Adatstruktúrák és algoritmusok gyakorlat

Tartalomjegyzék

Bevezetés

2020 március 23-tól kezdődően a koronavírus miatt Egyetemünkön távoktatás keretében folyik az oktató munka. Ez az oktatási segédanyag az Önök felkészülésében kíván segítséget nyújtani.

A segédanyagot folyamatosan szerkesztem, illetve javítom, heti rendszerességgel töltöm fel a honlapomra.

A feltöltött segédanyag erősen támaszkodik a Házy Attila, Nagy Ferenc Adatstruktúrák és algoritmusok című jegyzetre, ami báki számára hozzáférhető, illetve a saját oldalamra is felraktam. Házy tanár úr honlapján is érdemes körülnézni. Mindenkinek a figyelmébe ajánlom a "Cormen-féle" Új algoritmusok című könyvet is.

Minden hallgatómnak sikeres felkészülést kívánok. Vigyázzanak magukra illetve egymásra.

Glavosits Tamás

0.1. Lebegőpontos számábrázolás

A félév során minden gyakorlaton átpörgetjük az elméleti fogalmakat. Ez jól fog jönni a szigorlaton, viszont ha a későbbiekben a szakmájukból fognak élni, akkor is hasznos lesz.

Ezek a definíciók nem szigorú értelemben vett matematikai definíciók, hanem hétköznapi fogalmakkal körülírjuk, hogy mire gondolunk. Jobb megtanulni a definíciókat, mint IQ-ból rögtönözni, ez utóbbi nem mindig célravezető. A definíciók a Nagy Ferenc és Házy Attila Adtstruktúrák és algoritmusok című könyvből származnak és elég jól! meg vannak fogalmazva. A definíciókat nem kell szóról szóra bemagolni (bár ajánlott), másképpen megfogalmazott, de a lényeget lefedő definíciókat is elfogadunk.

A feladatok megoldása során, fontos, hogy lépésről lépésre definíció szerint számoljunk, ne kapkodjunk és ne akarjuk előre kitalálni az eredményt. A alsó és felső egészrész függvények esetén az sem árt, ha számegyenesen ábrázoljuk a számokat, vagy legalábbis elképzeljük őket a számegyenesen és átgondoljuk, hogy melyek a szám bal és jobb oldali egész szomszédai.

Alsó egészrész függvény: $\lfloor\cdot\rfloor : \mathbb{R} \to \mathbb{Z}$

$$\lfloor x \rfloor := \max\{z \in \mathbb{Z} \mid z \leqslant x\} \qquad (x \in \mathbb{R}),$$

tehát az alsó egészrész függvény minden valós számhoz hozzárendeli a számnál kisebb vagy egyenlő egészek közül a legnagyobbat.

Feladat *Határozza meg a* $\lfloor 7.13 \rfloor$ *és az* $\lfloor -7.13 \rfloor$ *értékét.*
Mo.:

$$\lfloor 7.13 \rfloor = 7,$$
$$\lfloor -7.13 \rfloor = -8.$$

Felső egészrész függvény: $\lceil\cdot\rceil : \mathbb{R} \to \mathbb{Z}$

$$\lceil x \rceil := \max\{z \in \mathbb{Z} \mid x \leqslant z\} \qquad (x \in \mathbb{R}),$$

tehát a felsőegészrész függvény minden valós számhoz hozzárendeli a számnál nagyobb vagy egyenlő egészek közül a legkisebbet.

Feladat *Határozza meg a* $\lceil 7.13 \rceil$ *és* $\lceil -7.13 \rceil$ *értékét.*
Mo.:

$$\lceil 7.13 \rceil = 8,$$
$$\lceil -7.13 \rceil = -7.$$

Kerekítőfüggvény: $\mathrm{round} : \mathbb{R} \to \mathbb{Z}$

$$\mathrm{round}(x) := \lfloor x + 0.5 \rfloor \qquad (x \in \mathbb{R}).$$

Feladat *Határozza meg a* $\mathrm{round}(7.13)$, $\mathrm{round}(7.56)$, $\mathrm{round}(-7.13)$, $\mathrm{round}(-7.56)$ *értékét.*
Mo.:

$$\begin{aligned}\mathrm{round}(7.13) &= \lfloor 7.63 \rfloor = 7,\\ \mathrm{round}(7.56) &= \lfloor 8.06 \rfloor = 8,\\ \mathrm{round}(-7.13) &= \lfloor -6.63 \rfloor = -7,\\ \mathrm{round}(-7.56) &= \lfloor -7.06 \rfloor = -8.\end{aligned}$$

Látható, hogy a round *függvény nem páratlan, ugyanis* $\mathrm{round}(0.5) = \lfloor 1 \rfloor = 1$, *azonban* $\mathrm{round}(-0.5) = \lfloor 0 \rfloor = 0$, *így* $\mathrm{round}(-0.5) \neq -\mathrm{round}(0.5)$.

Törtrészfüggvény: $\{\cdot\} : \mathbb{R} \to [0, 1[$

$$\{x\} := x - \lfloor x \rfloor \qquad (x \in \mathbb{R})$$

Feladat *Határozza meg az* $\{7.13\}$, $\{-7.13\}$ *értékeket.*
Mo.:

$$\begin{aligned}\{7.13\} &= 7.13 - \lfloor 7.13 \rfloor = 7.13 - 7 = 0.13,\\ \{-7.13\} &= -7.13 - \lfloor -7.13 \rfloor = -7.13 - (-8) = 0.87.\end{aligned}$$

A div függvény $\mathrm{div} : \mathbb{R} \times (\mathbb{R} \setminus \{0\}) \to \mathbb{Z}$

$$\mathrm{div}(a, b) := a \text{ div } b := \left\lfloor \frac{a}{b} \right\rfloor \qquad (a \in \mathbb{R}, b \in \mathbb{R} \setminus \{0\}).$$

Feladat *Határozza meg a* $13 \text{ div } 5$, $(-13) \text{ div } 5$, $13 \text{ div } (-5)$, $(-13) \text{ div } (-5)$, *értékét.*
Mo.:

$$\begin{aligned}13 \text{ div } 5 &= \left\lfloor \frac{13}{5} \right\rfloor = \lfloor 2.6 \rfloor = 2,\\ (-13) \text{ div } 5 &= \left\lfloor -\frac{13}{5} \right\rfloor = \lfloor -2.6 \rfloor = -3,\\ 13 \text{ div } (-5) &= \left\lfloor \frac{13}{-5} \right\rfloor = \lfloor -2.6 \rfloor = -3,\\ (-13) \text{ div } (-5) &= \left\lfloor -\frac{-13}{-5} \right\rfloor = \lfloor 2.6 \rfloor = 2.\end{aligned}$$

A mod függvény $\mathrm{mod}(\cdot, \cdot) : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$

$$\mathrm{mod}(a, b) := a \bmod b := \begin{cases} a, & \text{ha } b = 0,\\ a - b \cdot \left\lfloor \frac{a}{b} \right\rfloor, & \text{ha } b \neq 0.\end{cases}$$

Feladat *Számoljuk, ki a* $13 \bmod 5$, $(-13) \bmod 5$, $13 \bmod 5$, $(-13) \bmod (-5)$ *értékeket.*

Mo.:

$$13 \bmod 5 = 13 - 5 \cdot \left\lfloor \frac{13}{5} \right\rfloor = 13 - 5 \cdot 2 = 3,$$

$$(-13) \bmod 5 = -13 - 5 \cdot \left\lfloor -\frac{13}{5} \right\rfloor = -13 - 5 \cdot (-3) = 2,$$

$$13 \bmod (-5) = 13 - (-5) \cdot \left\lfloor -\frac{13}{5} \right\rfloor = 13 - (-5) \cdot (-3) = -2,$$

$$(-13) \bmod (-5) = -13 - (-5) \cdot \left\lfloor \frac{13}{5} \right\rfloor = -13 - (-5) \cdot 2 = -3.$$

A maradékos osztás tétele (Euklidész)

Ha a, $b \in \mathbb{R}$, $b \neq 0$, akkor egyértelműen léteznek olyan q, $r \in \mathbb{Z}$ számok, amelyekkel

$$a = qb + r, \qquad \text{ahol} \qquad a \leqslant r < |b|$$

Bizonyítás $q = a\text{div}b$, $r = a \mod b$ *megfelelőek.*

Elnevezések: az a számot ***osztandó****nak, a b számot* ***osztó****nak, a q számot* ***hányados****nak, az r számot* ***maradék****nak nevezzük.*

A maradékos osztást a Számelméleti algoritmusok című gyakorlaton fogjuk alkalmazni az Euklideszi algoritmus során. Az Euklideszi algoritmus egy olyan algoritmus, amely maradékos osztások segítségével határozza meg két pozitív egész szám legnagyobb közös osztóját.

0.1.1. Az $a \in \mathbb{R}$ szám felírása a $b \in \mathbb{Z}_+$ alapú számrendszerben

A 10-s számrendszert már kisiskolás korunk óta használjuk. Ennek a számrendszernek az a lényege, hogy mindig 10 egységet sorolunk egy nagyobb egységbe, illetve az egynél kisebb számok esetén az egységet mindig 10 egyenlő részre bontjuk. A számok leírásához szükségünk van 10 számjegyre, melyek a következők:

$$0, \quad 1, \quad 2, \quad 3, \quad 4, \quad 5, \quad 6, \quad 7, \quad 8, \quad 9$$

Így bármely szám (például a $113.625_{(10)}$) jegyeinek a 10-es számrendszerben van alaki és helyi értékük. A helyi értékek a 10 megfelelő egész kitevős hatványai.

Alapnak nemcsak a 10-t választhatjuk, hanem bármely más 1-nél nagyobb pozitív egészet. (a 10 feltehetőleg az ujjak száma miatt alakult ki). Az **alapszám**ot a továbbiakban b-vel jelöljük. **Számjegyek:** $0, 1, \ldots, b-1$. Ha a számrendszer alapszáma kisebb 10-nél, akkor használhatjuk a szokásos arab számjegyeket, ha azonban az alapszám nagyobb 10-nél, például 16-os számrendszert használunk, akkor további jegyekre van szükségünk. A 16-os számrendszerben a

$$0, \quad 1, \quad 2, \quad 3, \quad 4, \quad 5, \quad 6, \quad 7, \quad 8, \quad 9 \quad A, \quad B, \quad C, \quad D \quad E, \quad F$$

jegyeket használjuk. Számunkra legfontosabb a 2-es és a 16-os számrendszer.

Egy tetszőleges a pozitív valós szám rögzített b alapszám esetén egyértelműen írható fel

$$a = c_n \dots c_0.d_1 d_2 \dots_{(b)} \doteq c_n b^n + \dots + c_0 b^0 + d_1 b^{-1} + d_2 b^{-2} + \dots .$$

A felírás egyértelműségéhez szükséges, hogy nem engedjük meg, hogy valahonnan kezdve csupa $(b-1)$ jegyek álljanak. Például a 10-es azámrendszerben nem állhatnak valahonnan kezdve csupa 9-esek, a 2-es számrendszerben nem állhatnak valahonnan kezdve csupa 1-esek, illetve a 16-os számrendszerben nem állhatnak valahonnan kezdve csupa F jegyek. Ennek oka, hogy például a 10-es számrendszerben, a mértani sor összegképlete alapján kapjuk, hogy

$$0.99999\dots_{(10)} = \frac{9}{10} \cdot \sum_{k=0}^{\infty} \left(\frac{1}{10}\right)^k = \frac{9}{10} \cdot \frac{1}{1-\frac{1}{10}} = 1.$$

A középiskolában tanultuk, hogy egy szám pontosan akkor racionális, ha tizedestört alakja véges, vagy végtelen szakaszos. Ez a karakterizáció nem csak a tízes, hanem tetszőleges alapú számrendszerben érvényes marad.

Racionális számok karakterizációja *Egy $r \in \mathbb{R}$ szám pontosan akkor racionális, ha tetszőleges b alapú számrendszerben véges, vagy vételen szakaszos b-adikus tört.*

Tehát egy tetszőleges racionális szám b alapú számrendszereben a b alapszámtól függően vagy véges, vagy végtelen szakaszos b-adikus tört.

