ביה"ס למדעי המחשב ומתמטיקה המכללה האקדמית נתניה
סמסטר ב' תשע"ו

21.5.2016

מבני נתונים
פתרון תרגיל מס' 5

1.

בשאלה זו הסימון: i mod n

משמעותו: שארית החלוקה של i ב- n.

לדוגמה: 20 mod 8 שווה ל- 4

להלן תוכנית רקורסיבית בשם P1 שמקבלת פרמטרים מערך A ומספר k.

נתח/י את סיבוכיות זמן הריצה (במונחים של ה- O הקטן ביותר שניתן להשגה בשיטות שלמדנו) של התוכנית כתלות ב- n, כאשר n מציין את גודל המערך A שמועבר לפונקציה כפרמטר.

```
P1(A,k)
n = length(A)
if n == 1 return A[1]
if n == 2 return A[2]
  for (i = 1; i ≤ √n ; i++) {
    A[i mod n ]=A[i+3 mod n]
 }
z=P1(A[1: ⌊n/2⌋ ],k)· P1(A[1: ⌊n/6⌋ ],k)
if A[ ⌊n/2⌋ ] ) ≤ k return z + P1(A[1: ⌊n/2⌋ ],k)
return P1( A[1: ⌊n/6⌋ ],k ) + P1( A[ ⌊n/7⌋ : ⌊2n/7⌋ ],k)
```

פתרון

נסמן ב- T(n) את זמן ביצוע התכנית כתלות ב- n, כאשר n מציין את גודל המערך שמועבר לפונקציה כפרמטר.

נשים לב שהאפשרות שתנאי ה- if מתקיים לוקחת יותר זמן מהאפשרות שהתנאי לא מתקיים. . מאחר ואנחנו מנתחים סיבוכיות זמן במקרה הגרוע ביותר נניח שהקלט לתכנית הוא כזה שבכל שלב תנאי ה- if מתקיים. נוסחת הנסיגה היא לכן:

$$T(n) \le C_1 + C_2 \cdot \sqrt{(n)} + 2 \cdot T(\lfloor n/2 \rfloor) + T(\lfloor n/6 \rfloor)$$

בשאלה זו לא נוכל להשתמש בשיטת המסטר ולכן נשתמש בשיטת עץ רקורסיה.

