**1-4. sorular için:** Aşağıda verilen her bir fonksiyonun çalışma zamanının (run time) karmaşıklığını Big-O cinsinden tahmin ediniz.
**I-** O(nlogn) **II-** O(n) **III-** O(nlogn) **IV-** $O(n^2)$ **V-** $O(n^3)$

**1)** $T(n)=(2n+n-1)^2$
a) I b) II c) III d) IV e) V

**2)** $T(n)=(2n)\log(2n)$
a) I b) II c) III d) IV e) V
iki aynı cevap olduğu için ikiside doğru kabul edilecektir.

**3)** $T(n)=(2n-1)^2(n+1)$
a) I b) II c) III d) IV e) V

**4)** $T(n)=3(2n-1)+1$
a) I b) II c) III d) IV e) V

**5-7. sorular için:** Aşağıda verilen Java ve Python kodlarının hangi algoritmaya ait olduklarını bulunuz.
**I-** Bubblesort , **II-** Binarysearch, **III-** Selectionsort, **IV-** Mergesort, **V-** Insertionsort

**5)** ***Java***

```
void sort(int arr[]) {
  int n = arr.length;
  for (int i = 0; i < n-1; i++)
    for (int j = 0; j < n-i-1; j++)
      if (arr[j] > arr[j+1]) {
        int temp = arr[j];
        arr[j] = arr[j+1];
        arr[j+1] = temp;
      }
  }
```

***Python***

```
def sort(arr):
  n = len(arr)
  for i in range(n):
    for j in range(0, n-i-1):
      if arr[j] > arr[j+1] :
        arr[j],arr[j+1] =arr[j+1],arr[j]
```

a) I b) II c) III d) IV e) V

**6)** ***Java***

```
void sort(int arr[]) {
  int n = arr.length;
  for (int i=1; i<n; ++i) {
    int key = arr[i];
    int j = i-1;
    while (j>=0 && arr[j] > key) {
      arr[j+1] = arr[j];
      j = j-1; }
    arr[j+1] = key;
  }
}
```

***Python***

```
def sort(arr):
  for i in range(1, len(arr)):
    key = arr[i]
    j = i-1
    while j >=0 and key < arr[j] :
        arr[j+1] = arr[j]
```

a) I b) II c) III d) IV e) V

**7)** ***Java***

```
int s7(int arr[], int l, int r, int x){
  if (r>=l) {
    int mid = l + (r - l)/2;
    if (arr[mid] == x)
      return mid;
    if (arr[mid] > x)
      return s7(arr, l, mid-1, x);
    return s7(arr, mid+1, r, x);
  }
  return -1;
}
```

***Python***

```
def s7 (arr, l, r, x):
  if r >= l:
    mid = l + (r - l)/2
    if arr[mid] == x:
      return mid
    elif arr[mid] > x:
      return s7(arr, l, mid-1, x)
    else:
      return s7(arr, mid+1, r, x)
  else:
    return -1
```

a) I b) II c) III d) IV e) V

**8-10. sorular için:** Sırası ile 5-7 arasında verilen algoritmaların Big-O notasyonu cinsinden algoritma karmaşıklığını tahmin ediniz.
**I-** O(logn) **II-** O(n) **III-** O(nlogn) **IV-** $O(n^2)$ **V-** $O(n^3)$

**8)** (5.sorudaki algoritma için Big-O)
a) I b) II c) III d) IV e) V

**9)** (6.sorudaki algoritma için Big-O)
a) I b) II c) III d) IV e) V

**10)** (7.sorudaki algoritma için Big-O)
a) I b) II c) III d) IV e) V

**11-13.sorular için:** Bir LIFO yapısı için sırası ile aşağıdaki komutlar verilmiştir. (Soruları birbirinden bağımsız olarak düşünün)
push(1), push(3),pop(), push(5), top(), push(7), push(3), top()

**I-** 2, **II-** 3, **III-** 4, **IV-** 5, **V-** 7

**11)** Bu LIFO yapısında pop() komutu çıktısı ne olur?
a) I b) II c) III d) IV e) V

**12)** Bu LIFO yapısında size() komutu çıktısı ne olur?
a) I b) II c) III d) IV e) V

**13)** Bu LIFO yapısında sırası ile top(), size() komutu çıktısı ne olur?
a) I b) II c) III d) IV e) V

**14-15.sorular için:** Bir FIFO yapısı için sırası ile aşağıdaki komutlar verilmiştir. (Soruları birbirinden bağımsız olarak düşünün)
insert(1), insert(2),insert(3), remove(), front(), insert(4), insert(5)

**I-** 1, **II-** 2, **III-** 3, **IV-** 4, **V-** 5

**14)** Bu FIFO yapısında sırası ile remove() komutu çıktısı ne olur?
a) I b) II c) III d) IV e) V

**15)** Bu FIFO yapısında sırası ile front(), size() komutu çıktısı ne olur?
a) I b) II c) III d) IV e) V

*(Bu alanı karalama olarak kullanabilirsiniz)*

**Notlar**
(1) 1-17 sorular her biri 4 puandır.
(2) Test soruları cevap anahtarına, 18-19 sorular ayrılan yere cevaplanacaktır.
(3) 18-19.sorular için şu diller serbesttir. (Java/Python/Ruby/C)
(4) Cevaplarınız **okunaklı** ve gerekli açıklamaları içermelidir.
(5) Ön yüzde ve cevap anahtarı kısmında isim/numara yazmayı unutmayınız.
(6) Sınav süresi 75 dk. Sınav sonunda soru kağıdı teslim edilecektir.
(7) Harici bir cevap kağıdı kullanılmayacaktır.

