

1. Aşağıdaki işlemlerin hangisinde veya hangilerinde linkli liste kullanmak dizi kullanımına göre **genel olarak daha avantajlıdır?** (Örnek Cevap: 1,3,5) (4 puan, tahmini süre: 1 dk.)



1. Ekleme
2. Silme
3. Güncelleme
4. Arama
5. Min-Max Bulma

Cevap: 1,2

2. 5,8,9,12,24,48,64 şeklinde verilen bir sıralı diziden İkili Arama Ağacı oluşturulması isteniyor. Bu diziden belirli bir kurala göre oluşturulan ağaç;

- en iyi durumda dengelidir ve ..... seviyeden oluşur.
- en kötü durumda dengesizdir ve ..... seviyeden oluşur.

Oluşturulan ağacın kökünde ise;

- en iyi durumda ..... sayısı,
- en kötü durumda ise ..... sayısı bulunur.

**Not:** Hesaplamalarda kök 0.seviye kabul edilmelidir. (Örnek Cevap: 1,3,5,7) (4 puan, tahmini süre: 2 dk.)

**Cevap: 2,6,12,5**

3. Aşağıdaki her işlem için kullanılabilecek **en uygun** veri yapısını sıra ile yazınız.
- a. Bir derleyicinin fonksiyonlara gidip dönme işlemlerini düzenlemek için
  - b. Bir klavyede basılan tuşları ekrana yazarken
  - c. Dengeli olmayan bir ikili ağacı saklamak için
  - d. Satranç oyununda bir sonraki hamleyi belirlemek için

(4 puan, tahmini süre: 2 dk.)

Cevap : Yığın, Kuyruk, Linkli Liste, Ağaç

4. Aşağıdaki **program parçası çalıştırıldığı anda** kuyrukta **1 2 3 4** sayıları bulunmaktadır (**1** kuyruğun başında, **4** en sonundadır). Stack boştur. **Program parçasının çalışması bittiğinde** kuyrukta bulunan elemanları kuyruk başından itibaren sıra ile yazınız.

.....

```
while( !isEmpty(queue))  
{  
    value = enqueueQueue(queue);  
    pushStack(value,stack);  
}
```

```
while(!isEmpty(stack))  
{  
    value = popStack(stack);  
    enqueueQueue(value,queue);  
}  
....
```

(4 puan, tahmini süre: 1 dk.)

Cevap : 4,3,2,1

5. Bir ikili arama ağacında 5 sayısı aranırken ağaçta bulunan elemanlara hangi sıra ile bakılamaz?
- a. 10, 9, 8, 7, 6, 5
  - b. 1, 10, 2, 9, 3, 5
  - c. 2, 7, 3, 8, 4, 5
  - d. 1, 2, 10, 4, 8, 5
  - e. Hepsi doğru

(4 puan, tahmini süre: 3 dk.)

Cevap : C

6. En küçük elemanın kökte yer aldığı  $N$  elemanlı bir **heap** ağacında en küçük 3 elemanı silmenin karmaşıklığı en kötü durumda ne olur? Big-Oh notasyonu ile gösteriniz. (4 puan, tahmini süre: 1 dk.)

Cevap :  $O(\lg N)$

8. Verilen C kodunun (en kötü durumda) çalışma zamanı karmaşıklığını Big-Oh notasyonunda ifade ediniz.

```
int fonksiyon(int n) {
```

```
    int i, j, k = 0;
```

```
    for (i = n/2; i <= n; i++) {
```

```
        for (j = 2; j <= n; j*=2) {
```

```
            k += n/2;
```

```
        }
```

```
    }
```

```
    return k;
```

