1. gyakorlat

Téma:

Algoritmusok műveletigényének meghatározása, hatékonyság, hatékonyság jellemzése (aszimptotikus korlátok bevezetése). Az anyag úgy lett összeállítva, hogy akkor is elvégezhető, ha még nem volt előtte előadás.

Polinom helyettesítési értékének kiszámítása.

Adott egy n-ed fokú polinom, határozzuk meg egy adott x helyen felvett értékét:
$a_n*x^n+a_{n-1}*x^{n-1}+ \ldots.\ a_1*x+a_0$
(Tfh. nagyon sok polinomunk van, és nagyon sok helyen kell kiszámítani az értékét, ezért készítsünk minél hatékonyabb megoldást.)

A polinom együtthatóit egy nullától indexelt, n+1 méretű tömbben helyezzük el. (Megállapodás: ha a tömböt nem nullától indexeljük, a deklarációnál és a specifikációnál jelezzük, pl. A/1:T[n]. Most tehát Z:R[] ugyanaz, mint Z/0:R[].) A Z tömb mérete: Z.length (fontos, hogy: Z.length=n+1).

A megoldásoknál írjuk fel, hogy az egyes lépések hányszor hajtódnak végre. Vizsgáljuk meg a ciklusiterációk *it(n)*, a szorzások *S(n)* és az összeadások *Ö(n)* számát, a polinom fokszámának függvényében.

Feltehető, hogy n≥0, azaz Z.length>0

Első megoldás, az összegzés tételéből származik:

Polinom1(Z:R[]; x:R) :R	*Hányszor fut le (Z.length=n+1)*
y:= Z[0]	*1*
i = 1 to Z.length-1	*n+1 (ciklusfeltétel kiértékelés)*
h:=x	*n*
j= 1 to i-1	*1+2+3+…+n-1+n*
h:=h*x	*0+1+2+3+…+n-1*
y:=y+h*Z[i]	*n*
return y	*1*

$$S(n) = \frac{n*(n+1)}{2} = \frac{n^2+n}{2}$$

$\text{Ö}(n) = n$ *it(n) = S(n)*

Második megoldás, x hatványait rekurzívan számoljuk a h változóban: $x^i=x^{i-1}*x$, ha i>0, $x^0=1$

Rekurzív(Z:R[]; x:R) :R	*Hányszor fut le (Z.length=n+1)*
y:=Z[0] h:=1	*1*
i = 1 to Z.length-1	*n+1*
h:=h*x	*n*
y:=y+h*Z[i]	*n*
return y	*1*

$S(n) = 2*n$

$\text{Ö}(n) = n$ *it(n) = n*

Harmadik megoldás, a Horner séma:

$y=(\dots(a_n*x+a_{n-1})*x+a_{n-2})*x+ \dots + a_1)*x + a_0$

Horner(Z:R[]; x:R) :R	*Hányszor fut le (Z.length=n+1)*
y:=Z[Z.length-1]	*1*
i= Z.length-2 downto 0	*n+1*
y:=y*x+Z[i]	*n*
return y	*1*

$S(n) = n$ *it(n) = n*

$Ö(n) = n$

Jellemezzük a három megoldást a Θ aszimptotikus korlát segítségével. Itt is láthatjuk, és általánosságban is mondhatjuk, hogy *it(n)* a futási idő nagyságrendjét általában minden nemrekurzív program esetében megadja:

	Polinom1	Rekurzív	Horner
S(n)	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n)$
Ö(n)	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$
it(n)	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n)$

Buborék rendezés

Nézzük meg az összehasonlítások Öh(n) és cserék számát Cs(n). Cserék elemzésénél használjuk a mCs(n), MCs(n) ACs(n) (minimum, maximum, átlagos csereszám) jelöléseket. Átlagos csere számot nem kell pontosan kiszámolni, elég csak a „megérzés”-re támaszkodni.

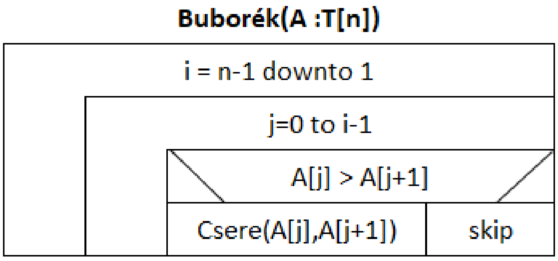
A rendezés menete egy rövid példán:

Buborék példa:

0	*1*	*2*	*3*	*4*	Csere
3	5	2	4	1	0
3	5	2	4	1	1
3	2	5	4	1	1
3	2	4	5	1	1
3	2	4	1	5	*1. menet vége, 5 a helyén van*
3	2	4	1	5	1
2	3	4	1	5	0
2	3	4	1	5	1
2	3	1	4	5	*2. menet vége*
2	3	1	4	5	0
2	3	1	4	5	1
2	1	3	4	5	*3. menet vége*
2	1	3	4	5	1
1	2	3	4	5	*4. menet vége, rendezett a tömb*

Csere összesen: **7**
Összehasonlítás összesen: **10**

A rendezendő kulcsokat (és a hozzájuk tartozó adatokat) egy A nevű tömbben helyeztük el. A.length = n, a rendezendő kulcsok darabszáma.



Hányszor fut le (A.length=n)

n

n+n-1+ … +2

n-1+n-2+ … +1

Csereszám?

