BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
BM208 VERİ YAPILARI VE ALGORİTMALAR DERSİ
UYGULAMA FÖYÜ

3. Deney Föyü: Ağaçlar (İkili Ağaç)

Adı Soyadı:		Öğrenci Numarası:		30.04.2019

1. Amaç:

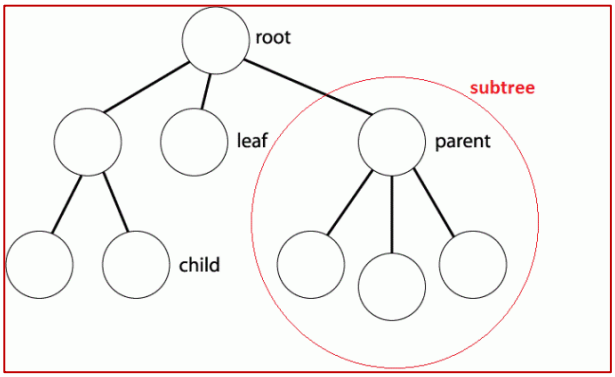
Ağaç veri yapısının incelenmesi, ikili ağaç ağaç veri yapısı içinde dolaşımlarının incelenmesi

2.Uygulama Dersinden Önce Yapılması Gerekenler

Ağaç veri yapısının nasıl oluştuğunun incelenmesi gerekmektedir. İkili ağaç düğümleri arasında dolaşımın hangi algoritmalar ile yapıldığının incelenmesi gerekmektedir. İkili ağaç veri yapısında ekleme, silme ve arama işlemlerinin nasıl yapıldığının incelenmesi gerekmektedir.

3.Uygulamaya Hazırlık:

Bir ağacın her bir elemanına node/düğüm denir.Veriler node'larda tutulur.
Düğümler birbirlerine edge/kenar/dal ile bağlanır.Bu bağlantılar iki node arasındaki ilişkiyi gösterir.
Düğümler ağaca seviyeler oluşturacak şekilde yerleştirilirler,
yani ağaçlar hiyerarşik bir yapıya sahiptir.Bu hiyerarşik yapının en tepesindeki düğüm; **root/kök,**
En ucundaki, çocuğu olmayan düğümler; **leaf/yaprak**,
Çocuğu olan düğümler; **parent/ebeveyn**, çocuklar ise **child/çocuk** olarak adlandırılır.
Bir düğümün alt ağaçlarına **subtree** denir.
Aynı parent'a sahip düğümler birbirlerinin **sibling/kardeş** node'larıdır.
Bir düğümden köke kadar izlenen yoldaki diğer tüm düğümler, o düğümün **ancestor/atalarıdır**. Bir düğümün çocuklarına bağlı olan tüm düğümler, o düğümün **descendant/torunlarıdır**.
Bir düğüme ulaşmak için üzerinden geçilen düğümler listesine path/yol/iz denir.
Düğümün depth/derinliği, o düğümden kök düğüme kadar olan yolun uzaklığıdır. (Kök düğümünün derinliği 1 kabul edilerek örnekleme yapılacaktır).
Bir ağacın **derinliği** kök düğümün en uçtaki yaprak düğüme olan uzaklığıyla ölçülür.



Düğümün **height/yüksekliği**, o düğümden kendisiyle ilişkili en uzak yaprak düğüme kadar giden yolun uzunluğudur.
Düğümün **level/düzey/seviyesi**, kök ve ilgili düğüm arasında bulunan düğümlerin sayısına eşittir.
Bir node'un **degree/derecesi**, çocuk sayısına eşittir.
Ağaçlar **döngü/cycle** içermezler.
Bir ağacın hiç düğümü yoksa, **boş/empty tree** olarak adlandırılır.

```c
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<stdio.h>

// Ağaç ekleme fonksiyonuna ait prototip
struct agac *Ekleme(struct agac *,int);

// Ağaç için oluşturulan structure
struct agac
{
 int bilgi;
 struct agac *solDugum;
 struct agac *sagDugum;
};

int main(void)
{
 struct agac *anaDugum;
 int  oge, sayac=0;
 anaDugum = NULL;
 // Agaca 5 eleman ilave ediliyor
 for(sayac=0;sayac<5;sayac++)
 {
 printf("\nYeni eleman ekle:");
       scanf("%d", &oge);
       anaDugum= Ekleme(anaDugum,oge);
  }
 return(0);
}
```

```c
/* Eleman Ekleme Fonksiyonu*/
struct agac *Ekleme(struct agac *anaDugum, int x)
{
 if(!anaDugum)
  {
    anaDugum=(struct agac*)malloc(sizeof(struct agac));
    anaDugum->bilgi = x;
    anaDugum->solDugum = NULL;
    anaDugum->sagDugum = NULL;
    return(anaDugum);
  }
  if(anaDugum->bilgi > x)
     anaDugum->solDugum = Ekleme(anaDugum->solDugum,x);
  else
   {
     if(anaDugum->bilgi < x)
    anaDugum->sagDugum = Ekleme(anaDugum->sagDugum,x);
   }
   return(anaDugum);
 }
```

4.Örnek Uygulama Kodu:

Örnekte ikili ağaç veri yapısının oluşturulması için, **agac** adında bir structure yapısı tanımlanmış ve ikili ağaca bir eleman eklemek için gerekli olan fonksiyon oluşturulmuştur. Oluşturulan ağaca Ekleme fonksiyonu ile 5 eleman ilave edilmiştir.

- Uygulama derslerinde ilgili deney föyünü **mutlaka** yanınızda **bulundurunuz.** **Arkalı önlü çıktı alabilirsiniz.**
- Adınızı, Soyadınızı ve Öğrenci Numaranızı **mutlaka yazınız.**
- Uygulama sorusunun cevabı doğru ise kâğıdın arka sayfasına **yazıp,** kodunuzu kısaca **açıklayınız**.

5. Uygulama Sorusu :

Değerlendiren Öğretim Elemanı:		Çalışma Durumu:	0	1	2	3	4	5	Not: