

יסודות האלגוריתמים והסיבוכיות

תרגול 03 - רקורסיה ומיון מיזוג



אוניברסיטת בן-גוריון בנגב
جامعة بن غوريون في النقب
Ben-Gurion University of the Negev

אלגוריתם רקורסיבי

אלגוריתם רקורסיבי הוא אלגוריתם הקורא לעצמו במהלך ריצתו, רקורסיה יעילה בפתירת בעיות המתחלקות בקלות לתתי בעיות דומות.



אלגוריתם רקורסיבי הוא אלגוריתם הקורא לעצמו במהלך ריצתו, רקורסיה יעילה בפתירת בעיות המתחלקות בקלות לתתי בעיות דומות.

ניתן לנתח את זמן הריצה של אלגוריתם רקורסיבי על פי הנוסחה הבאה:

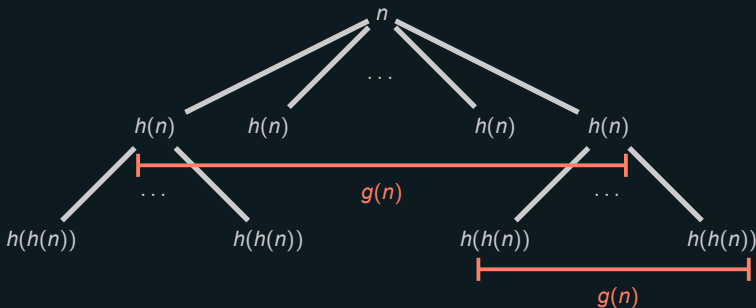
$$T(n) = g(n) \cdot T(h(n)) + f(n)$$



אלגוריתם רקורסיבי הוא אלגוריתם הקורא לעצמו במהלך ריצתו, רקורסיה יעילה בפתירת בעיות המתחלקות בקלות לתתי בעיות דומות.

ניתן לנתח את זמן הריצה של אלגוריתם רקורסיבי על פי הנוסחה הבאה:

$$T(n) = g(n) \cdot T(h(n)) + f(n)$$



איור 1: המחשה של מהלך הרקורסיה של אלגוריתם רקורסיבי, כל קריאה עם קלט בגודל n מייצרת עד $\mathcal{O}(g(n))$ קריאות נוספות עם קלט בגודל $h(n)$.



נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```



נתון המערך [16, 49, 39, 27, 43, 34, 46, 40]
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```

[16, 49, 39, 27, 43, 34, 46, 40]



נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```

$[16, 49, 39, 27, 43, 34, 46, 40]$

$[16, 49, 39, 27]$



נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```

$[16, 49, 39, 27, 43, 34, 46, 40]$

$[16, 49, 39, 27]$

$[16, 49]$



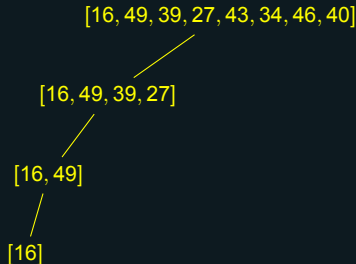
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





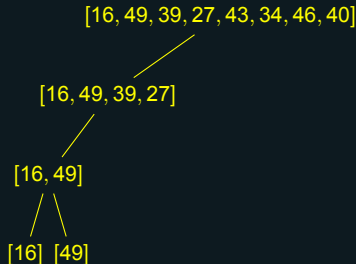
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```





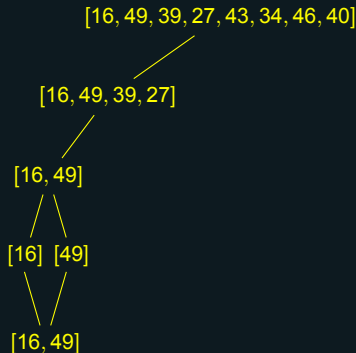
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```





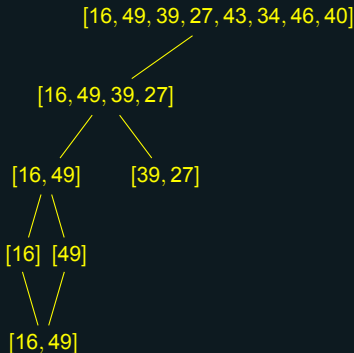
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```





