1. Gyakorlat

Algoritmusok és adatszerkezetek 1.

Eljárásrend

- Én: Máriás Zsigmond - Prog Mat Diploma 2007-ben, doktori abszolutórium 2013, egyébként a szakmában dolgozom, mint cégvezető
- Előfeltételek: nem ellenőrzöm, nézzenek utána
- Hogyan lesz a félév:
  - 2 db. ZH, mindkettőnek meg kell lennie a jegyhez
    - [1...3.5] lefele kerekítünk, (3.5..5] felfele kerekítünk
    - Ha az egyik nem sikerül → pótZH, ha a pótZH nem sikerül vagy egyik sem → GyakUV
  - Be kell járni az órára, lesznek órai tesztek (tantárgyi elvárás)
  - Plusz pontért lehet majd programozós feladatot megoldani

Alapok

- Algoritmus: eljárás, definiált lépéssorozat ami egy feladattípust old meg
  - Konstrukciós eszközök
    - Elemi lépések - értékadás, logikai műveletek
    - Programkonstrukciók - szekvencia, elágazás, ciklus
    - Függvények, rekurzió
    - Memóriakezelés, pointerek
  - Eszköz amit használunk: struktogram
- Adatszerkezet: dinamikus halmazok, olyan tulajdonságokkal és műveletekkel amelyek jól jönnek az algoritmusaink konstrukciója során

Ajánlott irodalom

- Hivatalos Algoritmusok jegyzet
- Rónyai-Ivanyos-Szabó: Algoritmusok
- Cormen, Leiserson, Rivest, Stein - Új algoritmusok

1. Félév anyaga

- Algoritmusok műveletigénye
- Alapvető adatszerkezetek
  - tömb, sor, verem
  - bináris fák, keresőfák
  - Láncolt ábrázolás
- Rendezések
- Keresés, kiválasztás
- Hashelés

Egyéb

- Ha esetleg elmarad egy gyakorlat akkor azt be fogjuk pótolni
- Lesz konzultáció a ZH-k előtt

És akkor a lényeg

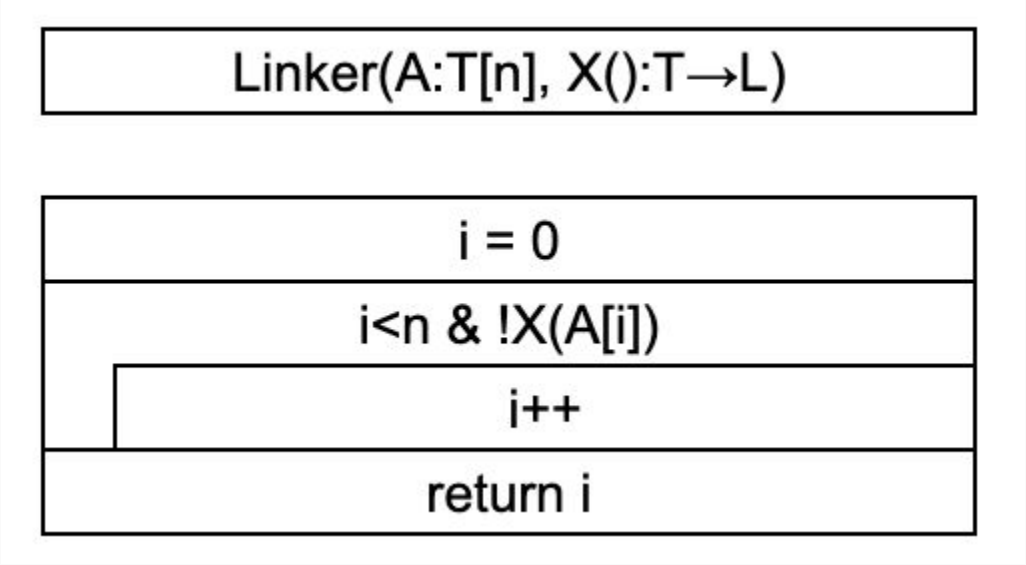
Lineáris keresés műveletigénye

1. Feladat: írjuk meg a lineáris keresés algoritmusát

Van egy T[] tömbünk, amelynek n db. eleme van (0..n-1 indexelünk), adjunk egy algoritmust, amely visszaadja az első i indexet, amelyen szereplő érték az X() tulajdonságnak megfelel, különben n-t

X(a) tulajdonság lehet pl. Hogy a==b, ekkor a tömbben a b elem első előfordulását keressük

Megoldás (nem az egyetlen)

