

יסודות האלגוריתמים והסיבוכיות

תרגיל 03 - רקורסיה ומילון מיזוג



אלגוריתם רקורסיבי הוא אלגוריתם הקורא לעצמו במהלך ריצתו, רקורסיה עיליה בפתרית בעיות המתחולקות בקלות לתתי בעיות דומות.



אלגוריתם רקורסיבי הוא אלגוריתם הקורא לעצמו במהלך ריצתו, רקורסיה עיליה בפתרית בעיות המתחקלקות בקלות לתתי בעיות דומות.

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

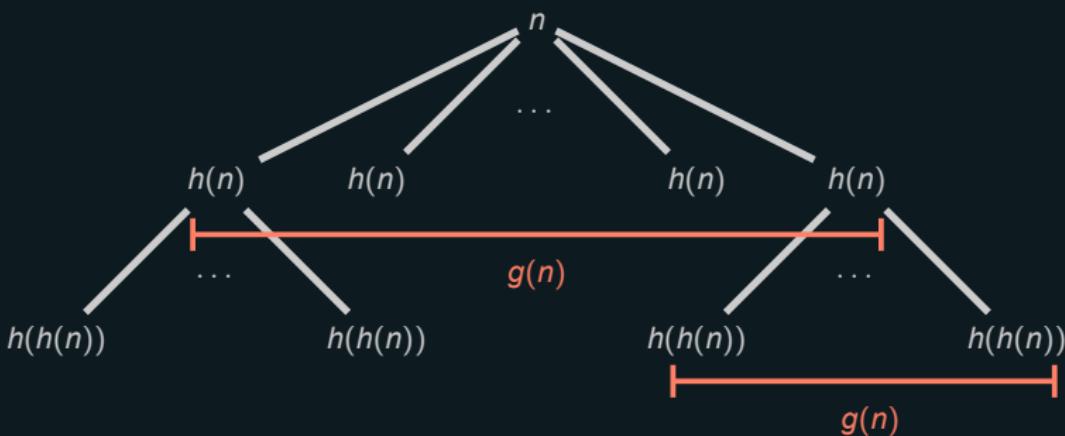
$$T(n) = g(n) \cdot T(h(n)) + f(n)$$



אלגוריתם רקורסיבי הוא אלגוריתם הקורא לעצמו במהלך ריצתו, רקורסיה עיליה בפתרית בעיות המתחקלות 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: המכחשה של מהלך הרקורסיה של אלגוריתם רקורסיבי, כל קריאה עם קלט בגודל h מייצרת עד $\mathcal{O}(g(n))$ קריאות נוספות עם קלט בגודל $(n) \cdot h$.



נתן 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  $B$ 
```



נתן 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( $B_1, B_2$ ).
```

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

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  $B$ 
```



נתן 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( $B_1, B_2$ ).
```

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

[16, 49, 39, 27]

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  $B$ 
```



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  $B$ .

---


```

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

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

[16, 49, 39, 27]

[16, 49]



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  $B$ .

---


```

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

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

[16, 49, 39, 27]

[16, 49]

[16]



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  $B$ .

---


```

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

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

[16, 49, 39, 27]

[16, 49]

[16] [49]



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  $B$ 
```

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

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

[16, 49, 39, 27]

[16, 49]

[16] [49]

[16, 49]



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  $B$ .

---


```

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

פרטו את שלבי מיון המיזוג.

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

[16, 49, 39, 27]

[16, 49]

[39, 27]

[16] [49]

[16, 49]



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  $B$ 
```

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

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

[16, 49, 39, 27]

[16, 49] [39, 27]

[16] [49] [39]

[16, 49]



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  $B$ .

---


```

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

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

[16, 49, 39, 27]

[16, 49] [39, 27]

[16] [49] [39] [27]

[16, 49]



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  $B$ .

---


```

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

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

[16, 49, 39, 27]

[16, 49] [39, 27]

[16] [49] [39] [27]

[16, 49] [27, 39]



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  $B$ .

---


```

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

פרטו את שלבי מיון המיזוג.