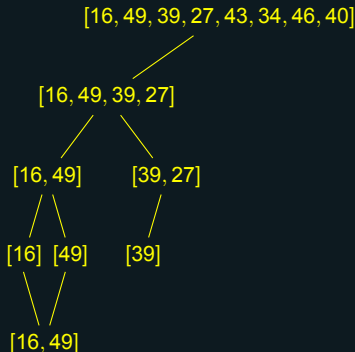
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```





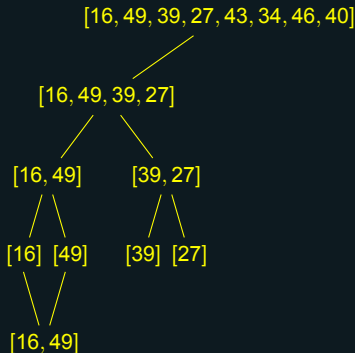
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```





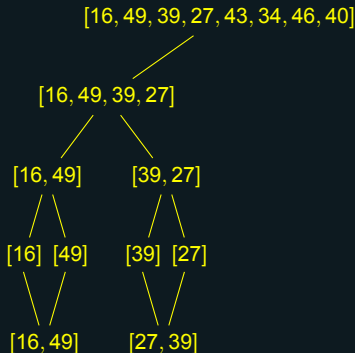
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```





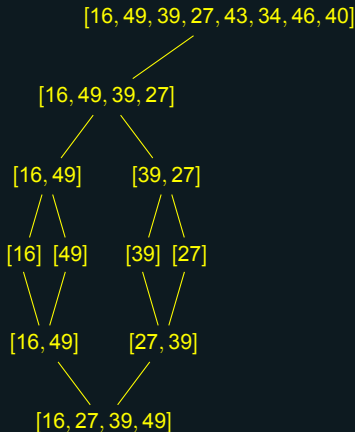
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





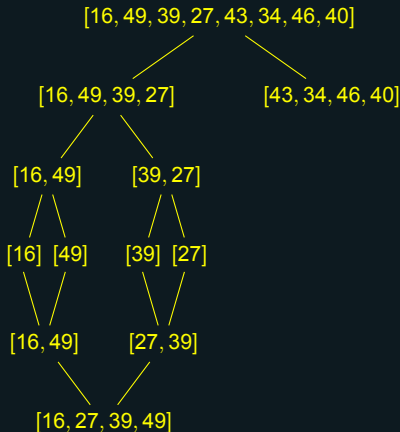
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





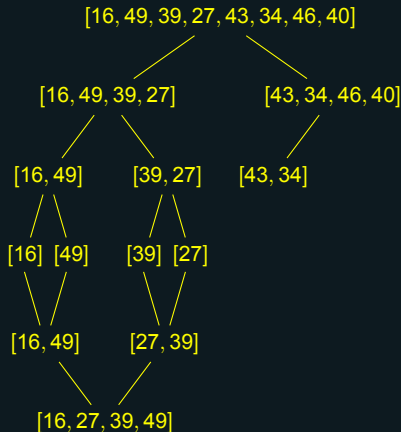
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





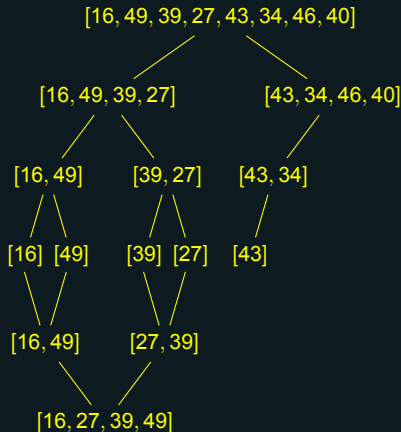
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```





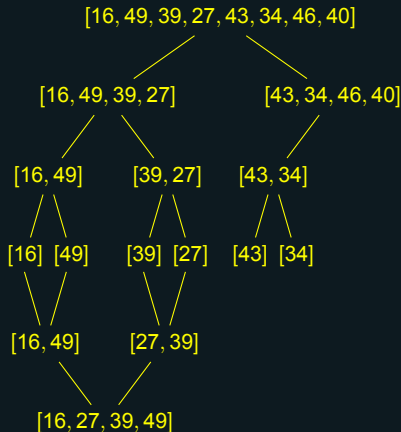
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