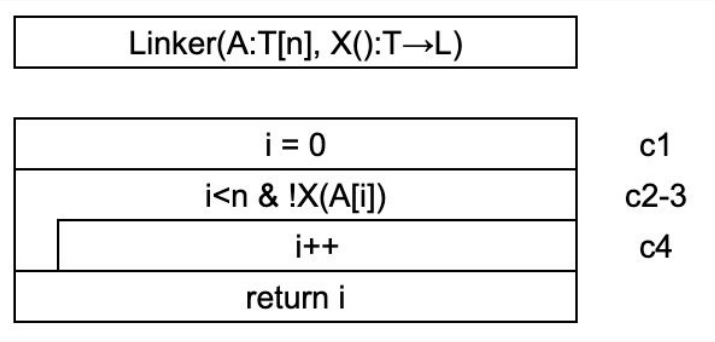


Műveletigénye

Konkrétan: az A egy 10 elemű tömb, az 5-öt keressük (X pontosan akkor igaz, ha az argumentum = 5)

A: [1,3,7,6,4,11,5,7,2,5]

Ekkor pontosan ki lehet számolni, hogy pontosan hány lépést végez



Műveletigénye

Ez így kicsit túl részletes.

Melyik műveletet vizsgáljuk?

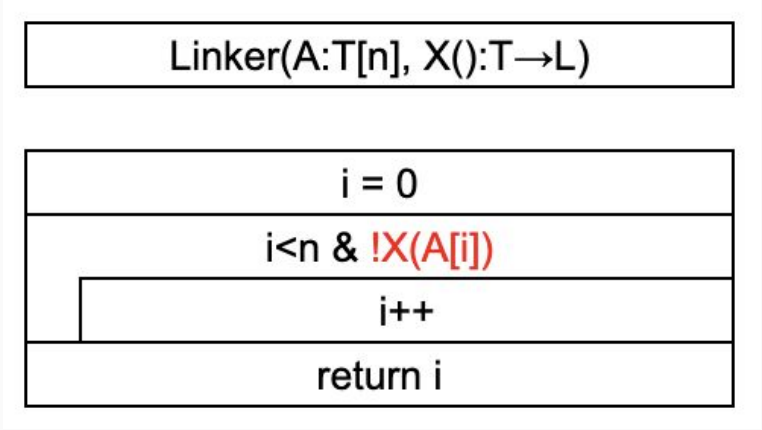
A legdrágábbat, ami a legtöbbször fut le.

Milyen értékekre:

N elemű tömbre vagyunk kíváncsiak

Random bemenetre // ettől azért sok minden függ

Legjobb, legrosszabb, átlagos eset?



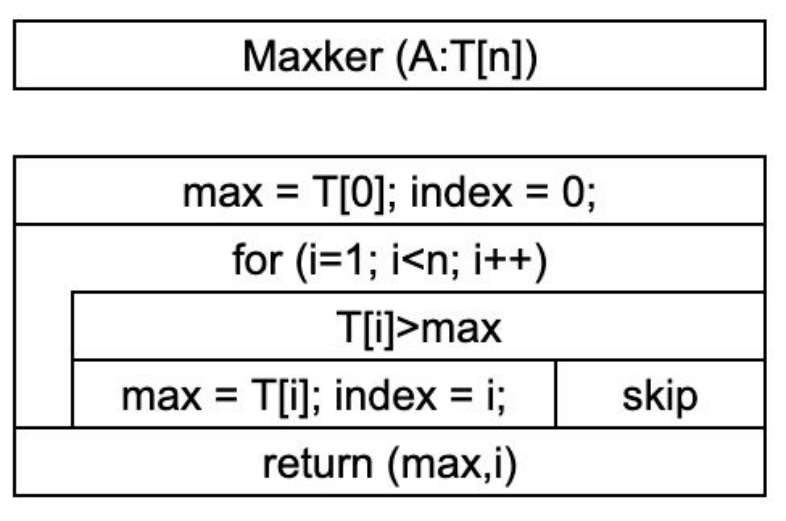
Feladat

Maxker algoritmus átbeszélése:

1. Mindenki írja meg a maximumkeresés struktogramját

2. Találja ki, hogy általában hány műveletet végez

Maxker



Műveletigény: n

Minden elemet meg kell nézni, hogy a legnagyobbat megtaláljuk

Legjobb, legrosszabb, átlagos eset?

Megegyezik, hiszen végig kell néznünk az összes elemet.