Véges b-adikus racionális számok karakterizációja *Legyen $q = \frac{r}{s} \in \mathbb{Q}$, ahol $r \in \mathbb{Z}$, $s \in \mathbb{Z}_+$ és $\text{lnko}(r,s) = 1$, azaz az $\frac{r}{s}$ egy redukált tört. A q szám pontosan akkor véges b-adikus tört, ha $s = 1$, vagy s prímtényezős előállításában csak olyan prímek szerepelnek, amelyek a b szám prímtényezős előállításában is szerepelnek.*

Példa *Könnyű látni, hogy az $\frac{1}{2}$ a tízes számrendszerben illetve a kettes számrendszerben véges, de a hármas számrendszerben végtelen szakaszos tört.*

$$\frac{1}{2} = 0.5_{(10)},$$
$$\frac{1}{2} = 0.1_{(2)},$$
$$\frac{1}{2} = 0.1111\dots = 0.\dot{1}_{(3)} .$$

Az utolsó egyenlőség könnyen kijön a mértani sor összegképlete alapján:

$$0.\dot{1}_{(3)} = \sum_{k=1}^{\infty} \left(\frac{1}{3}\right)^k = \frac{1}{3} \sum_{k=0}^{\infty} \left(\frac{1}{3}\right)^k = \frac{1}{3} \cdot \frac{1}{1-\frac{1}{3}} = \frac{1}{3} \cdot \frac{3}{2} = \frac{1}{2}.$$

Most megadjuk azokat az algoritmusokat, amelyek segítségével egy pozitív valós számot fel lehet írni tetszőleges b alapú számrendszerben.

Egy pozitív egész számot úgy írunk fel a b alapú számrendszerben, hogy maradékos osztások sorozatát végezzük. Az osztó a b szám. A maradékot a szám mellé, a hányadost a szám alá írjuk. Mindaddig ismételjük az eljárást, amíg a hányados nem válik nullává. A maradékok sorozata adja a szám jegyeit, a kiolvasás lentről felfelé történik.

Egy 0 és 1 közé eső számot úgy írunk fel a b alapú számrendszerben, hogy szorozzuk b-vel. A szorzat egész részét a szám mellé, a törtrészét a szám alá írjuk. Az eljárást addig ismételjük, amíg a törtrész nem válik nullává. Akkor is leállunk, ha a törtrészek ciklikus ismétlődését tapasztaljuk. Az egészrészek adják a számjegyeket, a kiolvasás fentről lefelé történik.

Írjuk fel az $\frac{1}{3}$-ot kettes alapú számrendszerben.

$\frac{1}{3}$	0
$\frac{2}{3}$	1
$\frac{1}{3}$	0
$\frac{2}{3}$	1
$\frac{1}{3}$	
$\vdots$	

Látható, hogy szakaszos diadikus törtet kapunk, a szakasz hossza 2. Ezt kétféle módon tudjuk jelölni, vagy a szakasz első és utolsó jegye fölé pontot teszünk, vagy a szakaszt felülvonással jelöljük. Tehát

$$\frac{1}{3} = 0.\overline{01}_{(2)}.$$

A kapott eredmény könnyen ellenőrizhető a mértani sor összegképlete alapján.

$$0.\overline{01}_{(2)} = \sum_{k=1}^{\infty} \left(\frac{1}{4}\right)^k = \frac{1}{4} \sum_{k=0}^{\infty} \left(\frac{1}{4}\right)^k = \frac{1}{4} \cdot \frac{1}{1 - \frac{1}{4}} = \frac{1}{4} \cdot \frac{1}{3} = \frac{1}{3}.$$

Feladat *Írjuk fel a 113-at, a 2-es, a 3-as, a 4-es, a 8-as és 16-os számrendszerben. A megoldás kulcsa, hogy a 113-at fel kell írni a kettes számrendszerbe, majd mivel a $2^2 = 4$, $2^3 = 8$, $2^4 = 16$, így a kettes számrendszerbeli jegyeket kettesével, hármasával négyesével csoportosítva megkapjuk a 113 jegyeit a 4, 8, 16 alapú számrendszerben.*

Kettes számrendszerben: ($b = 2$)

113	1
56	0
28	0
14	0
7	1
3	1
1	1
0	

Így kapjuk, hogy $113 = 1110001_{(2)}$.

Hármas számrendszerben: $(b = 3)$

113	2
37	1
12	0
4	1
1	1
0	

Tehát: $113 = 1101_{(3)}$.

A négyes számrendszerben: $(b = 4)$ $113 = 1|11|00|01_{(2)} = 1301_{(4)}$.

A nyolcas számrendszerben: $(b = 8)$ $113 = 1|110|001_{(2)} = 161_{(8)}$.

A tizenhatos számrendszerben : $(b = 16)$ $113 = 111|0001_{(2)} = 71_{(16)}$. *Eredményeinket foglaljuk össze egy táblázatban:*

2	1110001
3	11012
4	1301
8	161
10	113
16	71

A kapott táblázat jól illusztrálja a az alábbi tételt, amely megmondja, hogy egy pozitív egész számnak hány jegye van egy b alapú számrendszerben. Nyilván minél nagyobb az alapszám, annál rövidebb lesz a szám.

Tétel *Egy* $a \in \mathbb{Z}_+$ *szám számjegyeinek száma egy* b *alapú számrendszerben* $\lfloor \log_b(a) \rfloor + 1$.

Példa $\lfloor \log_2(113) \rfloor + 1 = 7$

Feladat *Írjuk fel 2-es számrendszerben a 0.625-öt.*

0.625	1
0.25	0
0.5	1
0	

$0.625_{(10)} = 0.101_{(2)}$.

Lebegőpontos számábrázolás *Az* ***IEEE 754*** *(1985) (Ejtsd: "áj tripl í") szabványt ismertetjük, amely előírja a számok lebegőpontos ábrázolásának módját. Az IEEE egy betűszó, ami az (Institute of Electrical and Electronics Engineers) szavak kezdőbetűiből áll.*

Négyféle ***lebegőpontos számformátum*** *létezik, az* ***egyszeres****, a* ***dupla****, a* ***kiterjesztett*** *és a* ***négyszeres*** *pontosságú. Ezek egymástól csak a komponenseinek bitszámaiban térnek el, ugyanúgy működnek, ezért csak az egyszeres pontosságú számábrázolást ismertetjük.*

Egyszeres pontosság esetén a számot 32 biten (= 4 byte) ábrázoljuk. A lebegőpontos számábrázoláshoz először fel kell írni a számot 2-es számrendszerben. Ha $x \in \mathbb{R}$ ($x \neq 0$) a szám, akkor

$$x = \pm 1.f_{(2)} \cdot 2^e.$$

Ezt az alakot ***normált alak****nak hívjuk.*

s: 1 bit ***előjelbit****. Ha a szám pozitív, akkor s=0, ha negatív, akkor s=1. (Később látni fogjuk, hogy a 0 lehet pozitív is és negatív is.)*

Ezt követően kiszámoljuk az eltolt kitevőt, ami $k = e + b$ ***karakterisztika****, ahol k az* ***eltolt kitevő*** *(kettes számrendszerben), e a* ***valós kitevő****, valamint $b = 127$. Amennyiben az eltolt kitevő nyolcnál kevesebb jegyből áll, akkor az eltolt kitevő elé annyi nullát írunk, hogy összesen nyolc bitet kapjunk. Tehát eddig 9 bitet használtunk el a rendelkezésünkre álló 32-ből.*

f ***mantissza****, ami a maradék 23 biten van tárolva. Mivel az x szám nem lehet 0, normált alakban a diadikus pont előtt mindig áll egy egyes, így ezt az 1-est nem tároljuk, kidobjuk a kukába. Erre az átírás és a visszaírás során egyaránt oda kell figyelni. Az f bitjeit (a 32 db. bit közül a tizediktől kezdve) beírjuk a megfelelő helyre, ha f kevesebb bitből áll, mint 23, akkor a hiányzó biteket nullákkal pótoljuk. A kapott bitsorozatot egy hexadecimális számmá alakítjuk. Ez a hexadecimális szám a x szám egyszeres pontosságú lebegőpontos alakja.*

Érdemes átgondolni, hogy a misztikus 127 miből jön ki, de előtte jöjjön egy kis játék csupa 1-esekből álló diadikus számokkal.

$$\begin{aligned}
&2^1 - 1 = 2 - 1 = 1_{(2)},\\
&2^2 - 1 = 4 - 1 = 3 = 11_{(2)},\\
&2^3 - 1 = 8 - 1 = 7 = 111_{(2)},\\
&\qquad\vdots\\
&2^n - 1 = 11\ldots1_{(2)},\ n \text{ darab egyes a kettes számrendszerben.}
\end{aligned}$$

Mivel az eltolt kitevő 8 biten tárolódik, így

$$2_7 - 1 = 127 = 11\ldots1_{(2)}\ n \text{ darab egyes a kettes számrendszerben},$$

ami 7 darab egyes a kettes számrendszerben. A valós kitevő lehet pozitív is, negatív is, sőt még 0 is, de hozzáadva 127-et, bele tudjuk tolni a pozitív tartományba.

Ha n jegyen tárolnánk az eltolt kitevőt, akkor $2^{n-1} - 1$-et kellene hozzáadnunk a valós kitevőhöz, hogy megkapjuk belőle az eltolt kitevőt.

Az eltolt kitevő legkisebb értéke az 1, a legnagyobb értéke $11\ldots10_{(2)} = 2^8 - 2$. tehát

$$1 \leqslant k = \text{ eltolt kitevő } = e + 127 \leqslant 2^8 - 2,$$

amiből kapjuk, hogy

$$-126 \leqslant e = \text{ valós kitevő } \leqslant 127,$$

így

$$\begin{aligned}
1.000\ldots_{(2)} \cdot 2^{-126} \approx 1.1574 \cdot 10^{-38} &\leqslant |\text{ lebegőpontos szám szám }| \leqslant\\
&\leqslant 1.11\ldots1_{(2)} \cdot 2^{127} = \left(\frac{2^{24}-1}{2^{23}}\right) \cdot 2^{127} = \left(2 - 2^{-23}\right) \cdot 2^{127} \approx\\
&\approx 3.4028 \cdot 10^{38},
\end{aligned}$$

Nézzük meg, hogy hogyan kezeli a szabvány a felülcsordulást.

- *Ha az eltolt kitevő 8 db egyes, a mantissza 23 db 0, akkor az előjelbittől függően $+\infty$-t vagy $-\infty$-t kapunk. Ezekkel az értékekkel bizonyos bizonyos aritmetikai műveleteket még el tudunk végezni.*

- *Ha az eltolt kitevő 8 db egyes de a mantissza nem csak nullákból áll, akkor nem számot kapunk (NaN).*

Ha a legkisebb (pozitív) lebegőpontos számból levonunk 2^{-149}-et, akkor megkapjuk a legnagyobb olyan pozitív számot, amely olyan kicsi, hogy lebegőpontosan már nem ábrázolható. Ezek az úgynevezett ***denormált számok****. Ezeket úgy ábrázoljuk, hogy az előjelkitevő lehet 0, vagy 1; az eltolt kitevő csupa 0, (ami azonban az 1 eltolt kitevőnek felel meg, most azonban a -126 valós kitevőt jelöli), a mantissza (f), azaz a diadikus pont utáni rész egy tetszőleges 23 bitből álló bitsorozat, amely azonban nem lehet csupa 0. Így*

$$2^{-149} \approx 1.4012\ldots 10^{-45} \leqslant |\ \textit{denormált szám}\ | \leqslant$$
$$\leqslant (1-2^{-23})\cdot 2^{-126} \approx 1.1754\cdot 10^{-38}.$$

A 2^{-149}-nél kisebb abszolútértékű számokat már nem tudjuk megkülönböztetni a 0-tól. Ezeknek a számoknak az ábrázolása: előjelbit és utána csupa 0. Tehát van $+0$ és -0. Úgy tudjuk a 0-t megkülönböztetni a denormált számoktól, hogy a denormált számok mantissza része nem lehet csupa 0.