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

[16, 49, 39, 27]

[16, 49] [39, 27]

[16] [49] [39] [27]

[16, 49] [27, 39]

[16, 27, 39, 49]



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  $B$ 
```

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

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

[16, 49, 39, 27]

[43, 34, 46, 40]

[16, 49] [39, 27]

[16] [49] [39] [27]

[16, 49] [27, 39]

[16, 27, 39, 49]



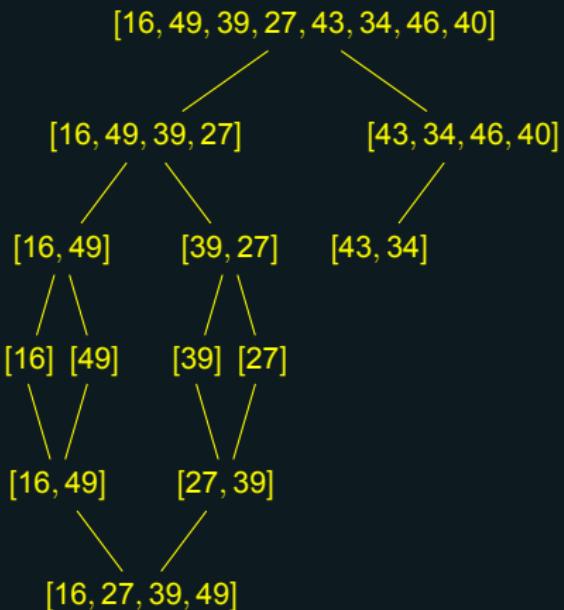
MergeSort($A[1, \dots, n]$)

```
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  $B$ 
```

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





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

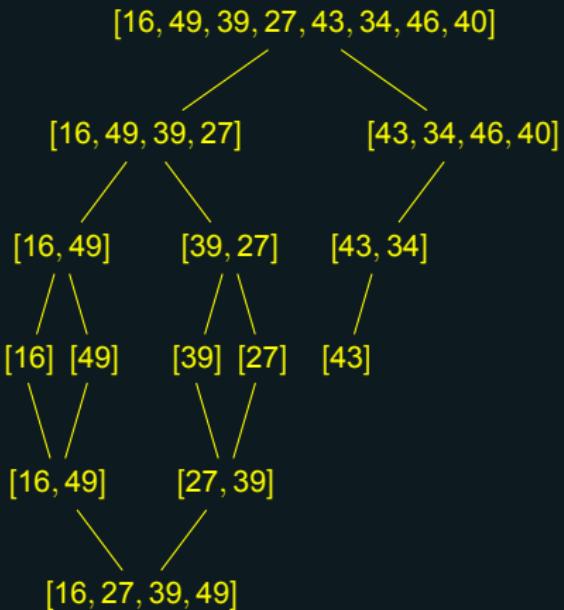
```
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2]).$ 
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n]).$ 
5: return Merge( $B_1, B_2$ ).
```

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  $B$ 
```

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

פרטו את שלבי מיון המיזוג.





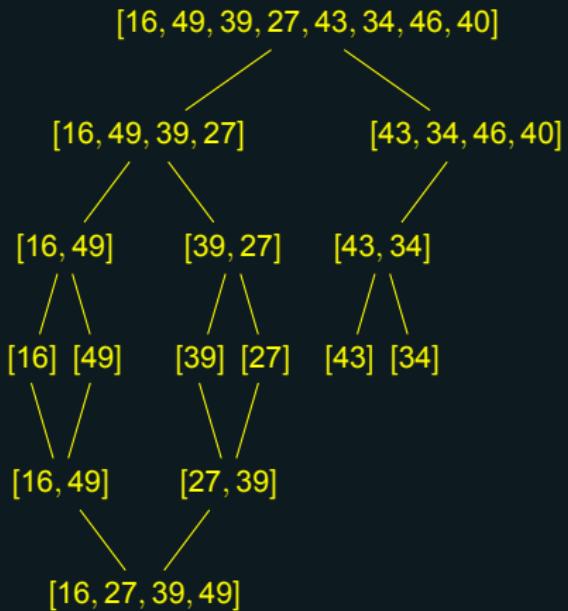
MergeSort($A[1, \dots, n]$)

```
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  $B$ 
```