AT(n)=MT(n)=mT(n)=O(n)

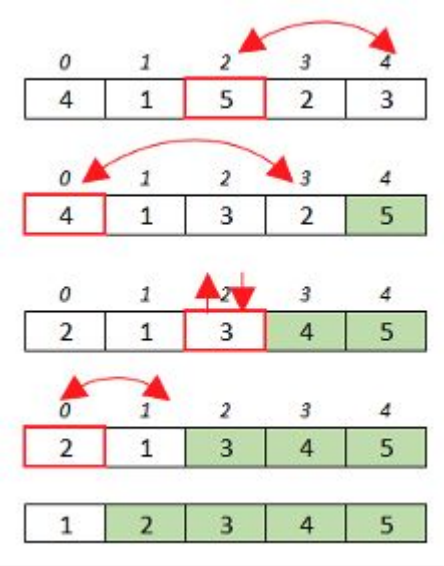
Max kiválasztásos rendező

Tekintsük az A[0..n) tömb rendezését a következő algoritmussal! Először megkeressük a tömb maximális elemét, majd megcseréljük A[n − 1]-gyel. Ezután megkeressük a második legnagyobb elemét és megcseréljük A[n−2]-vel. Folytassuk ezen a módon az A[0..n) utolsó (n−1) elemére! Pl.:

⟨3;1;9;2;7;6⟩ → ⟨3;1;6;2;7 | 9⟩ → ⟨3;1;6;2 | 7;9⟩ → ⟨3;1;2 | 6;7;9⟩ → ⟨2;1 | 3;6;7;9⟩ → ⟨1 | 2;3;6;7;9⟩ → ⟨1;2;3;6;7;9⟩

Írjunk struktogramot erre a maximumkiválasztásos rendezés néven közis- mert algoritmusra MaxKivRend(A : T[n]) néven! Mi lesz a fő ciklus invariánsa? Miért elég csak az utolsó (n − 1) elemre lefuttatni? Adjuk meg az MT(n) és mT(n) függvényeket a maximum kiválasztásos rendezésre a szokásos Θ-jelöléssel!

Max kiválasztásos rendező



MaxKivRend(A:T[n])	*Hányszor fut le (A.length=n)*
i= n-1 downto 1	*n*
ind := 0	*n-1*
j=1 to i	*n+n-1+ … +2*
A[j] > A[ind]	*n-1+n-2+ … +1*
ind:=j / skip	
Csere(A[ind],A[i])	*n-1*

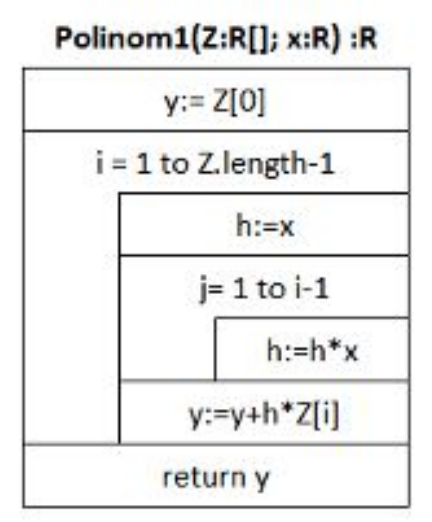
Polinom helyettesítési értékének kiszámolása

Adott egy polinom - reprezentáció: tömb, ahol a tömb értékei az együtthatók.

A: $a_0, a_1, \dots a_n$

Polinom: $a_0*X^0 + a_1*X^1 + \dots + a_n*X^n$

Naiv megoldás



Melyik művelet számít:

Amelyik a legdrágább és amelyik a legtöbbször fut le.

Hint: a legbelső ciklusmag általában

Ennek a futási ideje n^2

Legjobb, legrosszabb, átlagos eset?