**16)** Listeler için verilen ifadelerle ilgili doğru olan seçeneği işaretleyiniz.
a) Tek yönlü bağlı listede geriye doğru hareket mümkündür.
b) Çift yönlü bağlı listelerde iki yönlü hareket mümkündür.
c) Tek yönlü dairesel bağlı listede iki yönlü hareket mümkündür.
d) Çift yönlü dairesel bağlı listede her ***node*** sadece bir adres tutar.
e) Bağlı listelerde her ***node*** farklı iki ***node*** adresini tutar.

**17)** Her düğüm kendinden önceki ve sonraki düğümün adresini tutuyor, ilk düğümde *head*, son düğümde *tail* bilgisi var ise bu liste türü hangisidir?
a) Doğrusal liste (dizi)
b) Tek yönlü bağlı liste
c) Çift yönlü bağlı liste
d) Tek yönlü dairesel bağlı liste
e) Çift yönlü dairesel bağlı liste

**18)** Quicksort algoritmasını rekürsif olarak kodlayınız. (**12p**)

**19)** Paket programların çalışmasını simüle eden bir programda aşağıdaki işlemler yapılmak istenmektedir.
**I-** Her işlem sırası gelince çalışacaktır. Sırası geldiğinde tamamen çalışıp görevi bitecektir.
**II-** İlk maddede bahsedilen görevi biten işlem geri-al, yineleme amacı ile bir başka veri yapısına eklenecektir. Bu maddede ki veri yapısında ekleme/çekme işlemleri en tepeden yapılacaktır.

a) I için önerdiğiniz veri yapısı nedir, gerekçesi ile yazınız. (**4p**)
b) II için önerdiğin veri yapısı nedir, gerekçesi ile yazınız. (**4p**)
c) İlk iki maddenin çalışmasını simüle edecek *nesne tabanlı* programı yazınız. I için uygun ekleme/çıkarma işlemi metot olarak tanımlanmalıdır, II için sadece ekleme işlemi metot olarak tanımlanmalıdır ve I için yazdığını metot içinden uygun şekilde çağrılmalıdır. (**12p**)

---

**Cevaplar**

**Cevap-18) Python kodu**

```
def quicksort(list):
   boyut = len(list)
   if(boyut <= 1):
      return list
   else:
      pivot = list[0]
      buyuk = [ i for i in list[1:] if i > pivot ]
      kucuk = [ i for i in list[1:] if i <= pivot ]
      return quicksort(kucuk) + [pivot] + quicksort(buyuk)
```

**Cevap-19 a)**

FIFO yapısıdır. İlk giren ilk çıkacak, Bu FIFO'dan çıkan görevler II'de belirtilen LIFO'ya girecek.

**Cevap-19 b)**

LIFO yapısıdır. Bu LIFO'ya I'de belirtilen FIFO'dan çıkanlar eklenecek.

**Cevap-19 c)**

```
class s19():
   def __init__(self, limit):
      self.FIFO = []    # I için
      self.length = 0
      self.LIFO = []    # II için
      self.limit = limit

   def insert(self, item):
      self.FIFO.append(item)
      self.length = self.length + 1

   def remove(self):
      if self.length>0:
         item=self.entries[0]
         self.length = self.length - 1
         del self.FIFO[0]
         self.push(item).  # II için verilen LIFO burdan çağrılacak
         return item
      else:
         print("Kuyruk boş")

   def push(self, data):
      if len(self.stack) >= self.limit:
         raise StackOverflowError
      self.LIFO.append(data)
```

I- de istenen yapı FIFO'dur. S19 yapılandırıcı metot içinde FIFO ve LIFO tanımlanmıştır. *Insert* metodu FIFO'ya ekleme yapar, *remove* metodu FIFO'dan eleman çeker (silerek) ve bu elemanı II-de istenen LIFO'ya ekler. LIFO'ya ekleme metodu *push* olarak tanımlanmıştır ve soruda istendiği gibi *remove* metodu içinden çağrılmaktadır.

| | |
|---|---|
| **İsim:** | |
| **Numara:** | |
| **Puan:** | |

CEVAP FORMU

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | A | B | C | D | E | 11 | A | B | C | D | E |
| 2 | A | B | C | D | E | 12 | A | B | C | D | E |
| 3 | A | B | C | D | E | 13 | A | B | C | D | E |
| 4 | A | B | C | D | E | 14 | A | B | C | D | E |
| 5 | A | B | C | D | E | 15 | A | B | C | D | E |
| 6 | A | B | C | D | E | 16 | A | B | C | D | E |
| 7 | A | B | C | D | E | 17 | A | B | C | D | E |
| 8 | A | B | C | D | E | 18 | A | B | C | D | E |
| 9 | A | B | C | D | E | 19 | A | B | C | D | E |
| 10 | A | B | C | D | E | 20 | A | B | C | D | E |

*(Bu alanı karalama olarak kullanabilirsiniz)*