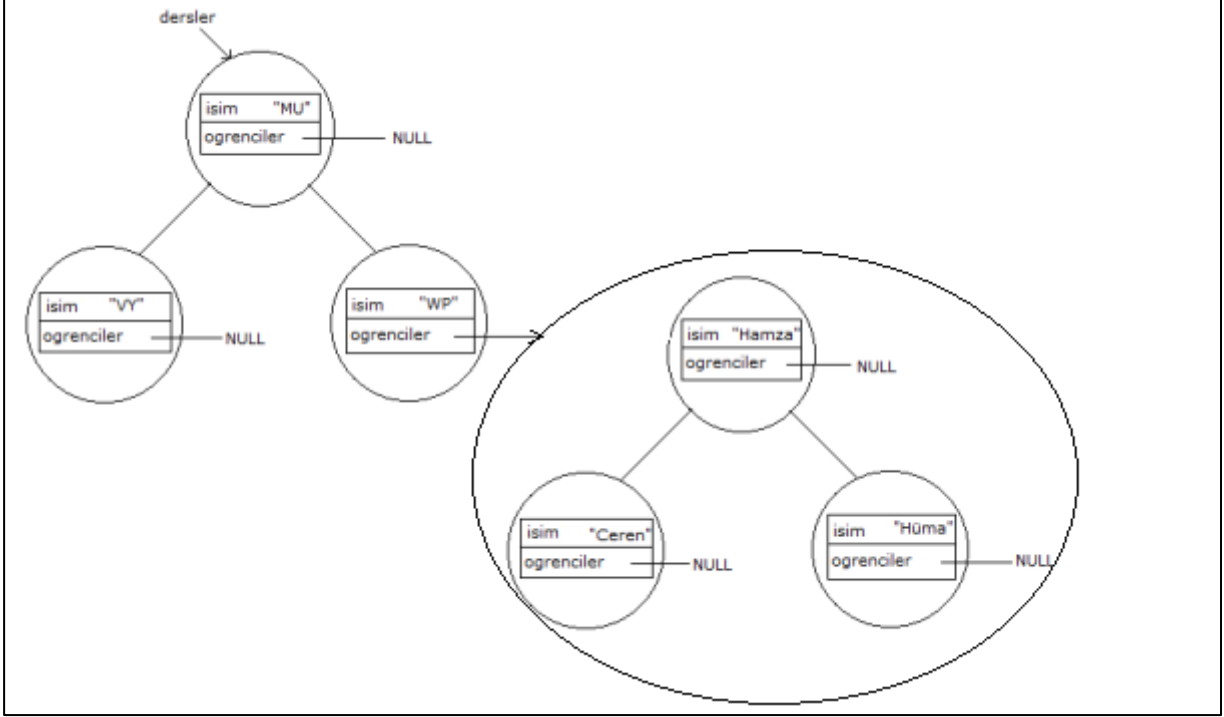
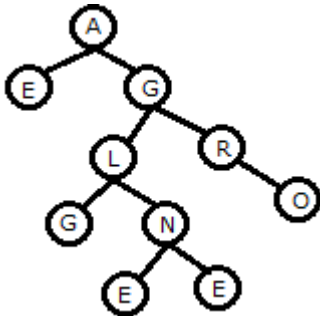


- 1- Aşağıda resmi verilen dersler isimli İkili Arama Ağacının Düğüm sınıfını yazınız. Başlık ve kaynak dosyası şeklinde ayırm yapmanıza gerek yok (20 p).



```
class Dugum{
public:
    string isim;
    Dugum *sol;
    Dugum *sag;
    IkiliAramaAgaci *ogrenciler;
    Dugum(string ism,Dugum *sl=NULL,Dugum *sg=NULL,IkiliAramaAgaci *ogr=NULL) {
        isim=ism;
        sol=sl;
        sag=sg;
        ogrenciler = ogr;
    }
};
```

- 2- Aşağıdaki İkili ağacı preorder, inorder ve postorder şeklinde dolaşınız (15 p).