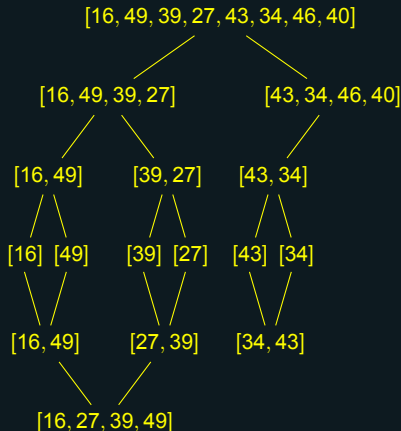
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return A
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return B
```





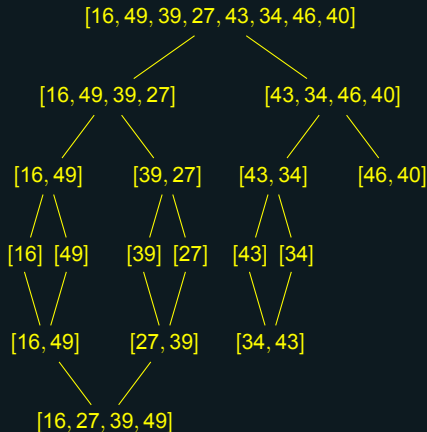
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





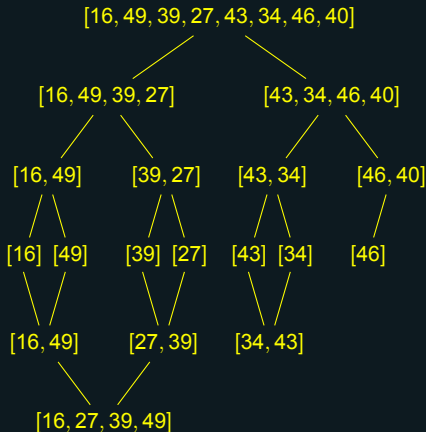
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





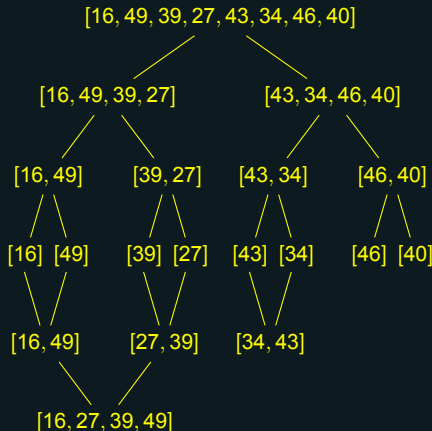
נתון המערך [16, 49, 39, 27, 43, 34, 46, 40]
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