A fentieket összefoglaljuk egy táblázatban:

	előjelbit (s) *1 bit*	*eltolt kitevő (k)* *8 bit*	*mantissza (f)* *23 bit*
NaN nem szám	0/1	11111111	$\neq 0\ldots 0$
$\pm$*végtelen*	0/1	11111111	$0\ldots 0$
lebegőpontos szám	0/1	00000001 *és* 11111110 *között*	*bármi*
denormált szám	0/1	00000000	$\neq 0\ldots 0$
nulla	0/1	00000000	$0\ldots 0$

Most megfogalmazunk néhány magától értetődő szabályt. Jelöljön x egy pozitív lebegőpontos számot. Ekkor

$$\pm\infty \pm x = \pm\infty,$$
$$\pm\infty \cdot x = \pm\infty, \qquad \pm\infty\cdot(-x) = \mp\infty,$$
$$\pm\infty / x = \pm\infty, \qquad \pm\infty/(-x) = \mp\infty,$$
$$x/(\pm\infty) = \pm 0, \qquad (-x)/(\pm\infty) = \mp 0,$$

A $0/0$, $(+\infty)-(+\infty)$ *stb.* NaN-*t eredményez. Jöjjön még egy összefoglaló táblázat.*

	s	k	f	$\sum$
short real (egyszeres)	1	8	23	32
long real (dupla)	1	11	52	64
temporary real (kiterjesztett)	1	15	64	80
quad real (négyszeres)	1	15	112	128

Feladat *Ábrázoljuk a 113.625-öt levegőpontosan egyszeres pontossággal (4 byte=32 bit). Mo:.*

1. *lépés: Átírás 2-es számrendszerbe.*

   $113.625 = 1110001,101_{(2)}$

2. *lépés: Átírás normált alakra.*

   $1110001.101_{(2)} = 1.110|0011|01_{(2)} \cdot 2^6$

3. *lépés: Az eltolt kitevő kiszámítása. A valós kitevő:* $e = 6$*, az eltolt kitevő:* $k = e + b = 6 + 127 = 133 = 100|0010|1_{(2)}$.

4. *lépés: Elhelyezés a megfelelő helyre*

4	2	E	3	4	0	0	0
0100	0010	1110	0011	0100	0000	0000	0000

5. *lépés: visszatérés a hexadecimális számmal. A lebegőpontos alak:* $42E34000$.

A következő táblázat segít az inverz feladat megoldásában.

0	0000	4	0100	8	1000	C	1100
1	0001	5	0101	9	1001	D	1101
2	0010	6	0110	A	1010	E	1110
3	0011	7	0111	B	1011	F	1111

Inverz feladat *Melyik szám lebegőpontos alakja a* $C28C0000$.

C	2	8	C	0	0	0	0
1100	0010	1000	1100	0000	0000	0000	0000

$s = 1$, *így a szám negatív.*

$k = \overset{7)}{1}\ 0000\ \overset{2)}{1}\ 0\ \overset{0)}{1}_{(2)} = 2^7 + 2^2 + 1 = 133$ *az eltolt kitevő*

$k = k - 127 = 133 - 127 = 6$ *a valós kitevő*

A normált alak: $-1.00011_{(2)} \cdot 2^6 = -\ \overset{6)}{1}\ 000\ \overset{2)1)}{1}\ 1\ 0 = -(2^6 + 2^2 + 2^1) = -(64 + 4 + 2) = -70$

Feladat *Ábrázoljuk lebegőpontosan a 117.875-öt.*

1.

117	1
58	0
29	1
14	0
7	1
3	1
1	1
0	

$117 = 1110101_{(2)}$

2.

0.875	1
0.75	1
0.5	1
0	

$0.875 = 0.111_{(2)}$

$117.875 = 1110101.111 = 1.110|1011|11 \cdot 2^6$

valós kitevő: $e = 6$

eltolt kitevő: $k = e + 127 = 6 + 127 = 5 + 128 = 100|0010|1_{(2)}$

0100	0010	1110	1011	1100	0000	0000	0000
4	2	E	B	c	0	0	0

Feladat *Ábrázoljuk lebegőpontosan a 0.1-et.*

0.1	0
0.2	0
0.4	0
0.8	1
0.6	1
0.2	

$0.1 = 0.0\overline{0011}_{(2)} = 1.\overline{1001}_{(2)} \cdot 2^{-4}$

valós kitevő: $e = -4$

eltolt kitevő: $k = e + 127 = -4 + 127 = 123 = 11|1101|1_{(2)}$

123	1
61	1
30	0
15	1
7	1
3	1
1	1
0	

$f = \overline{1001} = 100|\overline{1100}$

0011	1101	1100	1100	1100	1100	1100	1100
3	D	C	C	C	C	C	C

Feladat *Ábrázoljuk lebegőpontosan a 0.2-őt.*
$0.2 = 2 \cdot 0.1$, *így csak a kitevő változik, 1-el nő.*

0011	1101	0100	1100	1100	1100	1100	1100
3	E	4	C	C	C	C	C

0.2. Algoritmus, pszeudokód, növekedési rendek, ordó szimbolika

Absztrakt adat *valamilyen halmaznak az eleme*

Absztrakt adattípus *egy leírás, amely megadja az absztrakt adatok halmazát és a rajtuk elvégezhető műveleteket, nem törődve azok konkrét (gépi) realizálásával.*

A következő két fogalom azért fontos, mert szerepelnek a tantárgy nevében, és rém rossz lenne fél évig olyannal foglalkozni, amiről nem tudjuk, hogy mi.

Adatstruktúra *Adatstruktúrának nevezzük az absztrakt adattípus konkrét megjelenési formáját.*

Algoritmus *egy meghatározott számítási eljárás a számítási probléma megoldási eszköze.*

Pszeudókód *Az algoritmus lejegyzésének egy módja.*

A leírtakból kitűnik, hogy a félév során algoritmusokat fogunk ismertetni pszeudokódjuk segítségével. Az általunk használt pszeudokód tökéletesen megfelel a célnak és néhány kisebb eltéréstől eltekintve hasonlít más egyetemeken használt pszeudokódokhoz.

Intellektuális kihívás megismerkedni a különféle problémákra adott különféle algoritmusokkal. Van közöttük néhány, amire (kis nagyképűséggel) azt is mondhatnánk, hogy ezt akár mi is kitalálhattuk volna, a többségük azonban kicsit sem egyszerű, azonban zseniális. Ugyanannak a problémának a megoldására többféle módszer is adható, tehát nincs univerzálisan jó megoldás (ahogyan univerzálisan jó kocsi sincs...). Tanulmányaink abba is bepillantást engednek, hogy milyen szituációban melyik módszert kell alkalmazni.

A pszeudokóddal adott algoritmust kis méretű inputokra kézzel futtatni jellempróbáló és lélekölő feladat. De talán ez az egyedüli módja annak, hogy egy adott algoritmus működését megértsük. Oktatói oldalról így tudjuk mérni, hogy egy adott hallgató valóban érti-e az algoritmus működését, vagy csak érteni véli.

Egy pszeudokóddal helyesen leírt algoritmus csuklóból való kódolása általunk ismert programnyelven szintén fontos készség, azonban ez utóbbihoz nem szükséges ismerni az algoritmus működését és úgy gondoljuk, hogy ennek a készségnek a fejlesztése, beleértve a felvetődő számtalan technikai részletkérdés helyes megoldását nem ennek a tárgynak a feladata.

WHILE DO ciklus *egy előltesztelő ciklus, a ciklusblokk akkor lesz elvégezve, ha a feltétel igaz, és hamisnál lép ki a ciklusból.*

REPEAT UNTIL ciklus *egy hátultesztelő ciklus. A ciklusblokk mindaddig elvégzésre kerül, amíg a feltétel hamis, azaz akkor enged ki a ciklusból, ha a feltétel igazzá válik.*

A következő példa tanulságos, de kihagyható.

Példa *Írjunk algoritmust, amely egy*

$$p(x) = p_n x^n + p_{n-1}x^{n-1} + \cdots + p_1 x + p_0$$

n-ed fokú polinom helyettesítési értékét kiszámolja valamilyen $\alpha \in \mathbb{R}$ helyen.

Itt most egy kicsit nagyvonalúskodunk a fokszám fogalmával, ugyanis ha egy polinom konstans, de nem azonosan 0 (azaz $p(x) = a_0$, de $a_0 \neq 0$), akkor azt mondjuk, hogy a polinom fokszáma 0, míg az azonosan 0 polinomra (azaz $p(x) = a_0$ és $a_0 = 0$) esetén azt mondjuk, hogy a fokszáma $-\infty$. Mi most nem fogunk különbséget tenni, a konstans, de nem azonosan 0, illetve az azonosan 0 polinomok között, mindkétfélére azt mondjuk, hogy a fokszámuk 0.

A másik eltérés a későbbiekhez képest az, hogy kényelmi okokból, kizárólag ebben a példában egy tömb indexelését a 0-tól kezdjük, azonban a későbbiekben az indexelést 1-től fogjuk kezdeni. Ha P egy tömb, akkor a Hossz[P] *a P tömbben tárolt elemek számát fogja jelölni, azaz ha a fenti polinom együtthatóit tároljuk a P tömbben, akkor*

$$P[n] = p_n, \quad P[n-1] = p_{n-1}, \quad , \ldots , \quad P[0] = p_0.$$

Könnyű látni, hogyha a $p(x)$ egy n-ed fokú polinom, akkor Hossz[P] $= n + 1$.

Most megadunk négyféle megoldást a $p(x)$ polinom α helyen felvett helyettesítési értékének, azaz a $p(\alpha)$ kiszámítására. Mind a négy esetben:

Input: *P egy tömb, amely a polinom együtthatóit tartalmazza; α a hely, ahol a polinom helyettesítési értékét keressük;*

Output: *y a keresett helyettesítési érték, azaz a $p(\alpha)$ értéke.*

Mo.1. *Favágó módszer. Elég egyszerű, bár nem biztos, hogy a legjobb módszer.*

	Poli I.(P, α, y)
1.	$y \leftarrow 0$,
2.	*FOR* $i \leftarrow$ (Hossz$[P] - 1$) *DOWNTO* 0 *DO*
3.	$\quad x \leftarrow 1$,
4.	$\quad$ *FOR* $j \leftarrow$ 1 *TO* i
5.	$\quad\quad x \leftarrow \alpha x$,
6.	$\quad y \leftarrow y + P[i]x$,
7.	*RETURN(y).*

Az algoritmus műveletigénye:

$$(1 + 2 + \cdots + n) + 2n = \frac{1}{2}(n^2 + 5n),$$

tehát az algoritmus négyzetes futásidejű (a polinom fokszámának a függvényében). Kicsit lehet finomítani az algoritmuson:

Mo.2.

	Poli II.(P, α, y)
1.	$z \leftarrow 1$,
2.	$y \leftarrow P[0]$,
3.	*For* $i \leftarrow 1$ *TO* (Hossz$[P] - 1$) *DO*
4.	$\quad z \leftarrow \alpha z$,
5.	$\quad y \leftarrow y + P[i]z$,
6.	*RETURN(y).*

Az algoritmus műveletigénye: $3n$, *tehát ez egy lineáris időigényű algoritmus.*
Mo.3. Horner elrendezés:

Motiváció az algoritmushoz: Legyen például $p(x)$ *egy harmadfokú polinom.*

$$\begin{aligned} p(\alpha) &= p_3\alpha^3 + p_2\alpha^2 + p_1\alpha + p_0 = \\ &= (p_3\alpha^2 + p_2\alpha + p_1)\alpha + p_0 = \\ &= ((p_3\alpha + p_2)\alpha + p_1)\alpha + p_0 = \\ &= (((0 \cdot \alpha + p_3)\alpha + p_2)\alpha + p_1)\alpha + p_0 \end{aligned}$$

Az algoritmus:

	Horner(P, α, y)
1.	$y \leftarrow 0$,
2.	*FOR* $i \leftarrow$ (Hossz$[P] - 1$) *DOWNTO* 0 *DO*
3.	$y \leftarrow y\alpha + P[i]$,
4.	*RETURN(y).*

Legyen $p(x)$ *egy harmadfokú polinom és futtassuk futtassuk kézzel az algoritmust:*

$$\begin{aligned} &i = 3 \quad y \leftarrow y \cdot \alpha + P[3], (\text{ekkor } y = p_3); \\ &i = 2 \quad y \leftarrow y \cdot \alpha + P[2], (\text{ekkor } y = p_3\alpha + p_2; \\ &i = 1 \quad y \leftarrow y \cdot \alpha + P[1], (\text{ekkor } y = (p_3\alpha + p_2)\alpha + p_1 = p_3\alpha^2 + p_2\alpha + p_1); \\ &i = 0 \quad y \leftarrow y \cdot \alpha + P[0], (\text{ekkor } y = (p_3\alpha^2 + p_2\alpha + p_1)\alpha + p_0 = \\ &\qquad\qquad\qquad\qquad = p_3\alpha^3 + p_2\alpha^2 + p_1\alpha + p_0. \end{aligned}$$

Az algoritmus műveletigénye: $2n + 2$.