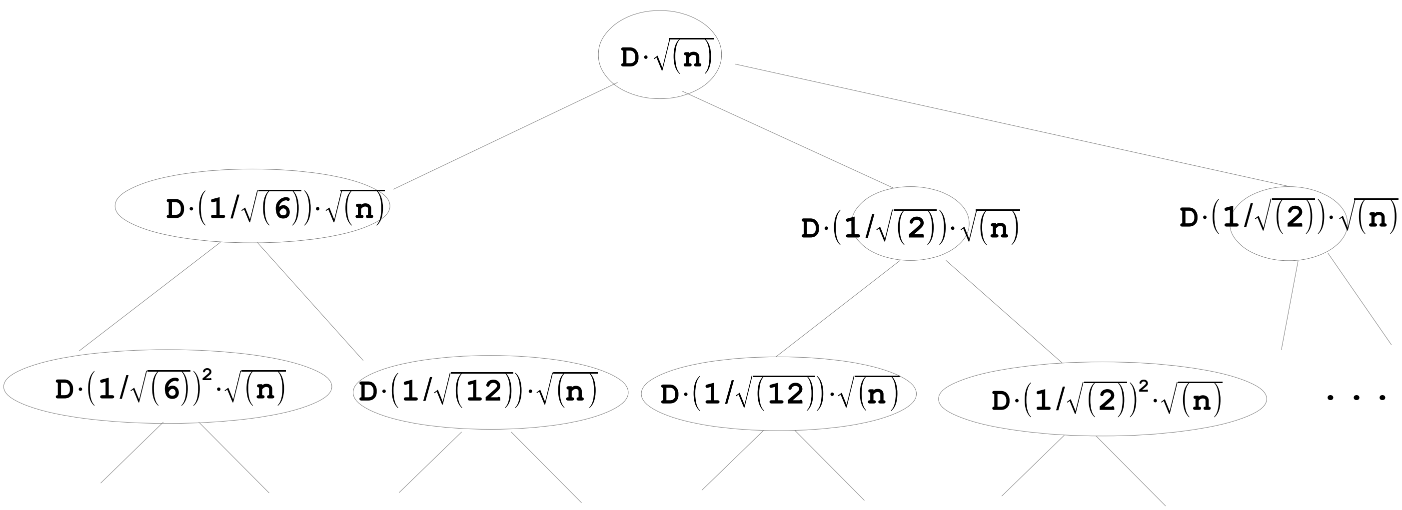
לכל n>1 קיים D כך ש-

$$C_1+C_2\cdot\sqrt{(n)}\le D\cdot\sqrt{(n)}$$

ולכן

$$T(n)\le D\cdot\sqrt{(n)}+2\cdot T(n/2)+T(n/6) \qquad (1)$$

עץ הרקורסיה המתאים לאי שוויון הנ"ל נראה כך:



סכום האיברים ברמה הראשונה בעץ הוא: $D\cdot\sqrt{(n)}$

סכום האיברים ברמה השניה בעץ הוא: $D\cdot(1/\sqrt{(6)}+2/\sqrt{(2)})\cdot\sqrt{(n)}$

סכום האיברים ברמה השלישית בעץ הוא: $D\cdot(1/\sqrt{(6)}+2/\sqrt{(2)})^2\cdot\sqrt{(n)}$

וכן הלאה.

נסמן $z=1/\sqrt{(6)}+2/\sqrt{(2)}$

נסמן ב- x את מספר הרמות בעץ.

$$T(n)\le D\cdot\sqrt{(n)}\cdot(1+z+z^2+\ldots+z^{x-1}) \qquad (2)$$

האיבר הימני ביותר ברמה 1 בעץ התקבל כתוצאה מהצבה של $T(n)$ בנוסחת הנסיגה (1)

האיבר הימני ביותר ברמה 2 בעץ התקבל כתוצאה מהצבה של $T(n/2)$ בנוסחת הנסיגה (1)

האיבר הימני ביותר ברמה 3 בעץ התקבל כתוצאה מהצבה של $T(n/2^2)$ בנוסחת הנסיגה (1)

...

האיבר הימני ביותר ברמה x בעץ התקבל כתוצאה מהצבה של $T(n/2^{x-1})$ בנוסחת הנסיגה (1)

מאחר והאיבר הימני ביותר ברמה x בעץ גדול מ- 0 הוא התקבל כתוצאה מהצבה של מספר שגדול או שווה ל- 1 בנוסחת הנסיגה. ולכן:

$$n/2^{x-1} \geq 1$$

ששקול ל-

$$2^{x-1} \leq n$$

ששקול ל-

$$(3) \quad x-1 \leq \log_2 n$$

מ- (3) ו- (2) נקבל:

$$(4) \quad T(n) \leq D \cdot \sqrt{(n)} \cdot \left(1+z+z^2+\ldots+z^{\log_2 n}\right)$$

ששקול ל-

$$(5) \quad T(n) \leq \frac{D \cdot \sqrt{(n)} \cdot \left(z^{\log_2 n+1}-1\right)}{z-1}$$

ולכן:

$$(6) \quad T(n) \leq D \cdot \sqrt{(n)} \cdot \left(z/(z-1)\right) \cdot z^{\log_2 n}$$

לפי חוקי לוגים נקבל:

$$(7) \quad z^{\log_2 n} = z^{\frac{\log_z n}{\log_z 2}} = \left(z^{\log_z n}\right)^{1/\log_z 2} = n^{1/\log_z 2}$$

מ- (6) ו- (7) נקבל:

$$(8) \quad T(n) \leq D \cdot \sqrt{(n)} \cdot \left(z/(z-1)\right) \cdot n^{1/\log_z 2} = \theta\left(n^{1/2+(1/\log_z 2)}\right)$$

ולכן הראנו ש-

$$(9) \quad T(n) = O\left(n^{1/2+(1/\log_z 2)}\right)$$

נציב את הערך של z ב- (9) ונקבל

$$(10) \quad T(n) = O\left(n^{1/2+\left(1/\log_{(1/\sqrt{(6)}+2/\sqrt{(2)})} 2\right)}\right)$$

.2

נתונה נוסחת הנסיגה הבאה:

$$T(n)=2\cdot T\left(\frac{n}{64}\right)+T\left(\frac{n}{27}\right)+T\left(\frac{n}{216}\right)+\sqrt[3]{n}$$

כאשר $T(i)=0$ עבור $i<1$.