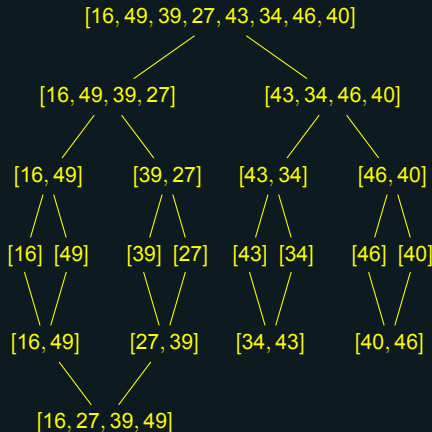
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





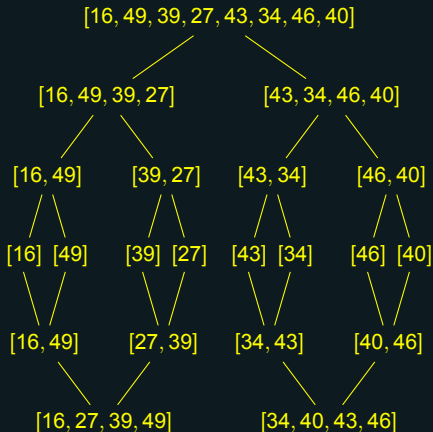
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





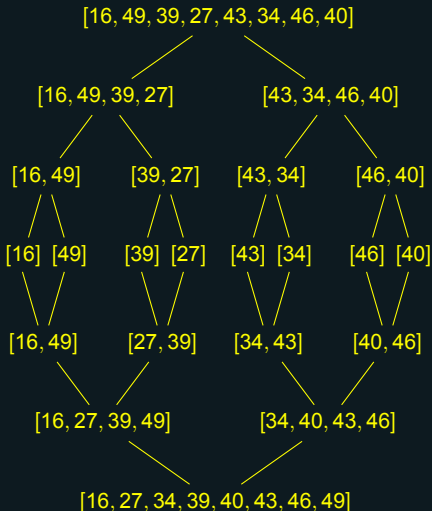
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





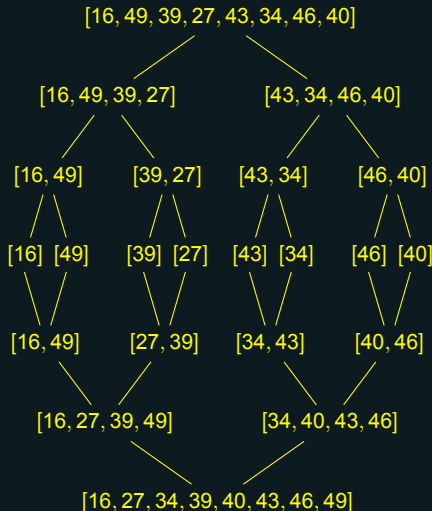
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





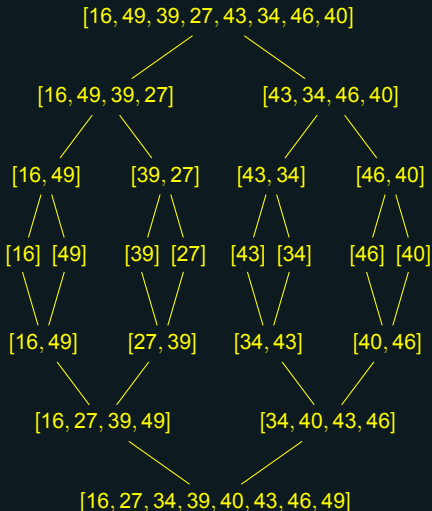
נתון המערך $[16, 49, 39, 27, 43, 34, 46, 40]$
פרטו את שלבי מיון המיזוג.

MergeSort($A[1, \dots, n]$)

