2016A-85



A[i] ← מקסימום; A[j] ← מינימום.

FindMax(A,p,r)

if p>=r then

return NIL

q←

if A[q-1]<A[q]>A[q+1] then

return q

else if A[q-1]<A[q]

if A[p]<A[q]

return FindMax(A.q,r)

else

return FindMax(A.p,q)

else return FindMax(A.p,q)

FindMin(A,p,r)

if p>=r then

return NIL

q←⌊(p+r)/2⌋

if A[q-1]>A[q]< [q+1] then

return q

else if A[q-1]>A[q] then

return FindMin(A,q,r)

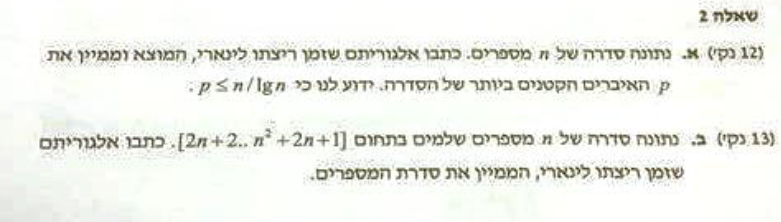
else

if A[p]>A[q]

return FindMin(A,p,q)

else

return FindMax(A,q,r)



SortPsmallest(A)

i←

x←Select(A,i)

Create P - an array in the size of i

P[1]←x

j←1

k←2

while j<=n and k<=i do

if A[j]<=x then

P[k]←A[j]

k←k+1

j←j+1

Merge-Sort[P]

return P

ב. נחסר את כל המספרים ב- (2n+2). נקבל סדרה בתחום שבין 0 לבין n^2-1. נציג כל מספר בהצגה בבסיס n. במקרה שלנו החזקה של היא 2 ולכן ספרת ראשונה תבטא ערך של n^0 וספרה שניה תבטא ערך של n^1. נשתמש במיון בסיס למיון המספרים. במערך הממויין שקיבלנו, כל איבר יקבל בחזרה 2n+2 .

סיבוכיות. מיון בסיס משתמש במיון מניה עבור מיון כל ספרה. סיבוכיות מיון מניה תלויה בגודלה המערך – n ובגודל טווח הערכים שהוא יכול להכיל. במקרה שלפנינו כל ספרה יכולה לקבל ערכים בתחום [0..n-1] ולכן סיבוכיות מיון כל ספרה תהיה Θ(n+n-1) = Θ(n). סיבוכיות מיון בסיסו בענייננו תהיה Θ(2\*n).



InversionCount(A,p,q,r)

counter←0

n1←q-p+1

n2←r-q

create arrays L[1..n1] and R[1..n2]

for i←1 to n1 do

L[i]←A[p+i-1]

for j←1 to n2 do

R[j]←A[q+j]

i←1

j←1

k←1

while i<=n1 and j<=n2

if L[i]<R[j] then

i←i+1

A[k]←L[i]

k←k+1

else

counter←counter+1

A[k]←R[j]

j←j+1

if i=n1 and j!=n2

copy the rest of R to A

else if i!=n1 and j=n2

copy the rest of L to A

InversionTotalCount(A,p,r)

x←0

if p>=r then

return x

else

q←

x←x+ InversionTotalCount(A,p,q)

x←x+ InversionTotalCount(A,q+1,r)

x←x+ InversionCount(a,p,q,r)



FindTraverses(T,x)

if left[x]=NIL and right[x]=NIL

dmin[x]←0

dmax[x]←0

if left[x]≠NIL and right[x]≠NIL

FindTraverses(T,left[x])