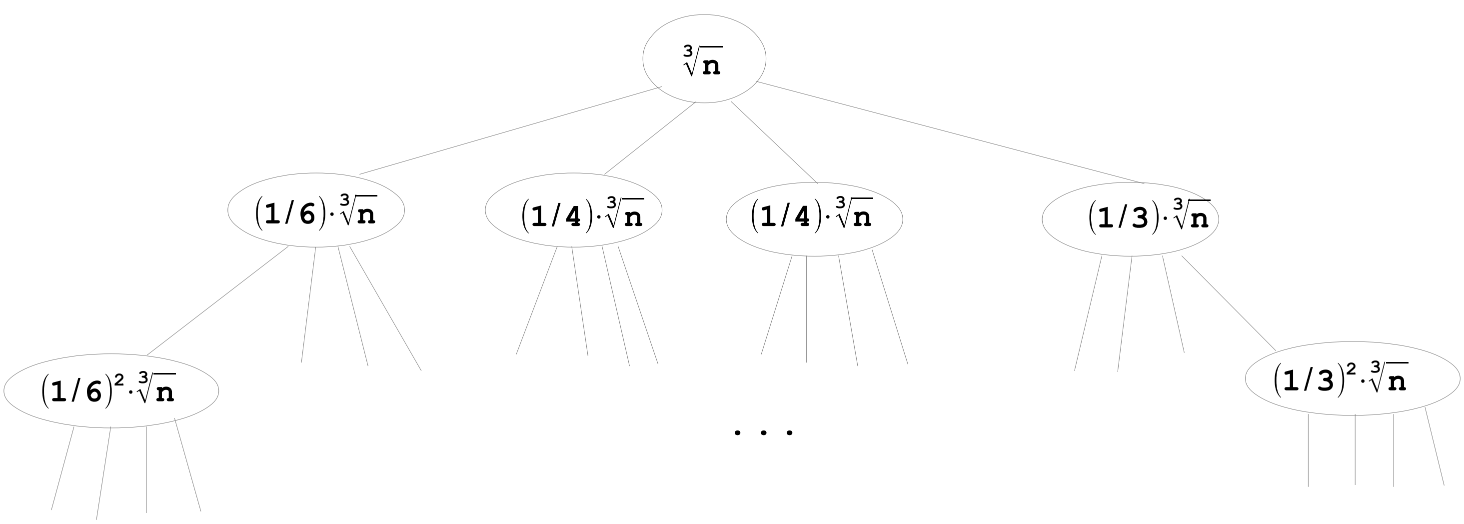
מצא/י את ה- O הקטן ביותר שאתה/את יודע/ת להשיג עבור $T(n)$.

נמק/י את תשובתך.

<u>פתרון</u>

בשאלה זו לא נוכל להשתמש בשיטת המסטר ולכן נשתמש בשיטת עץ רקורסיה.

עץ הרקורסיה המתאים לנוסחת הנסיגה הנ"ל נראה כך:



סכום האיברים ברמה הראשונה בעץ הוא: $\sqrt[3]{n}$

סכום האיברים ברמה השניה בעץ הוא: $\sqrt[3]{n}$

סכום האיברים ברמה השלישית בעץ הוא: $\sqrt[3]{n}$

וכן הלאה.

נסמן ב- **x** את מספר הרמות בעץ.

$(1) \quad T(n) \le \sqrt[3]{n} \cdot x$

האיבר הימני ביותר ברמה 1 בעץ התקבל כתוצאה מהצבה של $T(n)$ בנוסחת הנסיגה.

האיבר הימני ביותר ברמה 2 בעץ התקבל כתוצאה מהצבה של $T(n/27)$ בנוסחת הנסיגה.

האיבר הימני ביותר ברמה 3 בעץ התקבל כתוצאה מהצבה של $T(n/(27)^2)$ בנוסחת הנסיגה.

...

האיבר הימני ביותר ברמה x בעץ התקבל כתוצאה מהצבה של $T(n/(27)^{x-1})$ בנוסחת הנסיגה.