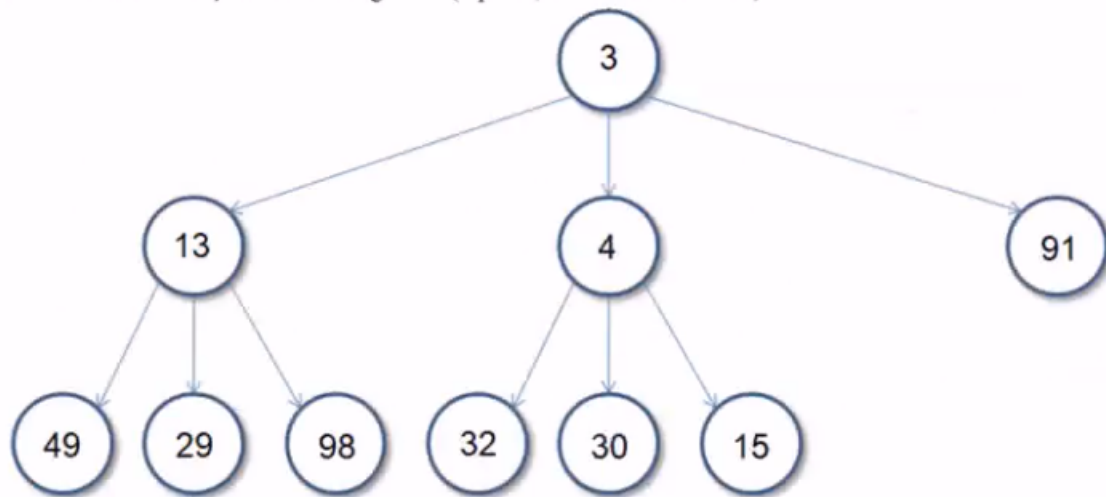
```
}
```

(4 puan, tahmini süre: 2 dk.)

Cevap:  $O(N \log N)$



9. Elimizde üçlü (ternary) min-heap veri yapısı bulunmaktadır (her ebeveyn düğümünün kendinden büyük değere sahip 3 alt düğümü olan tam (complete) üçlü ağaç). Aşağıda verilen min-ternary-heap'te en küçük eleman silindiğinde heap dizisinin son hali ne olur? Değerleri aralarında birer boşluk bırakarak giriniz. (4 puan, tahmini süre: 2 dk.)



Cevap: 4 13 15 91 49 29 98 32 30

10. En soldaki sütun, orijinal veri dizisidir. I, II, III sütunları ise 3 farklı sıralama algoritmasının herhangi bir adımında oluşan dizilerdir. Buna göre, I, II, III sütunlarındaki veriler hangi sıralama algoritması kullanılarak sıralanmaktadır? (4 puan, tahmini süre: 4 dk.)

ORJİNAL DİZİ	ALGORİTMA I	ALGORİTMA II	ALGORİTMA III
that	from	your	been
with	have	with	from
have	know	will	good

this	that	this	have
that	that	they	know
will	that	want	that
will	they	will	that
your	this	that	that
from	will	from	that
they	will	that	they
know	with	know	this
that	your	that	want
want	want	have	will
been	been	been	your

<b>that</b>	that	that	with
<b>good</b>	good	good	will

- a) I: Insertion Sort, II: Selection Sort, III: Heap Sort
- b) I: Selection Sort, II: Insertion Sort, III: Heap Sort
- c) I: Insertion Sort, II: Heap Sort, III: Selection Sort**
- d) I: Heap Sort, II: Insertion Sort, III: Selection Sort
- e) I: Selection Sort, II: Heap Sort, III: Insertion Sort

7. Verilen bir ifadedeki parantezlerin açma-kapama sayılarının dengeli olup olmadığını kontrol etmenin bir yolu, stack veri yapısından yararlanmaktır. Açılan parantezler stack'e push edilir. Kapama parantezi okunduğunda stack'ten bir eleman pop edilerek karşılaştırılır. Eğer eşleşiyorlarsa, işleme devam edilir. Tüm ifade okunup bittiğinde stack boş ise, parantezler dengelidir.

Rasgele bir sıralama ile 4 açılış ve 5 kapanış parantezi içeren bir ifade kontrol için veriliyor. Herhangi bir anda, stack'te bulunabilecek maksimum parantez sayısı kaç olabilir? (4 puan, tahmini süre: 2 dk.)

- a. 1
- b. 3
- c. 4**
- d. 6
- e. 7