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





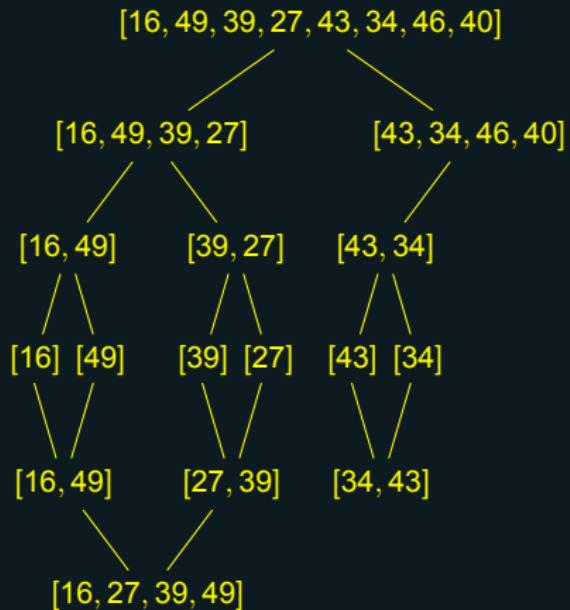
MergeSort($A[1, \dots, n]$)

```
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  $B$ 
```

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





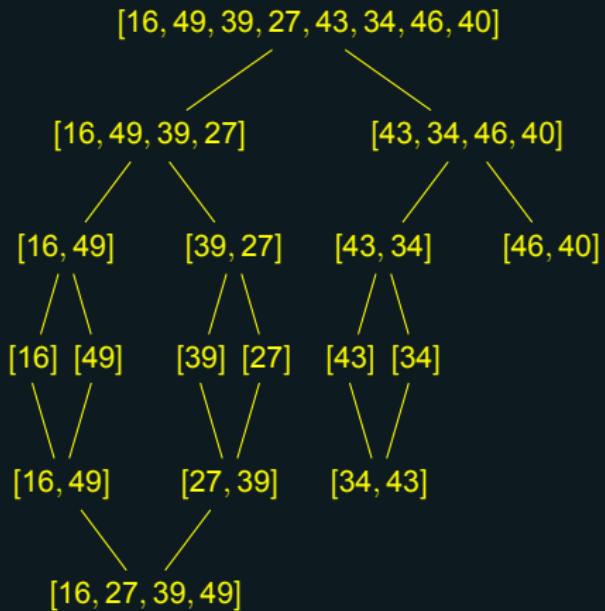
MergeSort($A[1, \dots, n]$)

```
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  $B$ 
```

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





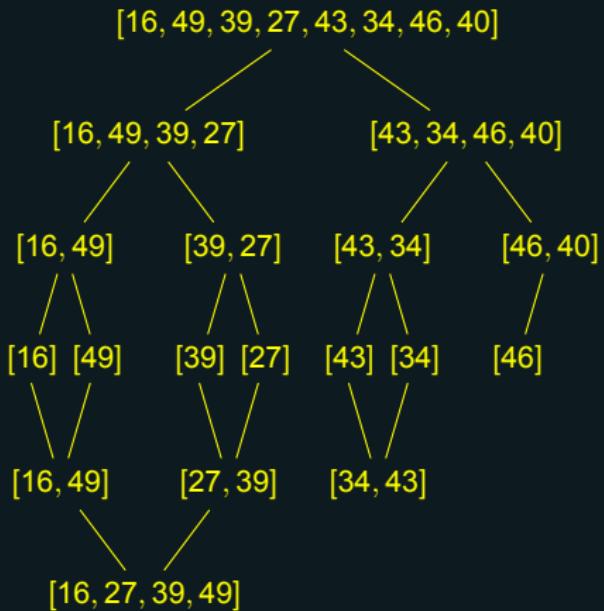
MergeSort($A[1, \dots, n]$)

```
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  $B$ 
```

נתן 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( $B_1, B_2$ ).

```

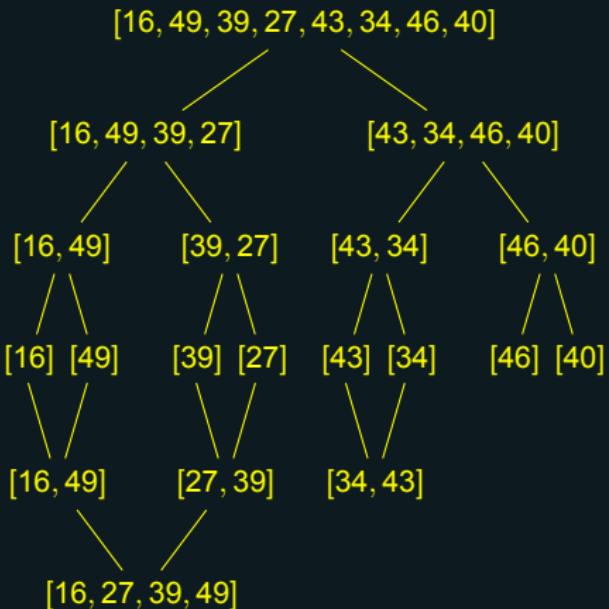
Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```

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  $B$ 