Végül megadjuk a Horner elrendezés rekurzív változatát, amely semmivel sem jobb az iteratív változatnál.
Mo.4.

	RH(P, k, α, y)
1.	*IF* (Hossz$[P] - k) = 1$ *THEN*
2.	$y = P[k]$,
3.	*ELSE*
4.	INC(k),
5.	RP(P, k, y),
6.	$y \leftarrow \alpha y + P[k]$,
6.	*RETURN(y).*

$k = 0$ *értékkel kell hívni.*

Oszd meg és uralkodj elv. *Ez a stratégia a kiinduló problémát kisebb méretű független részproblémákra bontja, amelyeket rekurzívan old meg. A kisebb méretű részproblémák megoldásait egyesítve kapja meg az eredeti probléma megoldását. (Pl. Euklideszi algoritmus.)*

0.2.1. Idő és tárbonyolultság

Definíció: *Legyen A egy algoritmus. Jelölje*

- *D az algoritmus inputjainak a halmazát;*
- *$|x|$ az $x \in D$ input méretét az, ami az x bitjeinek a száma;*
- *$t_A(x)$ az x input időigényét (Pl. a műveletek számát);*
- *$s_A(x)$ az x input tárigényét.*

Az algoritmus idő és tárbonyolultsága:

- ***Az algoritmus időbonyolultsága:***

$$T_A(n) = \sup\{t_A(x) \mid x \in D, |x| \leqslant n\} \quad (n \in \mathbb{Z}_+).$$

- ***Az algoritmus tárbonyolultsága:***

$$S_A(n) = \sup\{s_A(x) \mid x \in D, |x| \leqslant n\} \quad (n \in \mathbb{Z}_+).$$

A továbbiakban az időbonyolultságot vizsgáljuk, a tárbonyolultságot nem.

Növekedési függvények *A T_A, $S_A : \mathbb{Z}_+ \to \mathbb{Z}_+$ függvények monoton növekvő függvények. Az ilyen függvényeket* ***növekedési függvény****eknek nevezzük.*

Feladat *Van egy gép, amely 1 s alatt 10^6 műveletet végez. Könnyű kiszámolni, hogy adott időegység alatt hány műveletet végez el a gép.*

idő	1*mp*	1 *perc*	1 *óra*	1 *nap*	1 *év*	100 *év*
műveletek száma	10^6	$6 \cdot 10^7$	$3.6 \cdot 10^9$	$8.64 \cdot 10^{10}$	$3.1563 \cdot 10^{13}$	$3.1563 \cdot 10^{15}$

Példa *$T(n)$ ismeretében adott időegység alatt mekkora inputmérettel végez a gép?*

Például: $T(n) = n^2$ és időegység 1 nap.

Mo.: $n^2 = T(n) = 8.64 \cdot 10^{10}$, így kapjuk, hogy $n = \sqrt{8.64 \cdot 10^{10}} = 2.9394 \cdot 10^5$, tehát 1 nap alatt $2.9394 \cdot 10^5$ inputmérettel végez a gép.

Foglaljuk táblázatba, hogy 1 másodperc, 1 perc, 1 óra, 1 nap, 1 év, 100 év alatt mekkora inputmérettel végez a gép.

műv. sz.	10^6	$6 \cdot 10^7$	$3.6 \cdot 10^9$	$8.6 \cdot 10^{10}$	$3.1 \cdot 10^{13}$	$3.1 \cdot 10^{15}$
$T(n)$	1*mp*	1 *perc*	1 *óra*	1 *nap*	1 *év*	100 *év*
$\log(n)$	$10^{3 \cdot 10^5}$	-	-	-	-	-
$\sqrt{n}$	10^{12}	$3.6 \cdot 10^{15}$	$1.2 \cdot 10^{19}$	$7.4 \cdot 10^{21}$	$9.9 \cdot 10^{26}$	$9.9 \cdot 10^{30}$
n	10^6	$6 \cdot 10^7$	$3.6 \cdot 10^9$	$8.6 \cdot 10^{10}$	$3.1 \cdot 10^{13}$	$3.1 \cdot 10^{15}$
$n \log(n)$	62746	$2.8 \cdot 10^6$	$1.3 \cdot 10^8$	$2.7 \cdot 10^9$	$7.9 \cdot 10^{11}$	$6.8 \cdot 10^{13}$
n^2	1000	7746	60000	$2.9 \cdot 10^5$	$5.6 \cdot 10^6$	$5.6 \cdot 10^7$
n^3	100	391	1532	4420	31603	$1.4 \cdot 10^5$
2^n	19	25	31	36	44	51
$n!$	9	11	12	13	16	18

0.2.2. Növekedési rendek, ordó szimbolika

Legyenek $f(n)$, $g(n)$ növekedési függvények. Például: $f(n)$ egy vizsgált algoritmus időigénye, $g(n)$ valamilyen tipikus növekedési függvény. Szeretnénk megmutatni, hogy az $f(n)$ és $g(n)$ függvények közül melyik nő gyorsabban.

A fenti táblázat mutatja, hogy nem mindegy, hogy milyen algoritmust használunk, például egy logaritmikus időbonyolultságú algoritmus 1 másodperc alatt $10^{3\cdot 10^5}$ inputmérettel végez, míg egy $n!$ időbonyolultságú algoritmus 100 év alatt is csak 18 inputmérettel végez. Mindkét időbonyolultságú algoritmusra könnyű példát mondani.

A logaritmikus keresés logaritmikus idejű. A logaritmikus keresés lényege, hogy van egy n elemű tömbünk, amelyben számok vannak tárolva, azonban a tömb rendezett. Arra vagyunk kíváncsiak, hogy egy adott szám benne van-e a tömbben, illetve, ha benne van, akkor mi az indexe. Erre a problémára nagyon szép logaritmikus időbonyolultságú algoritmus írható.

Ha azonban n elem összes lehetséges sorrendjét akarjuk előállítani, akkor 19 elem esetén ne akarjuk megoldani a problémát egy olyan gépen, amely másodpercenként 10^6 műveletet végez, ha csak nincs rá legalább 100 évünk, hogy a futási időt kivárjuk.

Ha egy n elemű halmaz összes részhalmazát akarjuk megkeresni, akkor - mivel az n elemű halmaznak 2^n részhalmaza van - ezért az algoritmus legalább 2^n időbonyolultságú. Ilyen időbonyolultságú algoritmust könnyű írni, ugyanis, ha elszámolunk a kettes számrendszerben $00\ldots0$-tól $11\ldots1 = 2^n - 1$-ig, akkor legyártottuk az összes részhalmaz karakterisztikus függvényeit, például, ha az $\{A, B, C, D, E, F\}$ halmaz összes részhalmazát keressük, akkor a 001011 bitsorozat a $\{C, E, F\}$ részhalmazt állítja elő, ugyanis

A	B	C	D	E	F
0	0	1	0	1	1

Az általunk vizsgált ordó szimbolika a futási időket aszimptotikusan (tehát határértékben) vizsgálja, ami talán úgy érthető meg első közelítésben, hogy a kapott eredmények "nagy n-ekre" igazak. Kis n-ekre a futási időbe még nagyon bele tud szólni az n_0 küszöb, ami alatt nem tudjuk, hogy hogyan viselkedik az algoritmus, illetve a c_1 és c_2 konstansok.

A pszeudokóddal leírt algoritmusok esetén könnyű futási időt számolni, hiszen minden lépéshez odaírjuk az elvégzendő műveletek számát, majd a kapott számokat összeadjuk.

Rekurzív módon írt algoritmus esetén $T(n)$-re rekurzív összefüggést kapunk. Ha elég ügyesek vagyunk, akkor fel tudjuk oldani a rekurziót, azonban van egy matematikai módszer, a **Mester tétel**, amelynek alkalmazásával mechanikusan számolható, hogy egy adott rekurzív módon definiált növekedési függvény melyik függvényosztályba tartozik. A Mester tétellel és alkalmazásaival a következő gyakorlaton ismerkedünk meg.

A matematikai definíciók során használni fogunk bizonyos matematikai jeleket (kvantorok), melyek a következők:

$$\forall = \text{ minden}, \qquad \exists = \text{ létezik.}$$

A következő jelöléseket használjuk:

$$f(n) = \mathcal{O}(g(n)), \qquad f(n) = \Omega(g(n)), \qquad f(n) = o(g(n)), \qquad f(n) = \omega(g(n)), \qquad f(n) = \Theta(g(n)).$$

Ezeket az alábbi módon olvassuk ki "$\mathcal{O}$ =nagy ordó", "Ω =nagy omega", "o =kis ordó", "ω=kis omega".

Az egyenlőségjel használata egy kicsit suta, mivel rögzített $g(n)$ növekedési függvény esetén a $\mathcal{O}(g(n))$, $\Omega(g(n))$, $o(g(n))$, $\omega(g(n))$, $\mathcal{O}(g(n))$, $\Theta(g(n))$ nem függvényeket, hanem függvényosztályokat jelöl. Sajnos még azt sem mondhatjuk, hogy $f(n)$ abba a függvényosztályba tartozik, amelybe $g(n)$ ugyanis

$$g(n) \neq o(g(n)), \qquad \text{és} \qquad g(n) \neq \omega(g(n))$$

Azt is érdemes észrevenni, hogy a $\mathcal{O}$-nak van kistestvére, a o és a Ω-nak is van kistestvére, a ω, azonban a Θ-nak nincs kistestvére. (Hogy miért nincs, azt látni fogjuk.)

Most következzenek a formális definíciók:

$\boxed{f(n) = \mathcal{O}(g(n))}$ $\exists(c > 0)\exists(n_0 \in \mathbb{Z}_+)\forall(n \geqslant n_0) : f(n) \leqslant cg(n)$ ("$f(n)$ lassabban nő, mint a $g(n)$".)	$\boxed{f(n) = \Omega(g(n))}$ $\exists(c > 0)\exists(n_0 \in \mathbb{Z}_+)\forall(n \geqslant n_0) : cg(n) \leqslant f(n)$ ("$f(n)$ gyorsabban nő, mint a $g(n)$".)
$\boxed{f(n) = o(g(n))}$ $\forall(c > 0)\exists(n_0 \in \mathbb{Z}_+)\forall(n \geqslant n_0) : f(n) \leqslant cg(n)$ ("$f(n)$ szig. lassabban nő, mint a $g(n)$".)	$\boxed{f(n) = \omega(g(n))}$ $\forall(c > 0)\exists(n_0 \in \mathbb{Z}_+)\forall(n \geqslant n_0) : cg(n) \leqslant f(n)$ ("$f(n)$ szig. gyorsabban nő, mint a $g(n)$".)

illetve $f(n) = \Theta(g(n))$ akkor és csak akkor, ha

$$\exists(c_1 > 0)\exists(c_2 > 0)\exists(n_0 \in \mathbb{Z}_+)\forall(n \geqslant n_0) : f(n) \leqslant c_1 g(n) \text{ és } c_2 g(n) \leqslant f(n).$$

Most megfogalmazzuk a fenti definíciók átfogalmazásait az analízis nyelvén, ehhez azonban szükségük lesz a légyegében teljesülő tulajdonság fogalmára. Egy sorozatokra vonatkozó **tulajdonság lényegében teljesül**, ha valamely küszöbindextől teljesül.