```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```
1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else
7:      $B[i] = B_2[i_2]$ 
8:      $i_2 = i_2 + 1$ .
9:    $i = i + 1$ .
10: if  $i_1 > n/2$  then
11:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
12: if  $i_2 > n/2$  then
13:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
return  $B$ 
```





כעת נתמקד בשלב המיזוג, נשנה את אלגוריתם מיון מיזוג באופן הבא:

בכל שלב, במקום לחלק את המערך ל 2 תתי מערכים, נחלק את המערך ל k תתי מערכים בגודל זהה. כלומר כל תת מערך בגודל $\frac{n}{k}$ ($k > 1$ שלם). לאחר החלוקה נפעיל את האלגוריתם באופן רקורסיבי על כל אחד מתתי המערכים. אנו יודעים כי ניתן למיין בזמן קבוע מערך שגודלו חסום בקבוע. אם כך נותר לנו למזג את k תתי המערכים הממויינים.

תארו בקצרה איך למזג k תתי מערכים ממויינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .



תארו בקצרה איך למזג k תתי מערכים ממויינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .



תארו בקצרה איך למזג k תתי מערכים ממויינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. ניצור מערך עזר בגודל k .



תארו בקצרה איך למזג k תתי מערכים ממויינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. ניצור מערך עזר בגודל k .
2. ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינמלי במערך i .



תארו בקצרה איך למזג k תתי מערכים ממויינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. ניצור מערך עזר בגודל k .
2. ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינמלי במערך i .
3. נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.



תארו בקצרה איך למזג k תתי מערכים ממיינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. ניצור מערך עזר בגודל k .
2. ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינמלי במערך i .
3. נעתיק את k האיברים עליהם מצביעים המצבעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. ניצור מערך עזר בגודל k .
2. ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
א. נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. ניצור מערך עזר בגודל k .
2. ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. נקדם את המצביע שהצביע עליו לאיבר הבא במערך.



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. ניצור מערך עזר בגודל k .
2. ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. נקדם את המצביע שהצביע עליו לאיבר הבא במערך.
 - ג. נעתיק את האיבר החדש במקום האיבר המינימלי.



תארו בקצרה איך למזג k תתי מערכים ממויינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. $O(k)$ ניצור מערך עזר בגודל k .
2. ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. נקדם את המצביע שהצביע עליו לאיבר הבא במערך.
 - ג. נעתיק את האיבר החדש במקום האיבר המינימלי.



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. $O(k)$ ניצור מערך עזר בגודל k .
2. $O(k)$ ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. נקדם את המצביע שהצביע עליו לאיבר הבא במערך.
 - ג. נעתיק את האיבר החדש במקום האיבר המינימלי.



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. $O(k)$ ניצור מערך עזר בגודל k .
2. $O(k)$ ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. $O(k)$ נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. נקדם את המצביע שהצביע עליו לאיבר הבא במערך.
 - ג. נעתיק את האיבר החדש במקום האיבר המינימלי.



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. $O(k)$ ניצור מערך עזר בגודל k .
2. $O(k)$ ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. $O(k)$ נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. $O(k)$ נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. נקדם את המצביע שהצביע עליו לאיבר הבא במערך.
 - ג. נעתיק את האיבר החדש במקום האיבר המינימלי.



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. $O(k)$ ניצור מערך עזר בגודל k .
2. $O(k)$ ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. $O(k)$ נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. $O(k)$ נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. $O(1)$ נקדם את המצביע שהצביע עליו לאיבר הבא במערך.
 - ג. נעתיק את האיבר החדש במקום האיבר המינימלי.



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. $O(k)$ ניצור מערך עזר בגודל k .
2. $O(k)$ ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. $O(k)$ נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. $O(k)$ נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. $O(1)$ נקדם את המצביע שהצביע עליו לאיבר הבא במערך.
 - ג. $O(1)$ נעתיק את האיבר החדש במקום האיבר המינימלי.



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. $O(k)$ ניצור מערך עזר בגודל k .
2. $O(k)$ ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. $O(k)$ נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. $O(n)$ כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. $O(k)$ נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. $O(1)$ נקדם את המצביע שהצביע עליו לאיבר הבא במערך.
