

Öğrendiklerim

Bu ödev süresince birçok önemli bilgisayar bilimi konsepti öğrendim ve pratik ettim:

1. **Veri Yapıları:** İkili arama ağaçlarının yapısını, veri ekleme ve silme işlemlerini derinlemesine kavradım.
2. **Bağlı Listeler:** Dinamik bağlı listeler oluşturmayı, liste düğümleri üzerinde gezinmeyi ve listeyi yönetmeyi öğrendim.
3. **Dinamik Bellek Yönetimi:** Bellek tahsisi ve serbest bırakma konularında kapsamlı bilgi edindim.
4. **Kullanıcı Arayüzü Tasarımı:** Konsol tabanlı bir arayüzde kullanıcı etkileşimini yönetmeyi ve geri bildirim sağlamayı öğrendim.
5. **Görselleştirme:** İkili arama ağacını hiyerarşik bir yapıda görselleştirmenin önemini ve zorluklarını deneyimledim.

Proje Sürecinde Yapılanlar

1. İkili Arama Ağacı Tasarımı

- **TreeNode** yapısı ile her düğümde bir karakterin saklandığı bir yapı oluşturuldu.
- **BinaryTree** sınıfında:
    - insert fonksiyonu ile her karakterin doğru sırada eklenmesi sağlandı.
    - calculateTotalValue fonksiyonu ile her ağacın toplam değeri hesaplandı. Sol düğümlerin 2 katı ile sağ düğümlerin bire bir katkısı toplandı.
    - mirrorTree fonksiyonu ile ağacın aynalanması sağlandı.
    - printTree ile ağaç görselleştirildi. Görsel düzenleme ve hizalamalar, seviyeler ve yatay bağlantılar eklenerek geliştirildi.

2. Bağlı Liste Tasarımı

- **LinkedList** sınıfı ile ikili arama ağaçlarının birbirine bağlandığı bir yapı oluşturuldu.
- Her düğüm, bir ağacı temsil etti. addNode ve removeNode ile liste dinamik olarak güncellendi.
- Kullanıcı, "a" ve "d" tuşlarıyla listedeki düğümler arasında gezinebildi.
- printListWithArrows fonksiyonu ile liste, seçili düğümün oklarla işaretlendiği şekilde yazdırıldı.

3. Dosya İşleme

- **Processor** sınıfı ile dosyadan satır satır okuma ve her satır için bir ağaç oluşturma işlemi yapıldı.
- Dosyada yapılan değişiklikler RewriteFile fonksiyonu ile kaydedildi.

4. Kullanıcı İşlemleri

- Kullanıcı "w" tuşuna bastığında, seçili ağacın aynalanması sağlandı.
- "s" tuşu ile seçili düğüm silindi ve liste dinamik olarak güncellendi.
- Silme işleminde, son ağaç silindiğinde "Ağaç yok" mesajı gösterildi ve uygulama çökmemesi sağlandı.
- Liste başına ve sonuna ulaşıldığında uygun mesajlar verildi.

5. Görselleştirme

- Ağaçların her seviyesi arasında bağlantı çizgileri (-----) ile görsel ayrım sağlandı.
- Boş düğümler için yer tutucu boşluklar kullanıldı.

- Düğümler arasındaki mesafe, seviyeye göre dinamik olarak ayarlandı.

Karşılaşılan Zorluklar

1. **Toplam Değer Hesaplama:**
    - İlk başta, ASCII değerlerinin doğru şekilde toplandığından emin olunmakta zorlandık.
    - Sol düğümlerin 2 katı katkısı, sağ düğümlerin ise bire bir katkısı eklenerek problem çözüldü.
2. **Liste Yönetimi:**
    - Liste düğümleri arasında dinamik olarak gezinmek ve sayfalama (pagination) özelliğini eklemek zordu.
    - Liste 10 düğümle sınırlı gösterildi ve sayfa değiştirme işlemleri "a" ve "d" tuşları ile düzenlendi.
3. **Ağaç Görselleştirme:**
    - Ağaç seviyeleri arasındaki boşluk ve hizalama işlemleri başlangıçta karmaşıktı.
    - Çizgiler ve düğümler arasındaki boşluklar düzenlenerek görsel çıktılar geliştirildi.

Eksik Kalan Yerler

1. **Görselleştirme İyileştirmeleri:**
    - Çizim tam anlamıyla beklenen formatta değil. Daha profesyonel bir çıktıya ihtiyaç olabilir.
2. **Kullanıcı Dostu Arayüz:**
    - Konsol tabanlı arayüz, kullanımı kolaylaştırmak için daha fazla geliştirme gerektiriyor.

Sonuç

Bu proje, veri yapıları ve algoritmalar konularında teorik bilgileri pratikle birleştirmek için mükemmel bir fırsat sundu. İkili arama ağacı, bağlı liste ve dinamik bellek yönetimi gibi temel konuları derinlemesine öğrenmemi sağladı. Proje boyunca birçok zorlukla karşılaşmamıza rağmen, bu zorlukları çözerek projeyi büyük ölçüde başarıyla tamamladık.