Átfogalmazás *Legyenek $f(n)$ és $g(n)$ növekedési függvények. Az $\frac{f(n)}{g(n)}$ hányados viselkedését kell figyelni.*

1. *$f(n) = \mathcal{O}(g(n))$ akkor és csak akkor, ha $\frac{f(n)}{g(n)}$ sorozat lényegében felülről korlátos pozitív felső korláttal.*

2. *$f(n) = \Omega(g(n))$ akkor és csak akkor, ha az $\frac{f(n)}{g(n)}$ sorozat lényegében alulról korlátos pozitív alsó korláttal.*

3. *$f(n) = \Theta(g(n))$ akkor és csak akkor, ha az $\frac{f(n)}{g(n)}$ sorozat lényegében korlátos pozitív alsó és felső korláttal.*

4. *$f(n) = \Theta(g(n))$ akkor és csak akkor, ha $f(n) = \mathcal{O}(g(n))$ és $f(n) = \Omega(g(n))$.*

5. *$f(n) = o(g(n))$ akkor és csak akkor, ha $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.*

6. *$f(n) = \omega(g(n))$ akkor és csak akkor, ha $\lim_{n\to\infty} \frac{f(n)}{g(n)} = +\infty$.*

A leírtakból látszik, hogy a Θ-nak miért nincs kistestvére. Most megadunk néhány elégséges feltételt.

1. *Ha az* $f(n) = o(g(n))$, *akkor* $f(n) = \mathcal{O}(g(n))$.

2. $f(n) = \omega(g(n))$, *akkor* $f(n) = \Omega(g(n))$.

Talán érdemes megjegyezni, hogyha $f(n) = \Theta(g(n))$ és α, β pozitív valós számok, akkor $\alpha f(n) + \beta = \Theta(g(n))$. Analóg összefüggések fogalmazhatók meg a többi függvényosztállyal kapcsolatban is.

Könnyű látni, hogy az $f(n) = \Theta(g(n))$ egy ekvivalenciarelációt határoz meg a növekedési függvények halmazán. A kapott ekvivalenciareláció hatására a növekedési függvények halmaza ekvivalenciaosztályokra esik szét. A kapott ekvivalencia osztályokat **növekedési rend**eknek nevezzük.

Nevezetes növekedési rendek:

Konstans	$f(n) = \Theta(1)$
Lineáris	$f(n) = \Theta(n)$
Kvázilineáris	$f(n) = \Theta(n \log(n))$
Négyzetes	$f(n) = \Theta(n^2)$
Köbös	$f(n) = \Theta(n^3)$
Polinomiális	$f(n) = \Theta(n^k)$ *valamely* $k \in \mathbb{R}_+$ *esetén*
Logaritmikus	$f(n) = \Theta(\log(n))$
Exponenciális	$f(n) = \Theta(a^n)$ *valamely* $a \in \mathbb{R}_+$ $a > 1$ *esetén*

Feladatok *Igazoljuk, hogy*

1. $\frac{n^2}{2} - 3n = \Theta(n^2)$,

2. $3 \cdot 2^{n-1} = \Theta(2^n)$,

3. $\log(3n^2 + 2n) = \Theta(\log(n))$.

Megoldás

1. *Könnyű látni, hogy*

$$\frac{f(n)}{g(n)} = \frac{\frac{n^2}{2} - 3n}{n^2} \to \frac{1}{2} > 0.$$

Az alkalmas c_1 *és* c_2 *konstansok konstansokat úgy keressük meg, hogy a határértéktől kicsit megyünk jobbra, kicsit megyünk balra így kapjuk, hogy a* $c_1 = 0.4$, $c_2 = 0.6$ *választás megfelelő lesz.*

A küszöbindexek megkereséséhez egyszerű egyenlőtlenségeket kell megoldanunk.

$$0.4 \leqslant \frac{\frac{n^2}{2} - 3n}{n^2} \iff 0.4n^2 \leqslant \frac{n^2}{2} - 3n \iff$$
$$\iff 3n \leqslant 0.1n^2 \iff 3 \leqslant 0.1n \iff 30 \leqslant n,$$

illetve

$$\frac{\frac{n^2}{2} - 3n}{n^2} \leqslant 0.6 \iff \frac{n^2}{2} - 3n \leqslant 0.6n^2 \iff$$
$$\iff -3n \leqslant 0.1n^2 \iff -3 \leqslant 0.1n,$$

az utolsó egyenlőtlenség azonban minden $n \in \mathbb{Z}_+$ esetén teljesük. Így a $c_1 = 0.4$, $c_2 = 0.6$, $n_0 = 30$ választás megfelelő.

2. *Könnyű látni, hogy*

$$\frac{f(n)}{g(n)} = \frac{3 \cdot 2^{n-1}}{2^n} = \frac{3}{2} \to \frac{3}{2} > 0.$$

Így $3 \cdot 2^{n-1} = \Theta(2^n)$. A konstansok megkeresését nem részletezzük.

3. *Heurisztikus megoldás:* $\log(3n^2) = \log(3) + 2\log(n) = \Theta(\log(n))$.

Más úton is el lehet jutni a megoldáshoz. Elegendő belátni, hogy a $\frac{\log(3n^2+2n)}{\log(n)}$ sorozat pozitív határértékhez konvergál. Ehhez elegendő vizsgálni a $\lim_{x\to\infty} \frac{\log(3x^2+2x)}{\log(x)}$ függvényhatárértéket. A L'Hospital szabály alkalmazásával kapjuk, hogy

$$\lim_{x\to\infty} \frac{\log(3x^2+2x)}{\log(x)} = \lim_{x\to\infty} \frac{\frac{1}{\ln(2)}\ln(3x^3+2x)}{\frac{1}{\ln(2)}\ln(x)} =$$
$$= \lim_{x\to\infty} \frac{\frac{1}{3x^2+2x} \cdot (6x+2)}{\frac{1}{x}} = \lim_{x\to\infty} \frac{6x^2+2x}{3x^2+2x} = 2 > 0,$$

azaz teljesül az állítás.

További feladatok *Igazoljuk, hogy*

4. $6n^3 + 2n = \Omega(n^2)$,

5. $2n^2 + 3n = o(n^3)$,

6. $2^{n+1} = \Omega(2^n)$,

7. $2^n = o(3^n)$,

8. $n^k = o(2^n)$,

9. $2^n = o(n!)$.

A bizonyítások triviálisak az elemi analízis megfelelő állításai alapján, azonban a feladatok tanulságosak.

0.3. Rekurzív módon adott sorozatok, Mester tétel és alkalmazása, Számelméleti algoritmusok I

A Fibonacci sorozat *Leonardo Pisano (a pisai Leonardo), ismertebb nevén Fibonacci (Bonacci fia) (1170(?)-1240), olasz származású matematikus, tőle származik az úgynevezett Fibonacci sorozat.*

Fibonacci a középkor idején élt és alkotott. A középkor időben körülbelül az V. századtól a XV. századig tart, földrajzilag Európára és a Mediteránium térségére vonatkozik. Európában a tudományok hanyatlása jellemzi. Itt mutatkozik meg az arabok (Észak-Afrika) szerepe, akik átmentették az ókori tudományok (például a görög matematika) eredményeit, azonban nemcsak átmentették, hanem bővítették és tovább is fejlesztették ezeket az eredményeket.

Ebben a korszakban élt Fibonacci. Algírban tanulta meg a matematika alapjait, szerepe volt a középkori arab matematika, illetve az arab számjegyek használatának Európába való közvetítésében.

Fibonaccitól származik az alábbi feladat: Hány pár nyúl származhat egy évben egyetlen pártól, ha minden pár havonta egy új párnak ad életet, minden nyúlpár a második hónaptól lesz tenyészképes és egy nyúlpár sem pusztul el.

A feladat messze meghaladta a korát. Ugyanis Fibonacci feladatában egy rekurzív módon definiált sorozat 12. elemének a kiszámítása a cél.

A Fibonacci sorozat egy másodfokú rekurzió. Ma már ismerjük az n-edfokú lineáris rekurzió explicit alakjának a felírását is.

A Fibonacci sorozat ezen túlmenően a mai napig fontos, ugyanis a természet maga produkál Fibonacci sorozatot, illetve találkozhatunk a Fibonacci sorozattal a természettudományokban és különféle művészeti ágakban (zene, építészet, festészet) is.

Megoldás *Jelölje $F(n)$ az n-edik hónapban a nyúlpárok számát. Ekkor a Fibonacci sorozat eleget tesz az*

$$F(1) = 1, \qquad F(2) = 1$$
$$F(n+2) = F(n) + F(n+1) \qquad (n \geqslant 1)$$

kezdeti érték és rekurzió feltételeknek. A rekurzió alapján könnyű kiszámolni a sorozat alábbi tagjait.

n	0	1	2	3	4	5	6	7	8	9	10	11	12
$F(n)$	0	1	1	2	3	5	8	13	21	34	55	89	144

Tehát válaszolva Fibonacci kérdésére 144 pár nyulunk lesz. A rekurzív definíció felhasználásával az F_{100} kézzel történő kiszámolása már egy kicsit nehezebb.

Az alábbi Binet formula mutatja a rekurzió feloldását, azaz az explicit alakot.

A rekurzió feloldása *(Binet formula)*

$$F_n = \frac{1}{\sqrt{5}}\left(\Phi^n - \overline{\Phi}^n\right), \qquad \textit{ahol} \quad \Phi = \frac{1+\sqrt{5}}{2}, \quad \Phi = \frac{1-\sqrt{5}}{2}$$

A rekurzív definíció és az explicit forma között az a különbség, hogy a rekurzív forma esetén, ha F_{100} kiszámítása lenne a cél, akkor szépen el kellene lépegetnünk egyesével 100-ig, az explicit összefüggés viszont "kapásból" megadja az eredményt.

Útmutatás *Keressük a Fibonacci sorozatot $F(n) = x^n$ alakban. Ekkor a rekurzív összefüggés alapján kapjuk, hogy*

$$x^{n+2} = x^n + x^{n+1}.$$

A kapott egyenletet osszuk el x^n-el. Ekkor kapjuk, hogy

$$x^2 = x + 1.$$

Ennek az egyenletnek két gyöke van:

$$x_1 = \frac{1+\sqrt{5}}{2} \qquad x_1 = \frac{1-\sqrt{5}}{2}.$$

Tehát tetszőleges $\alpha, \beta \in \mathbb{R}$ esetén a

$$F_n = \alpha \left(\frac{1+\sqrt{5}}{2}\right)^n + \beta \left(\frac{1-\sqrt{5}}{2}\right)^n \qquad (0, 1, 2, \dots).$$

módon definiált sorozat teljesíti a rekurzív feltételt.

Most válasszuk meg az α és β értékét úgy, hogy $F_1 = 1$, $F_2 = 1$ feltételek (vagy az $F_0 = 0, F_1 = 1$) feltételek teljesüljenek. Ez egy kétismeretlenes két egyenletből álló lineáris egyenletrendszer megoldását jelenti.

Így kapjuk, hogy

$$\alpha = \frac{1}{\sqrt{5}}, \qquad \beta = -\frac{1}{\sqrt{5}}.$$

Ezzel bebizonyítottuk a Binet formulát.

A Binet formula alapján látható, hogy $F(100) = 3.5422 \cdot 10^{20}$, azaz ha a nyulak valóban úgy szaporodnának, ahogyan azt a Fibonacci féle rekurzív formula előírja, akkor 100 hónap elteltével mindent elborítanának a nyulak.

F_n előállítása kerekítéssel

$$F(n) = \text{Round}\left(\frac{1}{\sqrt{5}}\Phi^n\right)$$

Útmutató *Könnyű látni, hogy*

$$F(n) = \frac{1}{\sqrt{5}} \cdot \left(\frac{1+\sqrt{5}}{2}\right)^n + \left(-\frac{1}{\sqrt{5}}\right) \cdot \left(\frac{1-\sqrt{5}}{2}\right)^n \in \mathbb{Z}_+,$$

továbbá

$$\left|\left(-\frac{1}{\sqrt{5}}\right) \cdot \frac{1-\sqrt{5}}{2}\right| < 0.5 \qquad \text{és} \qquad \left|\frac{1-\sqrt{5}}{2}\right| < 1,$$

amiből következik az állítás.