 - ג. $O(1)$ נעתיק את האיבר החדש במקום האיבר המינימלי.



תארו בקצרה איך למזג k תתי מערכים ממוינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב n ו k .

1. $O(k)$ ניצור מערך עזר בגודל k .
2. $O(k)$ ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע i על האיבר המינימלי במערך i .
3. $O(k)$ נעתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. $O(n)$ כל עוד קיים מצביע שנמצא בתוך אחד המערכים:
 - א. $O(k)$ נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. $O(1)$ נקדם את המצביע שהצביע עליו לאיבר הבא במערך.
 - ג. $O(1)$ נעתיק את האיבר החדש במקום האיבר המינימלי.

זמן הריצה הכולל של האלגוריתם $O(n \cdot k)$



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך m כך ש- m היא חזקה של 3, מפצל את המערך לשלשה מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממוין אחד. מהו זמן הריצה של האלגוריתם הנ"ל? נמקו.



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך m כך ש- m היא חזקה של 3, מפצל את המערך לשלשה מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממיון אחד. מהו זמן הריצה של האלגוריתם הנ"ל? נמקו.

ראשית נבחן כמה רמות יש בעץ שבו מפצלים את המערך ל-3:



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך n כך ש- n היא חזקה של 3, מפצל את המערך לשלושה מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממוין אחד. מהו זמן הריצה של האלגוריתם הנ"ל? נמקו.

ראשית נבחן כמה רמות יש בעץ שבו מפצלים את המערך ל-3: מכיוון שבכל פעם אנו מחלקים את המערך ב-3, בכל שלב יש $\frac{n}{3^k}$ איברים בכל מערך (כאשר k הוא מספר הרמות). התהליך מסתיים כשיש בכל מערך איבר אחד בלבד, נתון זה מסגיר לנו שמספר הרמות הוא $\log_3(n)$ ע"פ:



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך n כך ש- n היא חזקה של 3, מפצל את המערך לשלושה מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממורכז אחד. מהו זמן הריצה של האלגוריתם הנ"ל? נמקו.

ראשית נבחן כמה רמות יש בעץ שבו מפצלים את המערך ל-3: מכיוון שבכל פעם אנו מחלקים את המערך ב-3, בכל שלב יש $\frac{n}{3^k}$ איברים בכל מערך (כאשר k הוא מספר הרמות). התהליך מסתיים כשיש בכל מערך איבר אחד בלבד, נתון זה מסגיר לנו שמספר הרמות הוא $\log_3(n)$ ע"פ:

$$1 = \frac{n}{3^k}$$



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך n כך ש- n היא חזקה של 3, מפצל את המערך לשלשה מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממורכב אחד. מהו זמן הריצה של האלגוריתם הנ"ל? נמקו.

ראשית נבחן כמה רמות יש בעץ שבו מפצלים את המערך ל-3: מכיוון שבכל פעם אנו מחלקים את המערך ב-3, בכל שלב יש $\frac{n}{3^k}$ איברים בכל מערך (כאשר k הוא מספר הרמות). התהליך מסתיים כשיש בכל מערך איבר אחד בלבד, נתון זה מסגיר לנו שמספר הרמות הוא $\log_3(n)$ ע"פ:

$$1 = \frac{n}{3^k} \Leftrightarrow n = 3^k$$



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך n כך ש- n היא חזקה של 3, מפצל את המערך לשלושה מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממורכז אחד. מהו זמן הריצה של האלגוריתם הנ"ל? נמקו.

ראשית נבחן כמה רמות יש בעץ שבו מפצלים את המערך ל-3: מכיוון שבכל פעם אנו מחלקים את המערך ב-3, בכל שלב יש $\frac{n}{3^k}$ איברים בכל מערך (כאשר k הוא מספר הרמות). התהליך מסתיים כשיש בכל מערך איבר אחד בלבד, נתון זה מסגיר לנו שמספר הרמות הוא $\log_3(n)$ ע"פ:

$$1 = \frac{n}{3^k} \Leftrightarrow n = 3^k \Leftrightarrow k = \log_3(n)$$



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך n כך ש- n היא חזקה של 3, מפצל את המערך לשלושה מערכים באורך שווה, ממין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממין אחד. מהו זמן הריצה של האלגוריתם הנ"ל? נמקו.

ראשית נבחן כמה רמות יש בעץ שבו מפצלים את המערך ל-3: מכיוון שבכל פעם אנו מחלקים את המערך ב-3, בכל שלב יש $\frac{n}{3^k}$ איברים בכל מערך (כאשר k הוא מספר הרמות). התהליך מסתיים כשיש בכל מערך איבר אחד בלבד, נתון זה מסגיר לנו שמספר הרמות הוא $\log_3(n)$ ע"פ:

$$1 = \frac{n}{3^k} \Leftrightarrow n = 3^k \Leftrightarrow k = \log_3(n)$$

לפי שאלה 2 מיזוג של k מערכים בגודל $\frac{n}{k}$ רץ בזמן $\mathcal{O}(3n) = \mathcal{O}(n)$ עבור $k = 3$. מכאן שעלות סך פעולות המיזוג בכל רמה הוא $\mathcal{O}(n)$ (ניתוח גלובלי - בכל רמה i ממזגים k^i מערכים בגודל $\frac{n}{k^i}$). הראנו שישנן $\mathcal{O}(\log_3(n))$ רמות בעץ ולכן סה"כ סיבוכיות זמן ריצת האלגוריתם יהיה $\mathcal{O}(n \cdot \log_3(n))$.



נתונה קבוצה של n שמות משפחה. בנוסף, נתון אלגוריתם בשם LYXO היודע להשוות שתי מילים ולהחזיר ב- $\mathcal{O}(\lg(n))$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשמו אלגוריתם המשתמש באלגוריתם הנ"ל על מנת למיין את כל המילים הנתונות בזמן $\mathcal{O}(n \lg^2(n))$.



נתונה קבוצה של n שמות משפחה. בנוסף, נתון אלגוריתם בשם LYXO היודע להשוות שתי מילים ולהחזיר ב- $\mathcal{O}(\lg(n))$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשמו אלגוריתם המשתמש באלגוריתם הנ"ל על מנת למיין את כל המילים הנתונות בזמן $\mathcal{O}(n \lg^2(n))$.

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

MergeSort($A[1, \dots, n]$)