```

נתן 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( $B_1, B_2$ ).

```

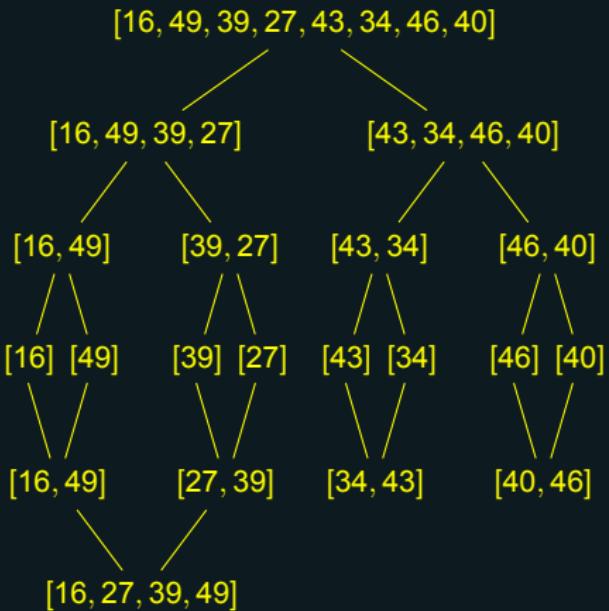
Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```

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  $B$ 

```

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



תרגיל 1



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

```

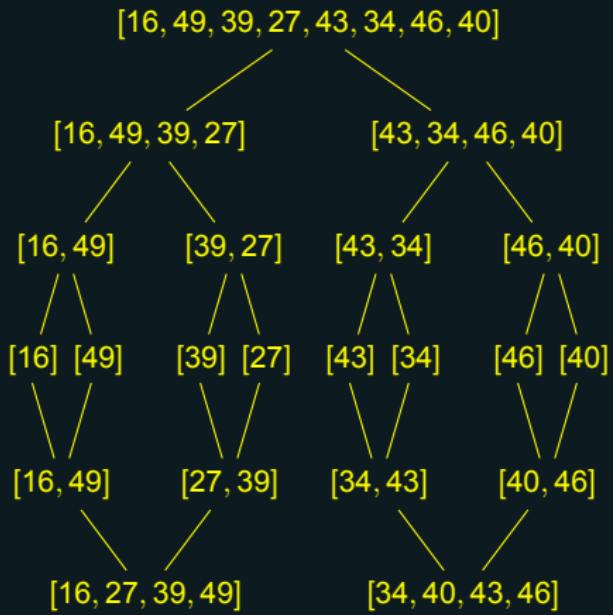
1: if  $n = 1$  then
2:   return  $A$ 
3:  $B_1 = \text{MergeSort}(A[1, \dots, n/2])$ .
4:  $B_2 = \text{MergeSort}(A[n/2 + 1, \dots, n])$ .
5: return Merge( $B_1, B_2$ ).
  
```

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  $B$ 
  
```

נתן 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  $B$ 

```

נתן 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( $B_1, B_2$ ).

```

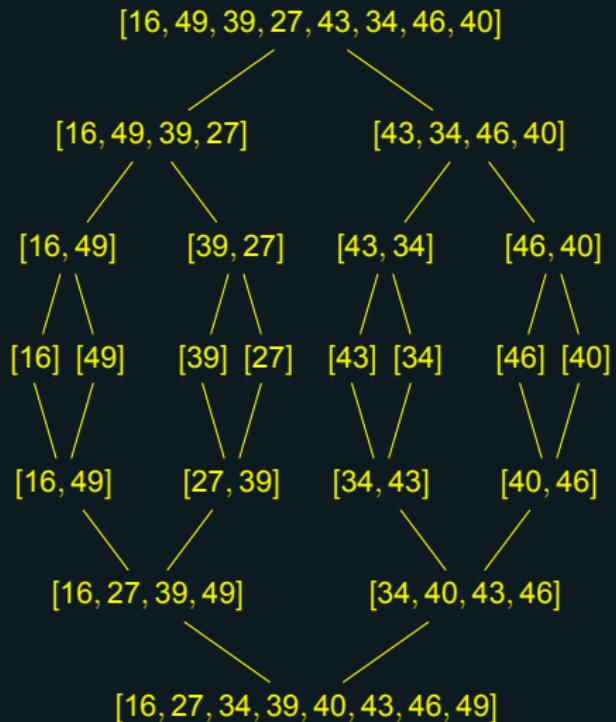
Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```

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  $B$ 