FindTraverses(T,right[x])

dmin[x]← min(dmin[left[x]],dmin[right[x]]) +1

dmax[x]← max(dmax[left[x]],dmax[right[x]]) +1

else if left[x]≠NIL

FindTraverses(T,left[x])

dmin[x]←dmin[left[x]]+1

dmax[x]←dmax[left[x]]+1

else

FindTraverses(T,right[x])

dmin[x]← dmin[right[x]]+1

dmax[x]← dmax[right[x]]+1

FindMaxFullTree(T,x)

if x=NIL

return

FindMaxFullTree(T,left[x])

FindMaxFullTree(T,right[x])

if dmin[x]=dmax[x]

maxFull = dmin[x]

return x



S –

T עא"ש ראשי. כל מפתח מצביע על ערימת מקסימום. באתחול – הערימה הזו עדיין ריקה.

n ערימות מקסימום עבור כל מפתח n. לכל ערימה צמוד משתנה size ובו גודל הערימה. כל תא בערימה בעל מצביע חזרה לצומת המתאים בעץ T

ערמת מינמום עם ערכי המפתחות בעץ T. כל צומת בערימה מצביע על צומת מתאים בעץ T

insert – מכניסים את המפתח k לעא"ש T כרגיל. אח"כ נכניס את המפתח לראש הערימה המתאימה לצומת k עם ערך השווה ל-size+1. נבצע MaxHeapify. לאחר מכן נכניס את הערך k לראש ערימה המינמום ונבצע MinHeapify.

DeleteLast(S,k) נחפש בעץ T את הערך k. לאחר שמצאהו נשתמש במצביע שלו כדי להגיע לערימת המקסימום המתאימה. נחזיר מהערימה את צומת האיבר המקסימלי. באמצעות המצביע שבצומת זה נחזור לעץ T, הפעם לצומת המתאים ונמחוק אותו.

DeleteMinLast(S,k) מחיקת האיבר שבראש ערימת המינימום. בגלל אופן ההכנסה ואופן פעולת שגרת MinHeapify אם האיבר האחרון שהוכנס הוא שווה ערך למינימום – לא תבוצע החלפה וכך יוצא שהמינמלי האחרון שנכנס הוא המינימלי האחרון בראש הערימה מבלי שנצטרך להוסיף פעולות.

S -

T עא"ש ראשי. כל צומת מחזיק בנוסף לשדות ה"רגילים" גם מצביע על מחסנית. באתחול – המחסנית ריקה.

n מחסניות ל-n מפתחות. כל תא במחסנית הוא מצביע חזרה לצומת המתאים בעץ T.

ערמת מינימום עם ערכי המפתחות בעץ T. כל צומת בערימה מצביע על צומת מתאים בעץ T.

insert – מכניסים את המפתח k לעא"ש T כרגיל. אם קיים כבר מפתח k – נהיה חייבים לעבור דרכו. "ניקח" ממנו את המצביע למחסנית. אחרי ההכנסה לעץ T. נכניס אח"כ מצביע לצומת שהוכנס למחסנית.

DeleteLast(S,k) - נחפש בעץ T את הערך k. לאחר שמצאנו נשתמש במצביע שלו כדי להגיע למחסנית המתאימה. נחזיר מהערימה את המצביע לצומת האחרון שהוכנס. באמצעות המצביע נחזור לעץ T, הפעם לצומת המתאים ונמחוק אותו.

DeleteMinLast(S) - מחיקת האיבר שבראש ערימת המינימום. בגלל אופן ההכנסה ואופן פעולת שגרת MinHeapify אם האיבר האחרון שהוכנס הוא שווה ערך למינימום – לא תבוצע החלפה וכך יוצא שהמינמלי האחרון שנכנס הוא המינימלי האחרון בראש הערימה מבלי שנצטרך להוסיף פעולות. נשתמש במצביע כדי לגשת לצומת המתאים בעץ T. הערך המתאים הוא עלה. את המחיקה נעשה באמצעות ניתוק העלה – לא באמצעות שגרת המחיקה המשתמשת בתת שגרות הפועלות בסיבוכיות Θ(h). רק ניתוק העלה ו-RB-Fixup.