```

1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).

```

```

1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else  $B[i] = B_2[i_2]$  and  $i_2 = i_2 + 1$ .
7:    $i = i + 1$ .
8: if  $i_1 > n/2$  then
9:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
10: if  $i_2 > n/2$  then
11:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
    return  $B$ 

```



נתונה קבוצה של n שמות משפחה. בנוסף, נתון אלגוריתם בשם LYXO היודע להשוות שתי מילים ולהחזיר ב- $\mathcal{O}(\lg(n))$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשמו אלגוריתם המשתמש באלגוריתם הנ"ל על מנת למיין את כל המילים הנתונות בזמן $\mathcal{O}(n \lg^2(n))$.

Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

MergeSort($A[1, \dots, n]$)

```

1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).

```

```

1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $B_1[i_1] \leq B_2[i_2]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else  $B[i] = B_2[i_2]$  and  $i_2 = i_2 + 1$ .
7:    $i = i + 1$ .
8: if  $i_1 > n/2$  then
9:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
10: if  $i_2 > n/2$  then
11:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
    return  $B$ 

```



נתונה קבוצה של n שמות משפחה. בנוסף, נתון אלגוריתם בשם LYXO היודע להשוות שתי מילים ולהחזיר ב- $\mathcal{O}(\lg(n))$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשמו אלגוריתם המשתמש באלגוריתם הנ"ל על מנת למיין את כל המילים הנתונות בזמן $\mathcal{O}(n \lg^2(n))$.

LYXOMerge($B_1[1, \dots, n], B_2[1, \dots, n]$)

MergeSort($A[1, \dots, n]$)

```

1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).

```

```

1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if LYXO( $B_1[i_1], B_2[i_2]$ ) =  $B_1[i_1]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else  $B[i] = B_2[i_2]$  and  $i_2 = i_2 + 1$ .
7:    $i = i + 1$ .
8: if  $i_1 > n/2$  then
9:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
10: if  $i_2 > n/2$  then
11:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
    return  $B$ 

```



נתונה קבוצה של n שמות משפחה. בנוסף, נתון אלגוריתם בשם LYXO היודע להשוות שתי מילים ולהחזיר ב- $\mathcal{O}(\lg(n))$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשמו אלגוריתם המשתמש באלגוריתם הנ"ל על מנת למיין את כל המילים הנתונות בזמן $\mathcal{O}(n \lg^2(n))$.

LYXOMerge($B_1[1, \dots, n], B_2[1, \dots, n]$)

MergeSort($A[1, \dots, n]$)

```

1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

```

1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if LYXO( $B_1[i_1], B_2[i_2]$ ) =  $B_1[i_1]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else  $B[i] = B_2[i_2]$  and  $i_2 = i_2 + 1$ .
7:    $i = i + 1$ .
8: if  $i_1 > n/2$  then
9:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
10: if  $i_2 > n/2$  then
11:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
    return  $B$ 
```

שורה 3 של LYXOMerge מתבצעת $\mathcal{O}(n)$ פעמים
בקריאה אחת.



נתונה קבוצה של n שמות משפחה. בנוסף, נתון אלגוריתם בשם LYXO היודע להשוות שתי מילים ולהחזיר ב- $\mathcal{O}(\lg(n))$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשמו אלגוריתם המשתמש באלגוריתם הנ"ל על מנת למיין את כל המילים הנתונות בזמן $\mathcal{O}(n \lg^2(n))$.

LYXOMerge($B_1[1, \dots, n], B_2[1, \dots, n]$)

MergeSort($A[1, \dots, n]$)

```

1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
```

```

1:  $i = i_1 = i_2 = 1$ .
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if LYXO( $B_1[i_1], B_2[i_2]$ ) =  $B_1[i_1]$  then
4:      $B[i] = B_1[i_1]$ .
5:      $i_1 = i_1 + 1$ .
6:   else  $B[i] = B_2[i_2]$  and  $i_2 = i_2 + 1$ .
7:    $i = i + 1$ .
8: if  $i_1 > n/2$  then
9:    $B[i, \dots, n] = B_2[i_2, \dots, n/2]$ .
10: if  $i_2 > n/2$  then
11:    $B[i, \dots, n] = B_1[i_1, \dots, n/2]$ .
    return  $B$ 
```

שורה 3 של LYXOMerge מתבצעת $\mathcal{O}(n)$ פעמים
בקריאה אחת.

בכל רמה הסיבוכיות של המיזוג היא $\mathcal{O}(n \cdot \lg(n))$ (השוואה של שתי מילים לוקחת $\lg n$ ובכל רמה זה קורה n פעמים). יש $\mathcal{O}(\lg n)$ רמות.

הסיבוכיות של כל האלגוריתם היא $\mathcal{O}(\frac{n}{2} \lg n) + \mathcal{O}(n \lg n) \cdot \mathcal{O}(\lg n) = \mathcal{O}(n \lg^2 n)$