Következmény $F(n) = \Theta\left(\left(\frac{1+\sqrt{5}}{2}\right)^n\right)$.

Szorgalmi feladat *Oldjuk fel az alábbi másodfokú lineáris rekurziókat. Mindhárom esetben legyenek* $T(1) = 1$, $T(2) = 1$,

1. $T(n+2) = 2T(n+1) + T(n)$;
2. $T(n+2) = T(n+1) + 2T(n)$;
3. $3T(n+2) = 2T(n+1) + T(n)$;

   Igazolja, hogy tetszőleges $T(1) = a$ *és* $T(2) = b$ *kezdeti érték feltételek esetén az* $T(n)$ *sorozat konvergens.*

   Határozza meg a $T(n)$ *sorozat határértékét az* a *és* b *számok függvényében; (1 kis piros-pont)*

Szorgalmi feladat *Határozzuk meg az alább rekurzív definícióval adott sorozat 100-adik tagját (számítógép használata nélkül).*

$$G(1) = G(2) = 1, \qquad G(n+2) = G(n+1) + G(n) + 1 \qquad (n > 1).$$

(Tehát a szomszédtól is kapunk minden hónapban egy tenyészképes nyúlpárt.)

Feladat *Adjuk meg rekurzív definícióval az alábbi sorozatokat:*

1. *számtani sorozat (*$a_1 = a$*, illetve adtt még a* d *differencia. A rekurzív definíció:* $a_{n+1} = a_n + d$*. A rekurzió feloldása:* $a_n = a + (n-1)d$*,* $(n = 2, 3, \dots)$*.*
2. *mértani sorozat;*
3. *faktoriális függvény;*
4. *binomiális állandó;*

Feladat *Oldjuk fel az alábbi rekurziót:* $F(1) = 1$, $T(n) = T\left(\frac{n}{2}\right) + 1$

- $T(1) = 1$
- $T(2) = T(1) + 1 = 1 + 1 = 2$
  $T(3) = -$
- $T(4) = T(2) + 1 = 2 + 1 = 3$
- $T(5) = -$
- $T(6) = T(3) + 1 = -$
- $T(7) = -$

- $T(8) = T(4) + 1 = 3 + 1 = 4$

Sejtés: ?

A rekurzió feloldása *$T(n)$ csak akkor számolható, ha $n = 2^k$. Definiáljuk az $U(k)$ sorozatot $U(k) := T(2^k)$ módon. A rekurzió:*

$$U(k) = T(2^k) = T(2^{k-1}) + 1 = U(k-1) + 1 \qquad (k = 1, 2, \dots)$$

$U(0) = 1$, $U(1) = 2$. Így $U(k)$ egy számtani sorozat $U(1) = 2$, $d = 1$ paraméterekkel. A kapott összefüggés $k = 1$ esetén is érvényes.

Ekkor $T(2^k) = k + 1$. Írjunk k helyére $\log(n)$-et, amiből kapjuk, hogy $T(n) = \log(n) + 1$ $n = 1, 2, \dots$, azaz $T(n) = \Theta(\log = (n))$.

0.3.1. A mester tétel és alkalmazása

Definíció *Legyen $p \geqslant 0$ és $f(n)$ egy növekedési függvény. Azt mondjuk, hogy*

- ***az $f(n)$ polinomiálisan lassabban nő, mint a n^p tesztpolinom**, ha $\exists \varepsilon > 0 : f(n) = \mathcal{O}(n^{p-\varepsilon})$*
- ***az $f(n)$ polinomiálisan gyorsabban nő, mint a n^p tesztpolinom**, ha $\exists \varepsilon > 0 : f(n) = \Omega(n^{p+\varepsilon})$*

Feladat *Bizonyítsuk be, hogy ha $\varepsilon > 0$, akkor*

$$\log(n) = \mathcal{O}(n^{\varepsilon}),$$

így $\log(n)$ polinomiálisan lassabban nő, mint az n^{ε} tesztpolinom.

Megoldás *A L'Hospital szabály alkalmazásával kapjuk, hogy*

$$\lim_{x\to\infty} \frac{\log(n)}{x^{\varepsilon}} = \lim_{x\to\infty} \frac{\frac{1}{\ln(2)}\frac{1}{x}}{\varepsilon x^{\varepsilon-1}} = \frac{1}{\ln(2)} \lim_{x\to\infty} \frac{1}{\varepsilon x^{\varepsilon}} = 0,$$

azaz $\log(n) = o(n^{\varepsilon})$, amiből kapjuk, hogy $\log(n) = \mathcal{O}(n^{\varepsilon})$.

Feladat *Legyenek $f(n) = n\log(n)$, $g(n) = n^p$ valamely $p \geqslant 0$ esetén. Ekkor*

a. *Ha $p > 1$, akkor $f(n) = n\log(n)$ polinomiálisan lassabban nő, mint a $g(n) = n^p$ tesztpolinom.*

b. *Ha $p = 1$, akkor $f(n) = n\log(n)$ gyorsabban nő, mint a $g(n) = n^p$ tesztpolinom, de nem polinomiálisan.*

c. *Ha $0 \leqslant p < 1$, akkor $f(n) = n\log(n)$ polinomiálisan gyorsabban nő, mint a $g(n) = n^p$ tesztpolinom.*

Mo.:

a. *Legyen $p > 1$. Ekkor létezik $\delta > 0$ úgy, hogy $1 + \delta$. Ekkor van olyan kicsi pozitív ε hogy $\delta - \varepsilon > 0$. Így*

$$\frac{n\log(n)}{n^{1+\delta-\varepsilon}} = \frac{\log(n)}{n^{\delta-\varepsilon}} \to 0,$$

azaz $n\log(n) = \mathcal{O}(n^{p-\varepsilon})$.

b. *Nyilvánvaló, hogy*

$$\frac{f(n)}{g(n)} = \frac{n\log(n)}{n} = \log(n) \longrightarrow +\infty$$

azaz $f(n) = n\log(n) = \Omega(n^1)$.

Azonban, ha $\varepsilon > 0$, akkor

$$\frac{f(n)}{g(n)} = \frac{n\log(n)}{n^{1+\varepsilon}} = \frac{\log(n)}{n^{\varepsilon}} \longrightarrow 0.$$

azaz $n\log(n) = \mathcal{O}(n^{1+\varepsilon})$, amiből kapjuk, hogy $n\log(n) \neq \Omega(n^{1+\varepsilon})$.

c. *Végül legyen $0 < p < 1$. Ekkor létezik $\varepsilon > 0$ úgy, hogy $p + \varepsilon < 1$, azaz $0 < 1 - (p + \varepsilon)$. Így*

$$\lim_{n\to\infty} \frac{n\log(n)}{n^{p+\varepsilon}} = \lim_{n\to\infty} n^{1-(p+\varepsilon)} \cdot \log(n) =$$
$$= \left(\lim_{n\to\infty} n^{1-(p+\varepsilon)}\right) \cdot \left(\lim_{n\to\infty} \log(n)\right) = (+\infty) \cdot (+\infty) = +\infty.$$

A Mester tétel arra használható, hogy bizonyos speciális alakú rekurzív módon adott növekedési függvényről a rekurzió feloldása nélkül meg tudjuk mondani, hogy milyen növekedési rendű.

A mester tétel *Legyenek $a \geqslant 1$, $b > 1$, $p = \log_b(a)$, $a, b \in \mathbb{R}$, $f : \mathbb{Z}_+ \to \mathbb{R}_+$ egy növekedési függvény, $g(n) = n^p$ a tesztpolinom.*

Rekurziós összefüggés:

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

M1 *Ha $f(n)$ polinomiálisan lassabb növekedésű, mint a $g(n) = n^p$ tesztpolinom, akkor $T(n) = \Theta(g(n))$.*

M2 *Ha $f(n) = \Theta(g(n))$, akkor $T(n) = \Theta(g(n)\log(n))$.*

M3 *Ha $f(n)$ polinomiálisan gyorsabb növekedésű, mint a $g(n) = n^p$ tesztpolinom és teljesül az úgynevezett* ***regularitási feltétel****, azaz*

$$\exists(c < 1)\exists(n_0 \in \mathbb{Z}_+)\forall(n \geqslant n_0) : af\left(\frac{n}{b}\right) \leqslant cf(n)$$

akkor $T(n) = \Theta(f(n))$.

Feladatok

1. $T(n) = T(\frac{n}{2}) + 1$.

   $a = 1$, $b = 2$, $p = \log_2(1) = 0$, $f(n) = n^0 = 1$, $g(n) = n^0 = 1$. *Ekkor* $f(n) = \Theta(g(n))$. *Így a* $M2$ *alapján* $T(n) = \Theta(1.\log(n))$.

2. $T(n) = 4T(\frac{n}{2}) + n$.

   $a = 4$, $b = 2$, $p = \log_b(a) = \log_2(4) = 2$, $f(n) = n^1$, $g(n) = n^2$. *Az* $f(n) = n^1$ *polinomiálisan lassabban nő, mint* $g(n) = n_2$ *tesztpolinom, így a* $M1$ *alapján* $T(n) = \Theta(g(n)) = \Theta(n^2)$.

3. $T(n) = 4T(\frac{n}{2}) + n^2$.

   $a = 4$, $b = 2$, $p = \log_b(a) = \log_2(4) = 2$, $f(n) = n^2$, $g(n) = n^2$. $f(n) = \Theta(g(n))$, $M2$ *alapján* $T(n) = \Theta(g(n)\log(n)) = \Theta(n^2\log(n))$.

4. $T(n) = 4T(\frac{n}{3}) + n^3$.

   $a = 4$, $b = 3$, $p = \log_b(a) = \log_3(4)$, $f(n) = n^3$, $g(n) = n^p$. $p = \log_3(4) < \log_3(27) = 3$ *(itt használtunk egy kis becslést, de elég lett volna* $\log_3(4)$*-et beütni a számológépbe ahhoz, hogy megkapjuk, hogy* $\log_3(4) < 3$*).* $f(n) = n^3$ *polinomiálisan gyorsabban nő, mint a* $g(n)$*. Az* $M3$*-hoz még ellenőrizni kell még a relgulatitási feltételt:*

   $$af\left(\frac{n}{b}\right) = 4 \cdot \left(\frac{n}{3}\right)^3 = \frac{4}{27} \cdot n^3 = \frac{4}{27} \cdot f(n), \qquad c = \frac{4}{27} < 1$$

   Így kapjuk, hogy $T(n) = \Theta(f(n)) = \Theta(n^3)$.

5. $T(n) = 3T(\frac{n}{4}) + n\log(n)$.

   $a = 3$, $b = 4$, $p = \log_b(a) = \log_4(3)$, $f(n) = n\log(n)$, $g(n) = n^p$

   $$p = \log_4(3) < \log_4(4) = 1$$

   Ekkor $f(n) = n\log(n)$ *polinomiálisan gyorsabban nő, mint a* $g(n) = n^{\log_4(3)}$*. Az* $M3$*-hoz ellenőrizni kell még a regularitási feltételt.*

   Regularitási feltétel:

   $$af\left(\frac{n}{b}\right) = 3\frac{n}{4}\log\left(\frac{n}{4}\right) < \frac{3}{4}n\log(n) = \frac{3}{4}f(n), \qquad c = \frac{3}{4} < 1$$

   Így $T(n) = \Theta(f(n)) = \Theta(n\log(n))$.

6. $T(n) = 2T(\frac{n}{2}) + n\log(n)$

   $a = 2$, $b = 2$, $p = \log_b(a) = \log_2(2) = 1$, $f(n) = n\log(n)$, $g(n) = n^p = n^1$. *Ekkor* $f(n)$ *gyorsabban nő, mint a* $g(n)$*, de nem polinomálisan, így* **a *Mester tétel nem alkalmazható***.