```

נתן 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( $B_1, B_2$ ).

```

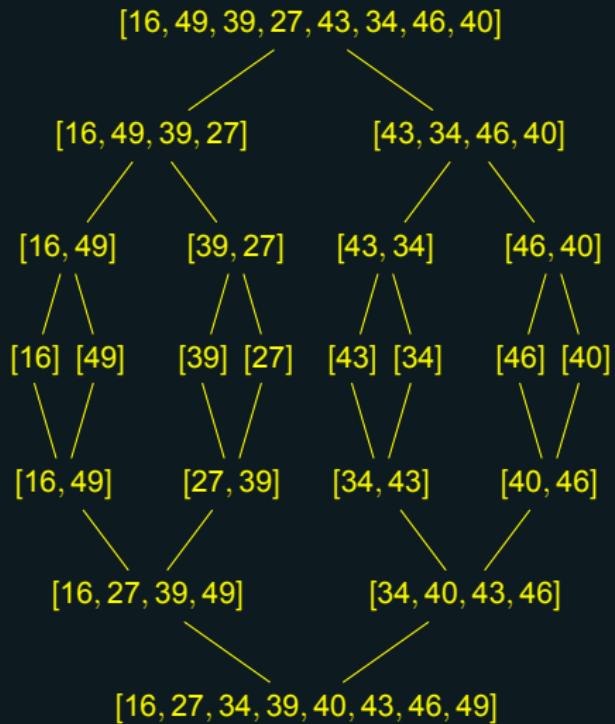
Merge($B_1[1, \dots, n], B_2[1, \dots, n]$)