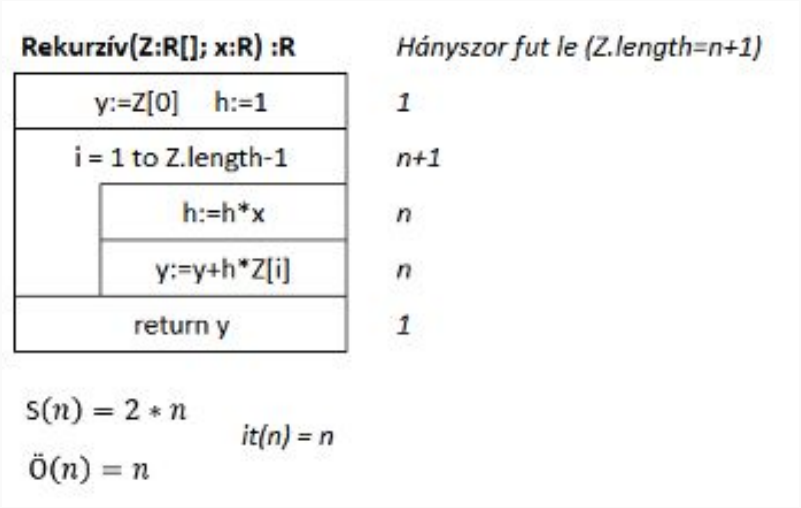
Műveletigény elemzés

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*

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

$\tilde{O}(n) = n$ *it(n) = S(n)*

Egy jobb megoldás



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*

$S(n) = 2 * n$

$\ddot{O}(n) = n$

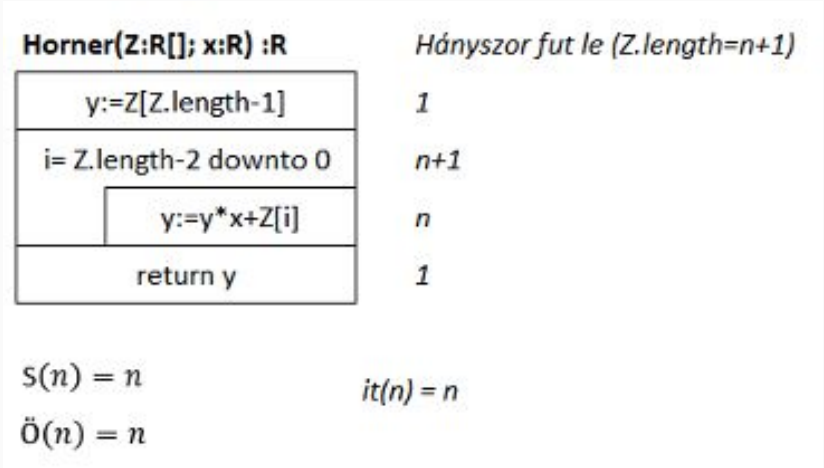
it(n) = n

$x^{i+1}=x^i*x$

Nem dobjuk el a kiszámított x^i értéket, hanem abból számoljuk a következő hatványt.

Ez már lineáris futási idejű (min, max, átlagos)

Horner algoritmus



Átzárójelezés:

$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_2 x^2 + a_1 x + a_0 =$

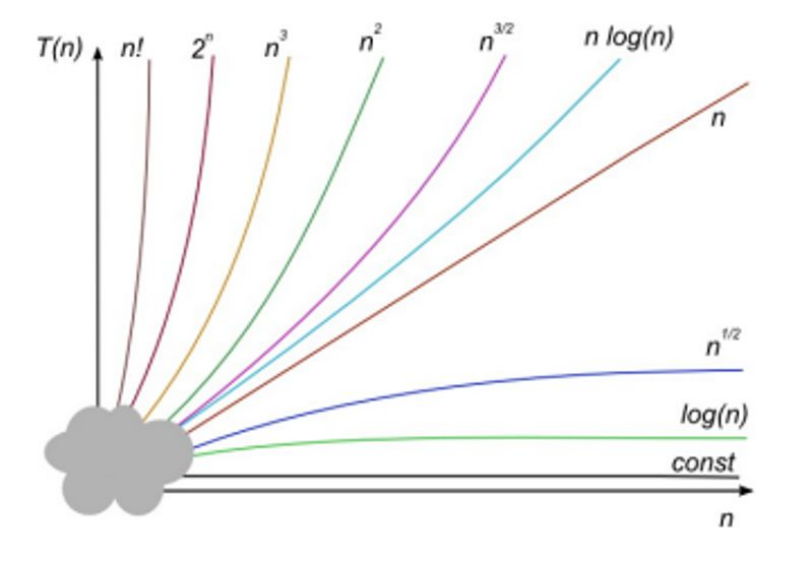
$(a_n x^{n-1} + a_{n-1} x^{n-2} + \ldots + a_2 x + a_1) x + a_0 =$

$((a_n x^{n-2} + a_{n-1} x^{n-3} + \ldots + a_2) x + a_1) x + a_0 =$

..

$(..(a_n x + a_{n-1}) x + \ldots + a_1) x + a_0$

Futási idő nagyságrendek



Futási idő példák

	100	1.000	10.000	1.000.000	1.000.000.000
logn	7	10	14	20	30
n	100	1.000	10.000	1.000.000	1.000.000.000
nlogn	700	10.000	140.000	20.000.000	30mrd
n^2	10.000	1.000.000	100.000.000	1.000.000.000	10^{18}