מאחר והאיבר הימני ביותר ברמה x בעץ גדול מ- 0 הוא התקבל כתוצאה מהצבה של מספר שגדול או שווה ל- 1 בנוסחת הנסיגה. ולכן:

$n/(27)^{x-1} \ge 1$

ששקול ל-

$(27)^{x-1} \le n$

ששקול ל-

$(3) \quad x-1 \le \log_{27} n$

ששקול ל-

$(4) \quad x \le 1+\log_{27} n$

מ- (4) ו- (1) נקבל:

$(5) \quad T(n) \le \sqrt[3]{n} \cdot (1+\log_{27} n) = \theta(\sqrt[3]{n} \cdot \log n)$

ולכן הראנו ש-

$(6) \quad T(n) = O(\sqrt[3]{n} \cdot \log n)$

.3

נתונה נוסחת הנסיגה הבאה:

$$T(n)=T\left(\frac{n-2}{2}\right)+T\left(\frac{n+3}{3}\right)+n$$

כאשר $T(i)=0$ עבור $i<1$.

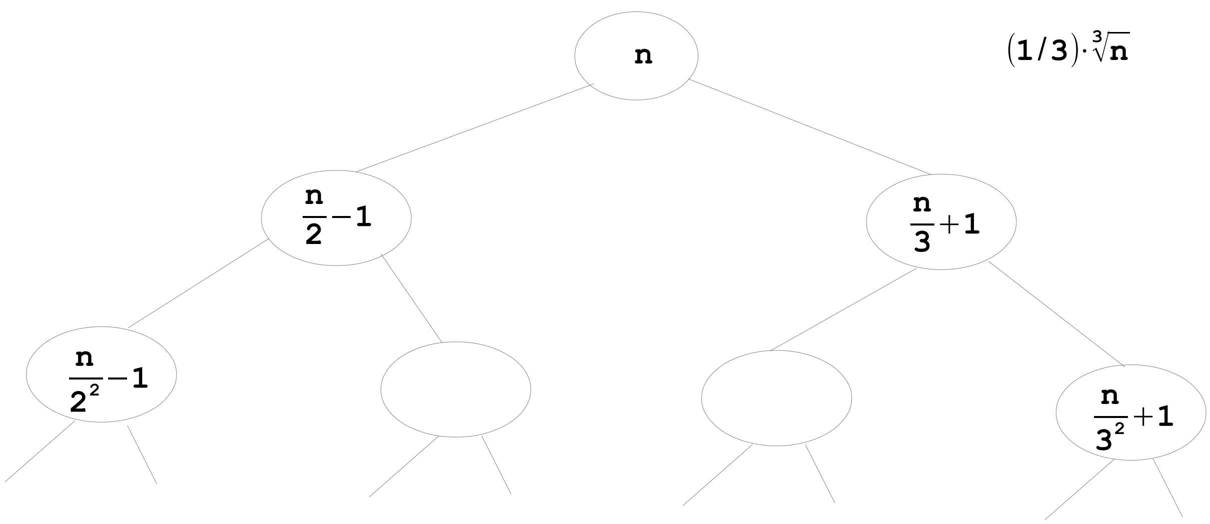
הערך/העריכי את $T(n)$ במונחים של θ .

נמק/י את תשובתך.

<u>פתרון</u>

בשאלה זו לא נוכל להשתמש בשיטת המסטר ולכן נשתמש בשיטת עץ רקורסיה.

עץ הרקורסיה המתאים לנוסחת הנסיגה הנ"ל נראה כך:



סכום האיברים ברמה הראשונה בעץ הוא: n

סכום האיברים ברמה השניה בעץ הוא: $(5/6)\cdot n$

סכום האיברים ברמה השלישית בעץ הוא: $(5/6)^2\cdot n$

וכן הלאה.

לכן:

$$T(n) \le n \cdot (1 + (5/6) + (5/6)^2 + \ldots) = n \cdot (\frac{1}{1-(5/6)}) = \theta(n)$$

לכן:

$$T(n) = O(n)$$

לכוון השני, מנוסחת הנסיגה נובע ש-

$$T(n) \ge n = \theta(n)$$

לכן:

$$T(n) = \Omega(n)$$

ולסיכום הראנו ש-

$$T(n) = \theta(n)$$

.4

נגדיר מבנה נתונים – "רשימת רשימות" L בעל התכונות הבאות :