```

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  $B$ 

```

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





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

בכל שלב, במקום לחלק את המערך ל 2 תת-מערכות, נחלק את המערך ל k תת-מערכות בגודל זהה. כלומר כל תת-מערך בגודל $\frac{n}{k}$ ($1 < k \leq n$). לאחר החלוקה נפעיל את האלגוריתם באופן רקורסיבי על כל אחד מהתת-המערכות. אנו יודעים כי ניתן למין בזמן קבוע שגדלו חסום בקבוע. אם כך נותר לנו למיזוג את k תת-המערכות הממיינניים.

תארו בקצרה איך למיזוג k תת-מערכות ממויינים לטור מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה בחז k .

תרגיל 2



תארו בקצרה איך למzag k תת-מערכות ממויינם לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התחילה ב $\Theta(1)$.

תרגיל 2



תארו בקצרה איך למzag k תת-מערכות ממוחנים לתוך מערך ממוחן אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התחוליה ב n ו- k .

1. ניצור מערך עזר בגודל k .



תארו בקצרה איך למזג k תת-מערכות ממויינים为了让ן מערכ ממוין אחד בגודל ch . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התחילה ב $ch + k$.

1. ניצור מערך עזר בגודל k .
2. ניצור k מצביעים, אחד עבור כל מערך. נקבע את המצביע / על האיבר המינימלי במערך /.



תארו בקצרה איך למזג k תת-מערכות ממויינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התחילה ב $\Theta(1)$.

1. ניצור מערך עזר בגודל k .
2. ניצור k מ מצבים, אחד עבור כל מערך. נקבע את המצבים / על האיבר המינימלי במערך /.
3. נעתק את k האיברים עליהם מ מצבים המצביעים לתוך מערך העזר.

תרגיל 2



תארו בקצרה איך למזג k תת-מערכות ממויינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התחילה ב $\Theta(1)$.

- .1. ניצור מערך עזר בגודל k .
- .2. ניצור k מצבים, אחד עבור כל מערך. נקבע את המצביע / על האיבר המינימלי במערך /.
- .3. העתיק את k האיברים עליהם מ מצבים המצביעים לתוך מערך העזר.
- .4. כל עוד קיים מצביע שנמצא בתוך אחד המערכים:



תארו בקצרה איך למzag k תת-מערכות ממויינים לתוך מערך ממויין אחד בגודל α . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התחילה ב α ו- k .

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



תארו בקצרה איך למzag k תתי מערכות ממויינם לתוך מערך ממויין אחד בגודל ch . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התחילה ב $ch + k$.

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



תארו בקצרה איך למגד 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 / על האיבר המינימלי במערך /.
3. העתיק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיים מצביע שנמצא בתוך אחד המרכיבים:
 - א. נמצא את האיבר המינימלי מתוך מערך העזר ונעתיק אותו למערך הפלט.
 - ב. נקדם את המצביע שהצביע עליי לאיבר הבא במערך.
 - ג. העתיק את האיבר החדש למקום האיבר המינימלי.



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

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

תרגיל 2



תארו בקצרה איך למגד 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. $O(k)$ ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע / על האיבר המינימלי במערך /.
3. העתק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. כל עוד קיימים מצביע שנמצא בתוך אחד המרכיבים:
 - א. נמצא את האיבר המינימלי מתוך מערך העזר ונעתק אותו למערך הפלט.
 - ב. נקדם את המצביע שהצביע עליי לאיבר הבא במערך.
 - ג. העתק את האיבר החדש במקום האיבר המינימלי.



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

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



תארו בקצרה איך למzag k תתי מערכות ממויינם לתוך מערך ממויין אחד בגודל ch . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב ch ו- k .

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



תארו בקצרה איך למzag k תתי מערכות ממויינם לתוך מערך ממויין אחד בגודל ch . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב ch ו- k .

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



תארו בקצרה איך למzag k תתי מערכות ממויינם לתוך מערך ממויין אחד בגודל ch . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב ch ו- k .

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



תארו בקצרה איך למzag k תתי מערכות ממויינם לתוך מערך ממויין אחד בגודל ch . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התלויה ב ch ו- k .

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



תארו בקצרה איך למגד k תתי מערכות ממויינים לתוך מערך ממויין אחד בגודל n . מהו זמן הריצה של האלגוריתם? תנו תשובה כפונקציה התחילה ב $\Theta(1)$.

1. $\Theta(k)$ ניצור מערך עזר בגודל k .
2. $\Theta(k)$ ניצור k מצביעים, אחד עבור כל מערך. נכוון את המצביע / על האיבר המינימלי במערך /.
3. $\Theta(k)$ נעתק את k האיברים עליהם מצביעים המצביעים לתוך מערך העזר.
4. $\Theta(n)$ כל עוד קיים מצביע שנמצא בתוך אחד המרכיבים:
 - א. $\Theta(k)$ נמצא את האיבר המינימלי מתוך מערך העזר ונטוק אותו למערך הפלט.
 - ב. $\Theta(1)$ נקדם את המצביע שהצביע עליי לאיבר הבא במערך.
 - ג. $\Theta(1)$ נעתק את האיבר החדש למקום האיבר המינימלי.

זמן הריצה הכלול של האלגוריתם $(k \cdot n)\Theta$

תרגיל 3