A rekurzió feloldása:

$$\begin{aligned}
&T(1) = a > 0\\
&T(2) = 2T(1) + 2\log(2) = 2a + 2\\
&T(4) = 2T(2) + 4\log(4) = 2(2a+2) + 8 = 4a + 12\\
&T(8) = 2T(4) + 8\log(8) = 2(4a+12) + 24 = 8a + 48.
\end{aligned}$$

A $T(2^n)$ függvényt

$$T(2^n) = 2^n a + a_n$$

alakban keressük. Felírjuk az a_n sorozatot definiáló rekurziót.

$$\begin{aligned}
T(2^n) &= 2T(2^{n-1}) + n2^n =\\
&= 2\left(2^{n-1}a + a_{n-1}\right) + n2^n =\\
&= 2^n a + 2a_{n-1} + n2^n.
\end{aligned}$$

Így kapjuk, hogy

$$a_1 = 2, \qquad a_n = 2a_{n-1} + n2^n \quad (n > 1).$$

Oldjuk fel az a_n-re vonatkozó rekurziót.

$$\begin{aligned}
a_n &= 2a_{n-1} + n2^n = 2(2a_{n-2} + (n-1)2^{n-1}) + n\cdot 2^n =\\
&= 2^2(a_{n-2} + (n-1)2^n + n2^n = 2^2 a_{n-2} + [n + (n-1)]2^n =\\
&\vdots\\
&= 2^k a_{n-k} + [n + (n-1) + \cdots + (n-k+1)]2^n\big|_{k=n-1} =\\
&= 2^{n-1}\cdot 2 + [n + (n-1) + \cdots + 2]\cdot 2^n = 2^n(1 + 2 + \cdots + n) =\\
&= 2^n \frac{n(n+1)}{2}.
\end{aligned}$$

(Tehát a_n a 2 n-edik hatványának és az n-edik háromszögszámnak a szorzata.)

Így kapjuk, hogy $T(2^n) = 2^n a + 2^n\frac{n(n+1)}{2}$. Végezzük el a $n \leftarrow \log(n)$ helyettesítést. Így kapjuk, hogy

$$\begin{aligned}
T(n) &= na + \frac{1}{2}n\log(n)(\log(n)+1) =\\
&= na + \frac{1}{2}n\log(n) + \frac{1}{2}n\log(n)\log(n) = \Theta(n\log(n)\log(n)).
\end{aligned}$$

7. $T(n) = 8T(\frac{n}{2}) + 2^n)$

   $a = 8$, $b = 2$, $p = \log_b(a) = \log_2(8) = 3$, $f(n) = 2^n$, $g(n) = n^3$. Az $f(n) = 2^n$ polinomiálisan gyorsabban nő, mint a $g(n) = n^3$ tesztpolinom. A mester tétel alkalmazásához ellenőrizni kell a regularitási feltételt.

Regularitási feltétel:

$$af\left(\frac{n}{b}\right) = 8 \cdot 2^{\frac{n}{2}} = 2^{\frac{n}{2}+3} < 2^{n-1} = \frac{1}{2} \cdot 2^n = \frac{1}{2}f(n),$$

ha $\frac{n}{2} + 3 < n - 1$, *mert* $4 < \frac{n}{2}$, *mivel* $8 < n$.

Így M3 alapján kapjuk, hogy $T(n) = \Theta(f(n)) = \Theta(2^n)$.

8. $T(n) = 2T(\frac{n}{3}) + n$

   $a = 2$, $b = 3$, $p = \log_b(a) = \log_3(2)$, $f(n) = n^1$, $g(n) = n^p$.

   $$p = \log_3(2) < \log_3(3) = 1$$

   Az $f(n)n^1$ *polinomiálisan gyorsabban nő, mint a* $g(n)n^{\log_3(2)}$

   Regulatitási feltétel:

   $$af\left(\frac{n}{b}\right) = 2\frac{n}{3} = \frac{2}{3}n = \frac{2}{3}f(n) \qquad \left(c = \frac{2}{3}\right)$$

   így az M3 alapján kapjuk, hogy $T(n) = \Theta(f(n)) = \Theta(n)$.

9. $T(n) = 3T(\frac{n}{2}) + n$

   $a = 3$, $b = 2$, $p = \log_b(a) = \log_2(3)$, $f(n) = n^1$, $g(n) = n^p$

   $$p = \log_2(3) > \log_2(2) = 1$$

   $f(n) = n^1$ *polinomiálisan lassabban nő, mint a* $g(n) = n^{\log_2(3)}$ *tesztpolinom, így* $M1$ *alapján* $T(n) = \Theta(n^{\log_2(3)})$.

10. $T(n) = 2T(\frac{n}{2}) + n$

    $a = 2$, $b = 2$, $p = \log_2(a) = \log_2(2) = 1$, $f(n) = n$, $g(n) = n$. *Ekkor* $f(n) = \Theta(g(n))$, *így* $M2$ *miatt* $T(n) = \Theta(g(n)) = \Theta(n)$.

11. $T(n) = T(\frac{n}{2}) + n$

    $a = 1$, $b = 2$, $p = \log_b(a) = \log_2(1) = 0$, $f(n) = n$, $g(n) = n^0 = 1$. $f(n) = n$ *polinomiálisan gyorsabban nő, mint a* $g(n) = 1$.

    Regularitási feltétel:

    $$af\left(\frac{n}{b}\right) = \frac{\frac{n}{2}}{2} = \frac{1}{2}\frac{n}{2} = \frac{1}{2}f(n) \qquad \left(c = \frac{1}{2}\right)$$

    Ekkor M3 alapján: $T(n) = \Theta(f(n)) = \Theta(n)$.

0.3.2. Számelméleti algoritmusok I.

Számelméleti definíciókat (fogalmak) és tételeket (állítások) a $\mathbb{Z}$ halmazon fogalmazzuk meg, az algoritmusok $\mathbb{Z}_+ \cup \{0\}$ halmazon futtatjuk. Ez egy nagyon szerencsés tárgyalásmód, mivel az egész számok halmazának nagyon jó algebrai tulajdonságai vannak. Például $\mathbb{Z} = \mathbb{Z}(+,\cdot)$ egy gyűrű, így az egész számok halmaza helyett jobb lenne az egész számok gyűrűje elnevezés. Ráadásul $\mathbb{Z}(+,\cdot,\leqslant)$ egy rendezett gyűrű, továbbá $\mathbb{Z}(+,\cdot)$ egy integritás tartomány, sőt Euklideszi gyűrű, azaz érvényes benne az Euklideszi osztás tétele. A számelméleti algoritmusok részben a szám szó mindig egész számot jelent.

Definíció ***Oszthatósági reláció*** *$\mathbb{Z}$-n.*

*Legyenek $a, b \in \mathbb{Z}$. "a" **osztója** "b"-nek (illetve "b" többszöröse "a"-nak), ha $\exists c \in \mathbb{Z} : b = ac$. Jele: $a|b$.*

Szorgalmi feladat *Bizonyítsa be, hogy $a|0$ akkor és csak akkor, ha $a = 0$.*

Definíció *Egy $p \in \mathbb{Z}_+$, $p > 1$ számot **prímszám**nak nevezünk, ha 1-en és önmagán kívül nincs más pozitív osztója.*

Megjegyzés *A prímszám szokásos definíciója: egy p 0-tól és ± 1-től különböző egész számot prímszámnak nevezünk, ha valhányszor osztója egy szorzatnak, mindannyiszor osztója a szorzat valamelyik tényezőjének, azaz*

$$p|ab \qquad \Longleftrightarrow \qquad p|a \text{ vagy } p|b.$$

*Mi ezt a tulajdonságot a későbbiekben **prímtulajdonság**nak fogjuk nevezni.*

*Egy p 0-tól és ± 1-től különböző egész számot felbonthatatlannak, vagy **irreducibilis**nek nevezünk, ha nincs valódi faktorizációja, azaz $p = ab$-ből következik, hogy $a = \pm 1$, vagy $a = \pm p$.*

Látható, hogy az általunk követett felépítésben az irreducibilis számokat nevezzük prímeknek. (Ugyanezt tettük a középiskolában is, bár az ott tanult számelmélet elég hézagosan volt felépítve.) Ebből a későbbiekben nem fog baj származni, ugyanis látható, hogy az egész számok halmazában a prímszámok és az irreducibilis számok ugyanazok.

Könnyű bizonyítani, hogy az egész számok halmazán minden prímszám irreducibilis.

Könnyű bebizonyítani, hogy az egész számok halmazán érvényes az úgynevezett általánosított prímtulajdonság, mely szerint ha $p|ab$ és $\mathrm{lnko}(p, a) = 1$, akkor $p|b$.

Az általánosított prímtulajdonság felhasználásával könnyű bizonyítani, hogy az egész számok halmazán minden irreducibilis szám prím., tehát a prímszámok és az irreducibilis számok valóban egybeesnek.

Definíció *Az a, $b \in \mathbb{Z}$ számok **legnagyobb közös osztó**ja*

$$\mathrm{lnko}(a,b) = \begin{cases} 0, & \text{ha } a = 0 \text{ és } b = 0 \\ \max\{d \in \mathbb{Z}_+ \mid d|a \text{ és } d|b\} & \text{egyébként} \end{cases}$$

Megjegyzés *Ha az a és b számok közül nem mindkettő 0, akkor egy δ számot az a és b számok* ***kitüntetett közös osztó****jának nevezünk, ha a δ szám*

1. *az a és b számok közös osztója;*
2. *az a és b számok bármely közös osztójának a többszöröse.*

Könnyű látni, hogy a kitüntetett közös osztó előjeltől eltekintve egyértelműen létezik. A kitüntetett közös osztó jobb fogalom, mint a legnagyobb közös osztó, mert a definíció nem használja fel a rendezési relációt, hanem közvetlenül az oszthatóság fogalmára épül.

Elemi számelméletben meg szokták mutatni, hogy ha az a és b számok közül nem mindkettő 0, akkor az a és b számok legnagyobb közös osztója egyúttal a kitüntetett közös osztójuk is. Ez a bizonyítás az általunk adott felépítésben is szerepelni fog.

0.3.3. Elméleti ismeretek

Jelölés *Tetszőleges $a,\ b \in \mathbb{Z}$ számok esetén legyen*

$$L(a,b) := \{ua + vb \mid u \in\in \mathbb{Z},\ v \in \mathbb{Z}\}.$$

Az $L(a,b)$ halmazt az a ***és*** *b* ***elemekből képzett lineáris kombinációk halmazá****nak nevezzük.*

Most megismerkedünk a reprezentációs, a redukciós és a rekurziós tétellel. A redukciós tétel az alapja a redukciós algoritmusnak, amely segítségével ki lehet számolni két szám legnagyobb közös osztóját.

Ez az algoritmus elég régi, már szerepel **Euklidész** által írt Elemek című könyvben (i. e. 300 körül), ami egy ókori egyetemi tankönyvnek tekinthető. Mai a könyvben szereplő felépítésben tanuljuk a középiskolában a geometriát.

A redukciós tételből könnyen kijön a rekurziós tétel (illetve az Euklidészi algoritmus), mivel, ha $0 < b < a$ (egész számok), akkor az $a \bmod b$ értéke ismételt kivonások segítségével kiszámítható.

A $\sqrt{2}$ irracionális voltának első bizonyítása is az Euklideszi algoritmushoz köthető.

A reprezentációs tétel azt mondja ki (többek között), hogy az a és b egész számok legnagyobb közös osztója felírható az a és b számok lineáris kombinációjaként, ennek a felírásnak a megkeresésére fog szolgálni a Euklidészi algoritmus egy továbbfejlesztett változata, az úgynevezett kibővített Euklidészi algoritmus.

A kibővített Euklidészi algoritmus segítségével fogjuk meghatározni a lineáris kongruencia rendszer alaprendszerét, ez fogalmi úton elvezet minket a multiplikatív inverz fogalmához. Ez utóbbira az RSA algoritmus működtetéséhez lesz szükségünk. (A másik szükséges algoritmus a moduláris hatványozás lesz.) Illetve a matematikai alapot az Euler-Fermat tétel szolgáltatja. (Szokták kiszenvedni a kis Fermat tételből is de ez az út nem túl elegáns.)

