1. לצורך ניתוח הסיבוכיות, נסמן:

K = מספר איברי L2 שקטנים מהאיבר הראשון (הקטן ביותר) של L1.

J = מספר איברי L1 שגדולים מהאיבר האחרון (הגדול ביותר) של L1.

N = מספר איברי L1 בתוספת מספר איברי L2 שאינם גדולים מהאיבר הגדול ביותר בL1.

M יוגדר:

מההגדרה ברור כי: וגם , ולכן .

|  |  |
| --- | --- |
|  | function merge (L1, L2) { |
| 1 | x = head(L1) |
| 1 | y = head(L2) |
| 1 | if (x.key > y.key) { |
|  | L1.head = y |
|  | while (x.key > y.key && y != nil) { |
|  | y = y.next |
|  | } |
|  | x.prev = y.prev |
|  | y.prev.next = x |
|  | } |
|  | while (x != nil && y != nil) { |
|  | if (x.key <= y. key) { |
|  | temp = x |
|  | x = x.next |
|  | } else { |
|  | temp = y.next |
|  | y.next = x.next |
|  | x.next.prev = y |
|  | x.next = y |
|  | y.prev = x |
|  | y = temp |
|  | } |
|  | } |
| 1 | if (y != nil) { |
|  | y.prev = temp |
|  | temp.next = y |
|  | } |
| 1 | return L1 |
|  | } |

נחשב את סיבוכיות הפונקציה:

ניתן להתייחס לביטוי כאילו הוא שווה 0, כי לעומת סדרי גודל של M ושל K הוא ניתן להזנחה.

כמו"כ ניתן לומר כי , מכיון שסוף-סוף מדובר על מספר-סופי-חסום של פעולות שנעשות מקסימום M פעמים, ומתוכן לפחות פעולה אחת מתבצעת בפועל M פעמים (וליתר דיוק: M+1).

נשארנו עם סיבוכיות של שהיא שקולה לכתיבה .

מסקנא: סיבוכיות הפונקציה היא בדיוק .

2.א.

|  |  |
| --- | --- |
|  | function sigma\_list(L1, L2) { |
| 1 | k = 1 |
| 1 | counter1 = head(L1) |
| 1 | counter2 = head(L2) |
| n +1 | while (counter1.next != nil && counter2.next != nil) { |
| n | sum = 0 |
| N | a = head(L1) |
| n | b = head(L2) |
|  | for i = 1 to k - 1 { |
|  | b = b.next |
|  | } |
|  | for i = 1 to k { |
|  | sum = sum + a.key \* b.key |
|  | a = a.next |
|  | b = b.prev |
|  | } |
| n | add(L3, sum) |
| n | k = k + 1 |
|  | } |
| 1 | return L3; |
|  | } |

מסקנא: סיבוכיות הפונקציה היא .

2.ב.

|  |  |
| --- | --- |
| 1 | function sigma\_list(L1, L2) { |
| 1 | k = 1 |
| 1 | counter = head(L1) |
| n +1 | while (counter.next != nil) { |
| n | sum = 0 |
| n | a = head(L1) |
|  | for i = 1 to k { |
|  | b = head(L2) |
|  | for j = 1 to i - 1 { |
|  | b = b.next |
|  | } |
|  | sum = sum + a.key \* b.key |
|  | a = a.next |
|  | } |
| n | add(L3, sum) |
| n | k = k + 1 |
|  | } |
| 1 | return L3; |
|  | } |

מסקנא: סיבוכיות הפונקציה היא .

3. הפונקציה מקבלת מערכים לרשימות מקושרות של שורות An, ו-Bn

for i:1to n //שורות

{

while(an[i]!=null ||bn!=null) //כל עוד לא הגיע לסוף

{

if(bn==null)

{

if(an!=null)

{

insert(Cn,Cm,i,key(An[i]))

An[i]<-next(An[i])

}

}

else if(An==null)

{

insert(Cn,Cm,i,key(Bn[i]))

Bn[i]<-next(Bn[i])

}

else //שיש ערך בשתיהם

{

insert(Cn,Cm,i,key(Bn[i])+key(An[i]))

An[i]<-next(An[i])

Bn[i]<-next(Bn[i])

}

}

}

הפונקציה רצה על כל השורות וכל העמודות, ומבצעת מספר סופי של פעולות בכל איטרציה, ולכן הסיבוכיות היא .

(הסיבוכיות אינה כי הפונקציה מדלגת על מקומות ריקים).