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



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך n כרך ש- h היא חזקה של 3, מפצל את המערך לשולש מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממויין 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 כרך של 3, מפצל את המערך לשולש מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממויין 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). בהתאם למספרם כשים בכל מערך איבר אחד בלבד, נתון זה מסגור לנו שמספר הרמות הוא $(\log_3 n)$.



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך n כרך ש- h היא חזקה של 3, מפצל את המערך לשולש מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממויין 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). בהתאם למסתiem שיש בכל מערך אחד בלבד, נתון זה מסגור לנו שמספר הרמות הוא $(\log_3 n)$.

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



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך n כרך ש- h היא חזקה של 3, מפצל את המערך לשולש מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממויין 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). בהתאם למסתiem שיש בכל מערך אחד בלבד, נתון זה מסגור לנו שמספר הרמות הוא $(\log_3 n)$.

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



להלן אלגוריתם מיון מיזוג חדש. האלגוריתם מקבל מערך באורך n כקלט. הוא חזקתו של א-3, מफצל את המערך לשולש מערכים באורך שווה, ממיין רקורסיבית את שלשת המערכים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המערכים למערך ממויין אחד. מהו זמן הריצה של האלגוריתם הנ"ל? נמכו.

ראשית נבחן כמה רמות יש בעץ שבו מפצלים את המערך לא-3: מכיוון שבכל פעם אותו מחלקים את המערך ב-3, בכל שלב יש $\frac{n}{3^k}$ איברים בכל מערך (כאשר k הוא מספר הרמות). בהתאם למספרם כושיש בכל מערך איבר אחד בלבד, נתון זה מסגיר לנו שמספר הרמות הוא (n) $\log_3(n)$ 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 כקלט. הוא חזקתו של אלגוריתם מיון מיזוג שווה, ממיין רקורסיבית את שלשת המרכיבים, ואז משתמש באלגוריתם של שאלה 2 כדי למזג את שלשת המרכיבים למערך ממויין 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). בהתאם מוסתים שיש בכל מערך איבר אחד בלבד, נתון זה מסגר לנו שמספר הרמות הוא $(\log_3 n)$ לפחות:

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

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



נתונה קבוצה של ch שמות משפחה. בנוסף, נתון אלגוריתם בשם OLYX Δ הידוע להשוות שתי מילימ'ם ולהציג ב- $(n \lg n)$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשםו אלגוריתם המשמש באלגוריתם הנ"ל על מנת למיין את כל המילימ'ם הנתונות בזמן $(n^2 \lg n)$.



נתונה קבוצה של ch שמות משפחה. בנוסף, נתן אלגוריתם בשם OLYXO הידוע להשווות שתי מילימ'ם ולהציג ב- $(n \lg n)$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשמו אלגוריתם המשמש באלגוריתם הנ"ל על מנת למיין את כל המילימ'ם הנתונות בזמן $(n \lg^2 n)$.

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 ch שמות משפחה. בנוסף, נתן אלגוריתם בשם OLYXO הידוע להשווות שתי מילימ'ולחצ'יר ב- $(n \lg n)$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשמו אלגוריתם המשמש באלגוריתם הנ"ל על מנת למיין את כל המילימ'ולחצ'יר הנთונות בזמן $(n \lg^2 n)$.

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 ch שמות משפחה. בנוסף, נתן אלגוריתם בשם LYXO הידוע להשוות שתי מילימ'ולחצ'יר ב- $(n \lg n)$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשםו אלגוריתם המשמש באלגוריתם הנ"ל על מנת למיין את כל המילימ'ולחצ'יר הנთונות בזמן $(n \lg^2 n)$.

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

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 ch שמות משפחה. בנוסף, נתן אלגוריתם בשם LYXO הידוע להשוות שתי מילימולות - $\mathcal{O}(lg(n))$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשםו אלגוריתם המשמש באלגוריתם הנ"ל על מנת למיין את כל המילימולות בזמן $\mathcal{O}(n \cdot lg^2(n))$.

`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  $\text{Merge}(B_1, B_2).$ 

```

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

```

1:  $i = i_1 = i_2 = 1.$ 
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $\text{LYXO}(B_1[i_1], B_2[i_2]) = B_1[i_1]$  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 ch שמות משפחה. בנוסף, נתן אלגוריתם בשם LYXO הידוע להשוות שתי מילימ'ם ולהציג ב- $(n \lg n)$ זמן, מי קודמת ע"פ הסדר הלקסיקוגרפי. רשמו אלגוריתם המשמש באלגוריתם הנ"ל על מנת למיין את כל המילימ'ם הנתונות בזמן $(n \lg^2 n)$.

`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  $\text{Merge}(B_1, B_2).$ 

```

```

1:  $i = i_1 = i_2 = 1.$ 
2: while  $i_1 \leq n/2$  and  $i_2 \leq n/2$  do
3:   if  $\text{LYXO}(B_1[i_1], B_2[i_2]) = B_1[i_1]$  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  $B$ 

```

בכל רמה הסיבוכיות של המיזוג היא $(n \lg n) \cdot n$ (השוואה של שתי מילימ'ם לוקחת ch ובסך הכל רמה זה קורה ch פעמים). יש $(n \lg n)$ רמות.

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