- L (בהמשך נקרא לה "הרשימה הראשית") היא רשימה מקושרת דו-כיוונית, מעגלית ממוינת עם זנב. דהינו, head(L) מצביע לראש הרשימה הראשית L ולכל איבר x ברשימה הראשית L ישנם השדות הבאים: list(x),key(x),next(x),prev(x).
- לכל איבר x ברשימה הראשית L יש שדה list(x) שמצביע לרשימה מקושרת דו-כיוונית מעגלית ממוינת וללא זנב <u>לא ריקה</u> Q_x (הרשימה של האיבר x). כל איבר y ברשימה Q_x מכיל את השדות: key(y),key1(y),info(y),next(y),prev(y). לרשימות מהסוג Q_x נקרא בהמשך "הרשימות המשניות". הערך של השדה key1(y) של איבר y ברשימה Q_x שווה לערך של key(x).
- האיברים ברשימה הראשית L מקיימים את התכונה הבאה: הערך של המפתח של איבר x ברשימה הראשית L שווה לערך של השדה key1(y) עבור כל איבר y שנמצא ברשימה Q_x .

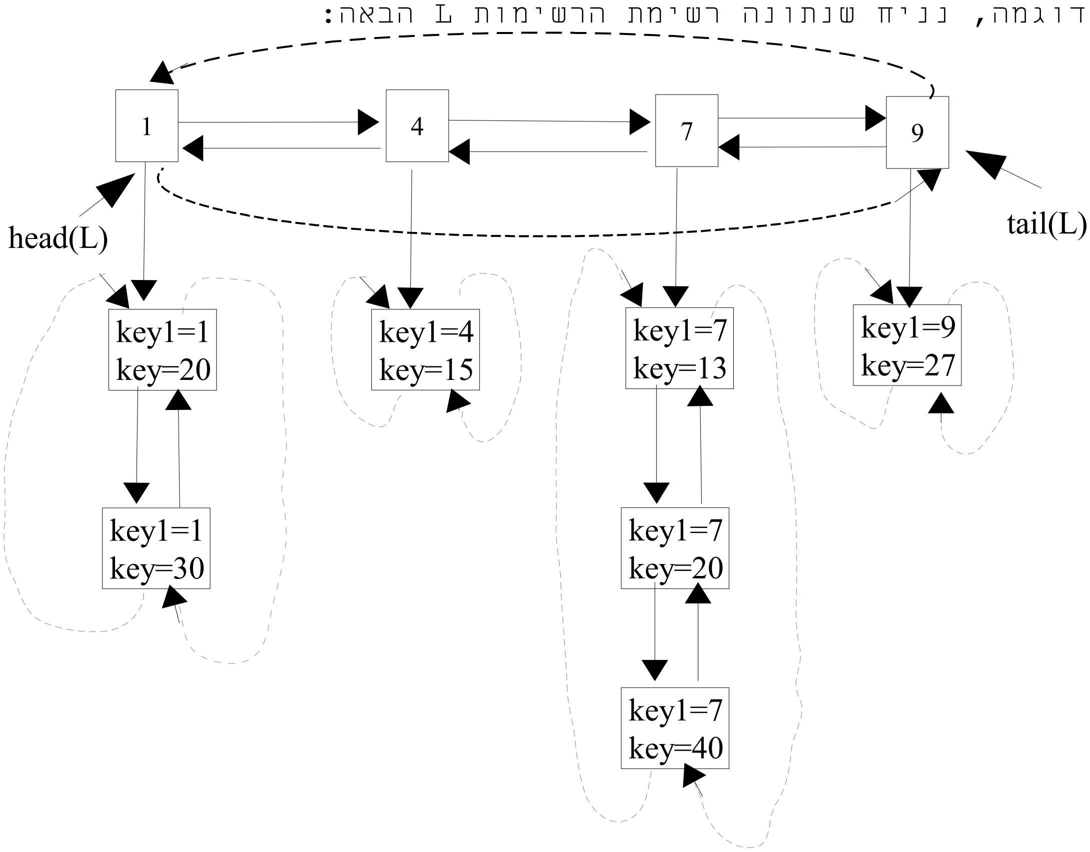
ניתן להניח שמפתחות כל האיברים ברשימה הראשית L שונים זה מזה וכן מפתחות כל האיברים ברשימות המשניות שונים זה מזה.