**PREORDER: A E G L G N E E R O**

**INORDER: E A G L E N E G R O**

**POSTORDER: E G E E N L O R G A**

- 3- Verilen arayüzlere göre, dikdörtgen içerisinde bulunan boşlukları doldurunuz. Açıklamalar arayüz tanımları içerisinde verilmiştir (20 p).

```
void HashTablosu::ekle(const string &giris)
{
    depolamaBirimi[hashKoduUret(giris)]->insert(depolamaBirimi[hashKoduUret(giris)]->begin(),giris);
}

void HashTablosu::yazdir(const string &giris)
{
    depolamaBirimi[hashKoduUret(giris)]->print(depolamaBirimi[hashKoduUret(giris)]->begin());
}
```

- 4- Bilgi.txt dosyası incelendiğinde aşağıdaki karakter ve frekanslar ortaya çıkmaktadır. Bu dosyayı Huffman kodlamayı kullanarak Huffman ağacını çiziniz. Her karakterin kodunu bulunuz. Dosyanın durumu ilk durumda ve sıkıştırıldıktan sonra kaç bit olmuştur ayrı ayrı yazınız. Ayrıca dosyada geçen **QEZA** kelimesi sıkıştırılmadan önce ve sıkıştırıldıktan sonra kaç bit olduğunu belirtiniz (15 p).

Karakter	Frekans
E	32
T	28
A	20
Q	4
Z	1

E: 0  
T: 11  
A: 101  
Q: 1001  
Z: 1000  
İlk durumda=  $32*8 + 28*8 + 20*8 + 4*8 + 1*8 = 680$   
Sıkıştırılmış=  $32*1 + 28*2 + 20*3 + 4*4 + 1*4 = 168$   
QEZA=  $8 + 8 + 8 + 8 = 32$   
QEZA(Sıkıştırılmış)=  $4 + 1 + 4 + 3 = 12$

5- Aşağıda kodu verilen **ve elemanlarının küçükten büyüğe sıralı olduğu** tek yönlü bağlı listenin ekle metodunu yazınız (15 p).

<pre> <b>struct dugum</b> {     int bilgi;     dugum *sonraki; };  <b>class Liste</b> {     private:         dugum *ilk;     public:         Liste()         {             ilk=NULL;         }         void yaz()         {             dugum *gecici=ilk;             while(gecici!=NULL)             {                 cout&lt;&lt;"Deger:"&lt;&lt;gecici-&gt;bilgi&lt;&lt;endl;                 gecici=gecici-&gt;sonraki;             }         }     } </pre>	<pre> <b>void ekle(int deger)</b> {     dugum *yeni=new dugum;     yeni-&gt;bilgi=deger;     yeni-&gt;sonraki=NULL;     if(ilk==NULL) ilk=yeni;     else     {         dugum *gecici=ilk;         dugum *onceki=NULL;         while(gecici!=NULL&amp;&amp;yeni-&gt;bilgi&gt;gecici-&gt;bilgi)         {             onceki=gecici;             gecici=gecici-&gt;sonraki;         }         if(onceki==NULL)         {             yeni-&gt;sonraki=ilk;             ilk=yeni;         }         else {             onceki-&gt;sonraki=yeni;             yeni-&gt;sonraki=gecici;         }     } } </pre>
--	--

6- Aşağıdaki ikili arama ağaçlarından Q düğümünün çıkarılması halinde ağaçları yeniden çizin. (10 p)

<p><b>Q'nun Solunun en sağındaki düğüm ya da Q'nun sağın en solundaki düğüm Q'nun yerine getirilir. P ve ya R doğru cevaptır.</b></p>	<p><b>Q'nun Solunun en sağındaki düğüm ya da Q'nun sağın en solundaki düğüm Q'nun yerine getirilir. P ve ya R doğru cevaptır.</b></p>

7- Bankalardaki müşterileri kabul etme sırası düşünüldüğünde (Kredi kartı ile fiş almak gibi.) en uygun veri yapısı **Öncelikli Kuyruk** verilebilir. (5 p)

Kuyruk yazan 5 puan

Öncelikli Kuyruk yazan 10 puan