**YZM 2124**

Veri Yapıları

**ÖDEV # RAPORU**

Öğrenci Adı: Hakan Bayazıt Habeş

**1.Giriş**

Ödevin amacı, ikili sıralama ağacı oluşturarak İlk Derinlik Arama (Depth First Search – DFS) ve İlk Genişlik Arama (Breadth First Search – BFS) algoritmalarını geliştirmektir. Geliştirilen arama algoritmaları ile oluşturulacak üç farklı ağaç üzerinde arama işlemleri gerçekleştirilecektir. Bu arama işlemleri için hangi sıralama ağacının daha uygun yapıda olduğu ve hangi arama algoritmasının daha performanslı çalıştığı araştırılacaktır.

**Kullanılan Methotlar:**

struct Dugum { //Agacta yer alan her bir dugumun veri yapisi

int veri; //Dugumde tutulan veri

Dugum\* sag; //Dugumun sag cocuguna isaretci

Dugum\* sol; //Dugumun sol cocuguna isaretci

};

struct IkiliSiralamaAgaci {

Dugum\* kok; //Agac kokune isaretci

void agacKur(int\*); //Dizi isaretcisi alir ve dizinin elemanlarini ikili siralama agacina uygun sekilde yerlestirir

void agacKapat(); //Agacin dugumleri icin alinan bellek gozlerini program sonlanmadan hemen once iade eder

};

bool DFS(Dugum\*, int); //Ilk derinlik arama fonksiyonu bir agac koku ve aranacak deger parametrelerini alir

bool BFS(Dugum\*, int);//Ilk genislik arama fonksiyonu bir agac koku ve aranacak deger parametrelerini alir

**2. Geliştirme ve Çalıştırma Ortamları**

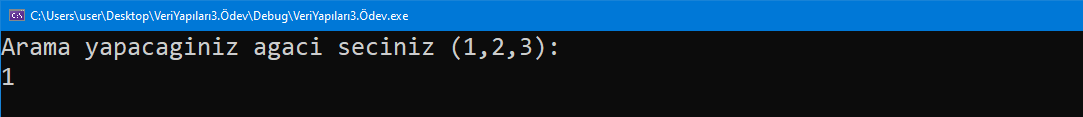
Ödev gerçekleştirilirken Windows makine ve Visual studio uygulaması üzerinde geliştirilmiştir.

**3. Veri Yapıları ve Değişkenler**

Ödevin gerçekleştirilmesi aşamasında ikili Ağaç (binary tree) ve Kuyruk ve Yığın veri yapıları kullanılmıştır. Binary Tree, bilgisayar bilimlerinde en çok kullanılan ağaç veri yapılarından olup, sıralı olması durumunda arama, ekleme ve silme işlemlerini çabuklaştırırlar. Kuyruk ve Yığın (Depth First Search – DFS) ve İlk Genişlik Arama (Breadth First Search – BFS) algoritmalarını oluşturmak için kullanılacaktır.

**4. Program Akışı**

1 🡪 Kullanıcı dosyayı açar ve işlem yapmak istediği ağacı seçer.



2 🡪 Daha sonra seçtiği ağaç ile arama algoritması seçer.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

3 🡪 Son olarakta seçilen ağaç üzerinde seçilen algoritma üzerinde aranacak değer girilir.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**5. Sonuç**

Ödevde sonuç olarak, oluşturduğumuz ağaçlardan denge durumuna göre performanslarından beklentim ağaç 1 ve ağaç 2’nin çalışma hızının birbirine yaklaşık bekliyordum, ağaç 3’ün ise biraz daha yavaş çalışmasını bekliyordum. Ödevi yapıp çalıştırdıktan sonra ağaç 3’ün diğerlerinden daha hızlı çalıştığını gözlemledim. Benim bilgisayarımdan kaynaklanabilir ikiside birbirine uç fark olmadan yakın çıkıyor. Ağaç 2 biraz daha yavaş kalıyor.

Depth First Search – DFS, Breadth First Search – BFS aramasında ağacın ilk değerlerinde büyük fark gözlemlemedim. Ağacın son değerlerine doğru breadth first search’ün daha hızlı çalıştığını gözlemledim.