כתוב/י פסאודו-קוד של פונקציה בשם P4(L,k1,k2,c) יעילה ככל האפשר שמקבלת כפרמטרים רשימת רשימות L (כפי שהוגדר למעלה), ושלושה מספרים k1,k2,c מחפשת את האיבר y שנמצא ברשימה משנית כלשהי ומקיים key1(y)=k1 ו- key(y)=k2. במידה והאיבר y לא קיים הפונקציה מודיעה על שגיאה. במידה והאיבר y קיים הפונקציה מציבה key1(y)=k1+c, מוציאה אותו מהרשימה המשנית בה הוא נמצא ומעבירה אותו לרשימה המשנית Q_x של האיבר x שמקיים key(x)=k1+c. במידה והאיבר x לא קיים התכנית יוצרת איבר חדש x ברשימה הראשית ומעבירה את y להיות האיבר היחיד ברשימה המשנית שלו Q_x.

נתח/י את סיבוכיות הזמן של הפונקציה P4 שכתבת כתלות ב- n ו- m כאשר n מציין את מספר האיברים ברשימה הראשית ו- m מציין את מספר האיברים ברשימה המשנית הגדולה ביותר מבין כל הרשימות המשניות.

מותר להשתמש בפונקציות עזר, אך יש לפרט את הפסיאודו קוד של פונקציות העזר.

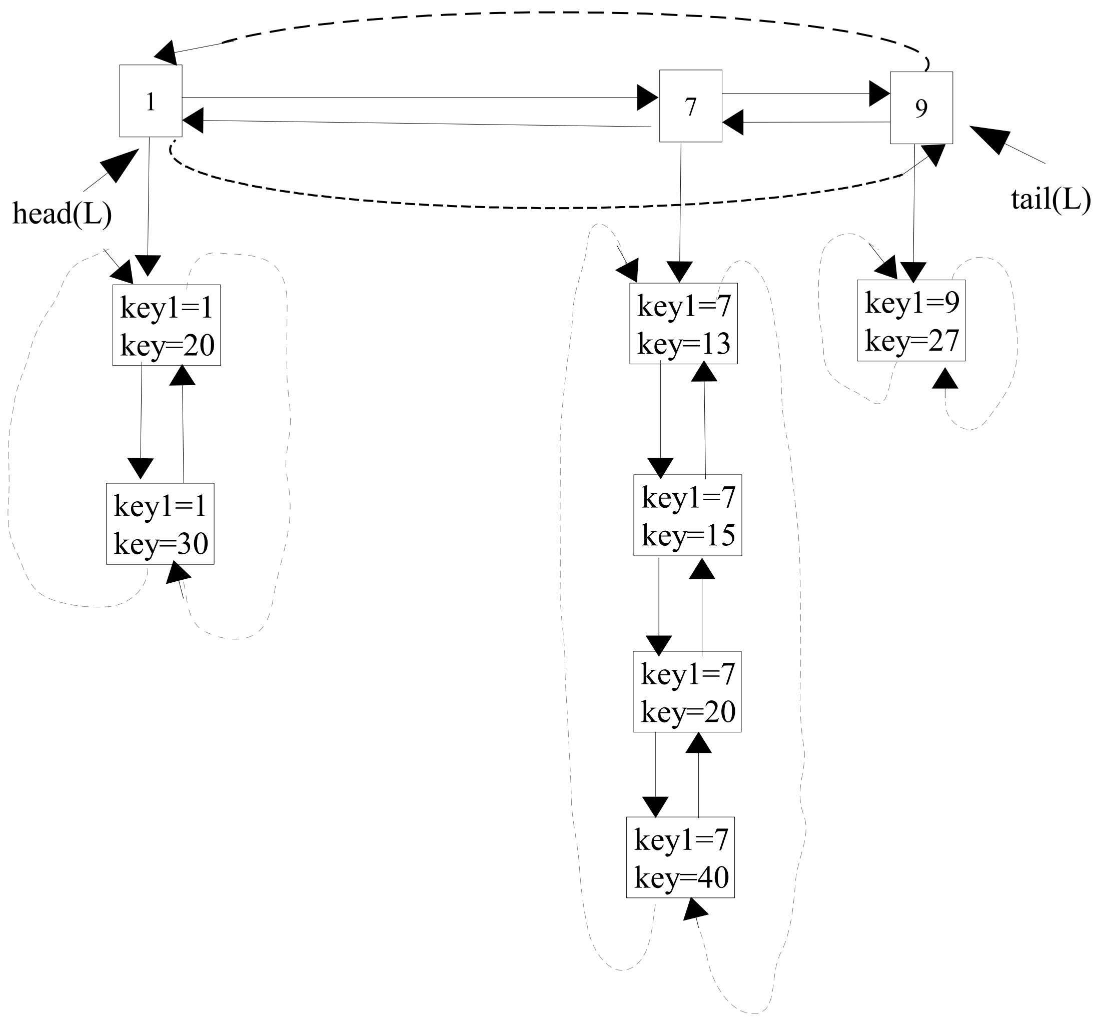
לדוגמה, נניח שנתונה רשימת הרשימות L הבאה:



לאחר הקריאה לפונקציה P4(L,4,11,3) תתקבל הודעה : איבר לא קיים.