Az összehasonlítások száma $Ö(n) = \sum_{i=1}^{n-1} i = \frac{n*(n-1)}{2} = \frac{n^2-n}{2} \in \Theta(n^2)$

Cserék számát hogyan tudjuk meghatározni?

Cserék száma a rendezendő adatsorban található inverziók számával egyenlő. Lásd a példában 7 inverzió van: 3,2 3,1 5,2 5,4 5,1 2,1 4,1

Ebből adódik, hogy mCs(n)=0 (nincs inverzió, azaz növekvően rendezett a bemenet)

MCs(n)= Ö(n) (minden összehasonlítást csere követ, azaz fordítottan rendezett a tömb)

ACs(n)= $\frac{n*(n-1)}{4} = \Theta(n^2)$ Ezt nem kell pontosan levezetni, a lejjebb megadott linken megtalálható.

Vezessük be az Ω és O aszimptotikus korlátokat, és használjuk a csere számra:
mCs(n)=0, MCs(n)=$\Theta(n^2)$ azaz Cs(n)=$O(n^2)$

Az átlagos futási idő kiszámítása részletesen megtalálható dr Fekete István jegyzetében:
https://people.inf.elte.hu/fekete/algoritmusok_jegyzet/01_fejezet_Muveletigeny.pdf

A buborék rendezés javítási módszerei:

- figyelhetjük egy logikai változóval, hogy volt-e csere, ha nem volt akkor a külső ciklus álljon le,
- megjegyezhetjük az utolsó csere helyét: ha ez u és u+1 indexen történt, akkor u+1-től már a tömb rendezett, a külső ciklus változót u-ra lehet csökkenteni. Legkedvezőbb és legrosszabb esetek: mÖ(n)$\in\Theta$(n), MÖ(n)$\in\Theta(n^2)$.
  mT(n)$\in\Theta$(n), MT(n)$\in\Theta(n^2)$; azaz mT(n),MT(n)$\in\Omega$(n), mT(n),MT(n)$\in O(n^2)$

Javított buborék példa:

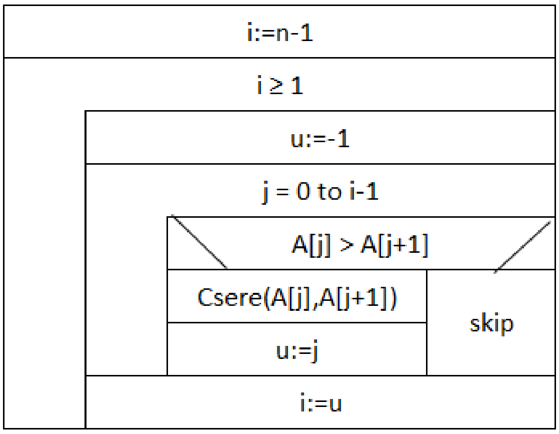
0	*1*	*2*	*3*	*4*	Csere	
2	3	1	4	5	0	
2	3	1	4	5	1	u=1
2	1	3	4	5	0	
2	1	3	4	5	0	
2	1	3	4	5	*1. menet vége 3,4,5 rendezett*	
2	1	3	4	5	1	u=0
1	2	3	4	5	*kész*	

Csere összesen: **2**

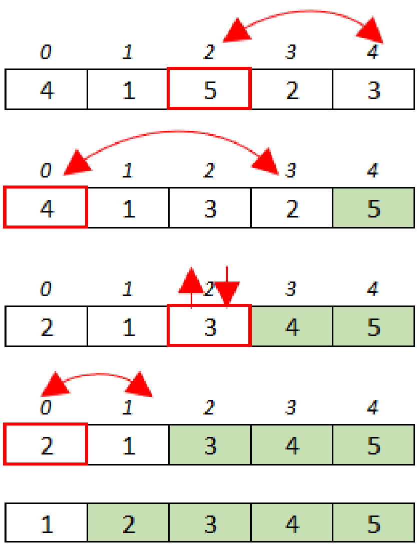
Összehasonlítás összesen: **5**

Struktogramja, elemzés nélkül:

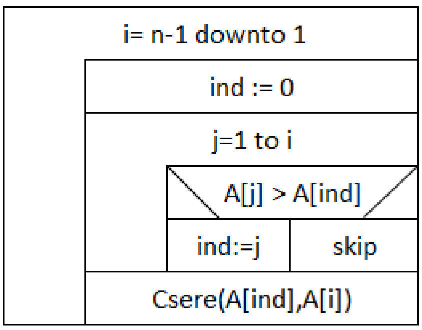
JavítottBuborék(A :T[n])



1. A maximum kiválasztásos rendezés



MaxKivRend(A:T[n])



Hányszor fut le (A.length=n)

n

n-1

n+n-1+ … +2

n-1+n-2+ … +1

n-1

Mit mondhatunk a MaxKivRend rendezés összehasonlításainak számáról, csereszámáról, műveletigényéről?

- mCs(n)$\in\Theta$(n), MCs(n)$\in\Theta$(n) (minden menet végén csere van!)
  mÖ(n)$\in\Theta$(n^2), MÖ(n)$\in\Theta$(n^2) (minden esetben minden összehasonlítást megcsinál)
  mT(n)$\in\Theta$(n^2), MT(n)$\in\Theta$(n^2)

Házi feladat (a kvízhez kell!): Milyen lehet a MinKivRend rendezés?