לאחר הקריאה לפונקציה P4(L,4,15,3) האיבר שהמפתחות שלו הם key1=4 key=15 יוצא מהרשימה המשנית של האיבר שהמפתח שלו הוא 4 ויועבר לרשימה המשנית של האיבר שהמפתח שלו הוא 7 במקום המתאים. מאחר ואיבר ברשימה הראשית שהמפתח שלו הוא 4 אין איברים ברשימה המשנית הוא יוצא מהרשימה הראשית.

הרשימה שתתקבל לאחר הפעולה הנ"ל היא:



יש להגיש את התרגיל בתא הקורס שנמצא בתא מספר 6 בתאים שנמצאים מול מול מזכירות מנהל עסקים (לא בתא של המרצה).
כתוב על התא: הגשת תרגילים במבנה נתונים.

<u>פתרון</u>

<u>**P4(L,k1,k2,c)**</u>

```
x=head(L)
if (x==NULL) { print "element does not exist"; return }
while true {
 if (key(x)==k1) { P4.1(L,k1,k2,c,x); return}
 if (next(x)==head(L)) { print "element does not exist"
                         return }
 x-next(x)
 }
```

<u>**P4.1(L,k1,k2,c,x)**</u>

```
z=list(x)
z1=list(x)
while true {
 if (key(z)==k2) { P4.2(L,k1,k2,c,x,z); return}
 if (next(z)==z1) { print "element does not exist"
                    return }
 z=next(z)
 }
```

```
P4.2(L,k1,k2,c,x,z)
delete_list2(Qx,z)
t=new element of secondary linked list
key1(t)=k1+c
key(t)=k2
if (head(Qx)==NULL) {delete_list1(L,x)}
w=search(L,k1+c)
if (w==NULL) { v=new element of main linked list
                key(v)=k1+c
                create_list2(L1)
                insert_list2(L1,t)
               list(v)=head(L1)
               insert_list1(L,v)
               return
              }
insert_list2(Qw,t)
```

```
create_list2(L)
head(L)=NULL
```

```
insert_list1(L,s)
x=head(L)
if (x==NULL) {head(L}=s; next(s)=s; prev(s)=s; tail(L)=s;
              return}
while true {
  if (key(x)>key(s)) {
        if (x != head(L) {
             next(s)=x; prev(s)=prev(x)
             next(prev(x))=s; prev(x)=s; return
             }
        head(L)=s; next(s)=x; prev(s)=prev(x)
        prev(x)=s; return
      }
  if (next(x)=head(L)) {
     next(x)=s; next(s)=head(L); tail(L)=s; prev(s)=x
     return
    }
  x=next(x)
 }
```

```
insert_list2(L,s)
x=head(L)
if (x==NULL) {head(L}=s; next(s)=s; prev(s)=s;
              return}
while true {
  if (key(x)>key(s)) {
        if (x != head(L) {
             next(s)=x; prev(s)=prev(x)
             next(prev(x))=s; prev(x)=s; return
             }
        head(L)=s; next(s)=x; prev(s)=prev(x)
        prev(x)=s; return
      }
  if (next(x)=head(L)) {
     next(x)=s; next(s)=head(L); prev(s)=x
     return
    }
  x=next(x)
 }
```

```
delete_list1(L,s)
if (head(L)==s) && (tail(L)==s) {
   head(L)=null; tail(L)==null; return
   }
if (head(L)==s) { head(L)=next(s) }
if (tail(L)==s) { tail(L)=prev(s) }
prev(next(s))=prev(s);next(prev(s))=next(s)
```

```
delete_list2(L,s)
if (next(s)==s) {
   head(L)=null; return
   }
if (head(L)==s) { head(L)=next(s) }
```

```
prev(next(s))=prev(s); next(prev(s))=next(s)
```