Az RSA algoritmus egy titkosításra használt algoritmus, amely azért alkalmas a titkosításra, mert a (nagy) pozitív egészek prímfakrorizációja a jelenleg használt gépekkel és algoritmusokkal nem kivitelezhető a nagy futási idő miatt. Ha majd kivitelezhető lesz, (remélem) akkor sem fognak eltűnni az algoritmikus számelmélet elemei a tananyagból.

1. Reprezentációs tétel *Ha a és b egyidejűleg nem 0, akkor*

$$\text{lnko}(a,b) = \min\{x \in L(a,b) \mid x > 0\}.$$

Következmény *A legnagyobb közös osztó bármely közös osztó többszöröse.*

2. Redukciós tétel *Legyenek $a, b \in \mathbb{Z}$. Ekkor*

$$\text{lnko}(a,b) = \text{lnko}(a-b,b)$$

3. Rekurziós tétel *Legyenek $a, b \in \mathbb{Z}$. Ekkor*

$$\text{lnko}(a,b) = \text{lnko}(b, a \mod b)$$

0.3.4. Algoritmusok

1. Redukciós algoritmus

	REDUK(a,b || d^*)
INPUT	$a, b \in \mathbb{Z}_+ \cup \{0\}$
OUTPUT	d^*
1.	*WHILE* $(b \neq 0)$
2.	*IF* $b > a$ *THEN CSERE*(a,b)
3.	$a \longleftarrow a - b$
4.	$d^* \longleftarrow a$
5.	*RETURN*(d^*)

2. Euklideszi algoritmus

	EUK(a,b || d^*)
INPUT	$a, b \in \mathbb{Z}_+ \cup \{0\}$
OUTPUT	d^*
1.	*WHILE* $(b \neq 0)$
2.	$\binom{a}{b} \longleftarrow \binom{b}{a \mod b}$
3.	$d^* \longleftarrow a$
4.	*RETURN*(d^*)

Példa *Határozzuk meg a redukciós algoritmus és a rekurziós algoritmus segítségével a 90 és a 24 legnagyobb közös osztóját. A bulet módon megjelölt lépések azok, amelyeken keresztül a rekurziós algoritmus eljut a legnagyobb közös osztóhoz.*

$$\dot{\text{lnko}}\ (90,24) = \text{lnko}(66,24) = \text{lnko}(42,24) = \text{lnko}(18,24) = \dot{\text{lnko}}\ (24,18) = \text{lnko}(6,18) =$$
$$= \dot{\text{lnko}}\ (18,6) = \text{lnko}(12,6) = \text{lnko}(6,6) = \text{lnko}(0,6) = \dot{\text{lnko}}\ (6,0) = 6$$

A rekurziós algoritmus, mint Euklideszi osztások sorozata *A rekurziós algoritmust lehet úgy értelmezni, mint Euklideszi osztások sorozatát, melyben*

- *Minden egyes lépésben az osztóból lesz az osztandó és a maradékból lesz az osztó;*
- *Ezt az eljárást addig iteráljuk, amíg a maradék nem válik nullává;*
- *A legnagyobb közös osztó a maradékok sorozatában az utolsó zérustól különböző maradék.*

$$\begin{aligned} a_0 &= q_0 b_0 + r_0 \\ b_0 &= q_1 r_0 + r_1 \\ r_0 &= q_2 r_1 + r_2 \\ &\vdots \\ r_{n-2} &= q_n r_{n-1} + \boxed{r_n} \\ r_{n-1} &= q_{n+1} r_n + 0 \end{aligned}$$

Lamé tétele *Ha az euklideszi algoritmusban $a > b \geqslant 0$ és $F_{k+1} > b$, akkor a rekurzív hívások száma kevesebb, mint k.*

0.4. Számelméleti algoritmusok II

0.4.1. Egy kis számelméleti alapozás

A gyakorlatoknak nem feladata az állítások bizonyítása, azonban bizonyos ravasz egy-két soros bizonyítások esetén a hézagmentesség kedvéért kivételt teszünk.

Jelölés: *Ebben az alfejezetben az a és b számok legnagyobb közös osztóját (a, b) módon fogjuk jelölni.*

Az oszthatósági reláció tulajdonságai:

1. *Ha $d|a$ és $d|b$, akkor $d|a + b$.*

2. *$a|b$ és $b|a$ akkor és csak akkor, ha $a = \pm b$.*

A legnagyobb közös osztó tulajdonságai

1. *Asszociatív, azaz $(a, (b, c)) = ((a, b), c)$.*

2. *$(ca, cb) = c(a, b)$.*

Az általánosított prímtulajdonság *Ha $d|ab$ és $(d, a) = 1$, akkor $d|b$.*

Bizonyítás

$$d = (ab, d) = (ab, (db, d)) = ((ab, db), d) = (b(a, d), d) = (b, d),$$

tehát $d|b$.

Tétel *Ha $(a, b) = 1$, akkor $(a, bc) = (a, c)$.*

Bizonyítás *Legyen $d = (a, bc)$.*
Mivel

$$d|(ac, bc) = c(a, b) = c,$$

amiből kapjuk, hogy $d|(a, c)$.
Az $(a, c)|(a, bc)$ reláció triviális módon teljesül.

Az utolsó tétel, illetve a redukciós algoritmus egyszerű következménye az alábbi, úgynevezett Bináris legnagyobb közös osztó algoritmus, amelyet sokan azért szeretnek, mert a kettes számrendszerben páros szám 2-vel való osztása könnyen kivitelezhető.

Bináris legnagyobb közös osztó algoritmus *Legyenek* $a, b \in \mathbb{Z}_+ \cup \{0\}$

$$\text{lnko}(a,b) = \begin{cases} a, & \text{ha } b = 0 \\ b, & \text{ha } a = 0 \\ c, & \text{egyébként,} \end{cases}$$

ahol a c szám az alábbi táblázatban van definiálva:

$a \setminus b$	ps	pt
ps	$2\text{lnko}\left(\frac{a}{2}, \frac{b}{2}\right)$	$\text{lnko}\left(\frac{a}{2}, b\right)$
pt	$\text{lnko}\left(a, \frac{b}{2}\right)$	$\text{lnko}\left(\frac{a-b}{2}, b\right)$

Feladat *A bináris közös osztó algoritmussal számoljuk ki a 90 és a 24 legnagyobb közös osztóját.*
Mo.:

$$\begin{aligned} \text{lnko}(90,24) &= 2\text{lnko}(45,12) = 2\text{lnko}(45,6) = \\ &= 2\text{lnko}(45,3) = 2\text{lnko}(21,3) = \\ &= 2\text{lnko}(18,3) = 2\text{lnko}(9,3) = \\ &= 2\text{lnko}(3,3) = 2\text{lnko}(0,3) = 2 \cdot 3 = 6 \end{aligned}$$

0.4.2. A kibővített euklideszi algoritmus rekurzív változata

Két nemnegatív egész szám a és b legnagyobb közös osztóját $d^* = \text{lnko}(a,b)$ *módon jelöljük. A reprezentációs tétel alapján* d^* *előállítható az a és b számok lineáris kombinációjaként, ahol az együtthatók* $\mathbb{Z}$*-beliek, azaz*

$$\exists(x^* \in \mathbb{Z})\exists(y^* \in \mathbb{Z}) : d^* = x^*a + y^*b.$$

A kibővített euklideszi algoritmus a d^**-ot és az* x^* *és* $y*$ *együtthatókat is szolgáltatja.*

	KREUK(a,b $\|$ d^*, x^*, y^**)*
INPUT	$a, b \in \mathbb{Z}_+ \cup \{0\}$
OUTPUT	$d^* \in \mathbb{Z}_+ \cup \{0\}, x^*, y^* \in \mathbb{Z} : d^* = \text{lnko}(a,b), d^* = x^*a + y^*b$
1.	*IF* $(b = 0)$
2.	*THEN* $d^* \longleftarrow a$
3.	$x^* \longleftarrow 1$
4.	$y^* \longleftarrow 0$
5.	*ELSE KREUK* $(b, a \mod b, d^*, x^*, y^*)$
6.	$\binom{x^*}{y^*} \longleftarrow \binom{y^*}{x^* - \lfloor \frac{a}{b} \rfloor y^*}$
7.	*RETURN(*d^*, x^*, y^**)*

1. Feladat *Futtassuk a kibővített euklideszi algoritmust az* $(a,b) = (133, 157)$ *számpáron.*

E: $(-72)133 + 61 \cdot 157 = 1$.

k	a_k	b_k	q_k	r_k	d	x_k	y_k
0	*133*	*157*	*0*	*133*	*1*	*-72*	$61 - 0 \cdot (-72) = 61$
1	*157*	*133*	*1*	*24*	*1*	*61*	$-11 - 1 \cdot 61 = -72$
2	*133*	*24*	*5*	*13*	*1*	*-11*	$6 - 5 \cdot (-11) = 61$
3	*24*	*13*	*1*	*11*	*1*	*6*	$-5 - 1 \cdot 6 = -11$
4	*13*	*11*	*1*	*2*	*1*	*-5*	$1 - 1 \cdot (-5) = 6$
5	*11*	*2*	*5*	*1*	*1*	*1*	$0 - 5 \cdot 1 = -5$
6	*2*	*1*	*2*	*0*	*1*	*0*	$1 - 2 \cdot 0 = 1$
7	*1*	*0*	-	-	*1*	*1*	*0*

2. Feladat $(a, b) = (90, 24)$.

E: $(-1) \cdot 90 + 4 \cdot 24 = 6$.

k	a_k	b_k	q_k	r_k	d	x_k	y_k
0	*90*	*24*	*3*	*18*	*6*	*-1*	$1 - 3 \cdot (-1) = 4$
1	*24*	*18*	*1*	*6*	*6*	*1*	$0 - 1 \cdot 1 = -1$
2	*18*	*6*	*3*	*0*	*6*	*0*	$1 - 3 \cdot 0 = 1$
3	*6*	*0*	-	-	*6*	*1*	*0*

3. Feladat $(a, b) = (628, 44)$

E: $(-19) \cdot 628 + 27 \cdot 442 = 2$.

k	a_k	b_k	q_k	r_k	d	x_k	y_k
0	*628*	*442*	*1*	*186*	*2*	*-19*	$8 - 1 \cdot (-19) = 27$
1	*442*	*186*	*2*	*70*	*2*	*8*	$-3 - 2 \cdot 8 = -19$
2	*186*	*70*	*2*	*46*	*2*	*-3*	$2 - 2 \cdot (-3) = 8$
3	*70*	*46*	*1*	*24*	*2*	*2*	$-1 - 1 \cdot 2 = -3$
4	*46*	*24*	*1*	*22*	*2*	*-1*	$1 - 1 \cdot (-1) = 2$
5	*24*	*22*	*1*	*2*	*2*	*1*	$0 - 1 \cdot 1 = -1$
6	*22*	*2*	*11*	*0*	*2*	*0*	$1 - 11 \cdot 0 = 1$
7	*2*	*0*	-	-	*2*	*1*	*0*

4. Feladat $(a, b) = (988, 610)$

E: $(-71)\cdot 988 + 115\cdot 610 = 2$.

k	a_k	b_k	q_k	r_k	d	x_k	y_k
0	*988*	*610*	*1*	*378*	*2*	*-71*	$44-1\cdot(-71)=115$
1	*610*	*378*	*1*	*232*	*2*	*44*	$-27-1\cdot 44=-71$
2	*378*	*232*	*1*	*146*	*2*	*-27*	$17-1\cdot(-27)=44$
3	*232*	*146*	*1*	*86*	*2*	*17*	$-10-1\cdot 17=-27$
4	*146*	*86*	*1*	*60*	*2*	*-10*	$7-1\cdot(-10)$
5	*86*	*60*	*1*	*26*	*2*	*7*	$-3-1\cdot 7=-10$
6	*60*	*26*	*2*	*8*	*2*	*-3*	$1-2\cdot(-3)=7$
7	*26*	*8*	*3*	*2*	*2*	*1*	$0-3\cdot 1$
8	*8*	*2*	*4*	*0*	*2*	*0*	$1-4\cdot 0=1$
9	*2*	*0*	-	-	*2*	*1*	*0*

0.4.3. A kongruencia fogalma

Definíció *Legyenek* $a, b \in \mathbb{Z}$, $m > 1$. *Definiáljuk az* m *modulus szerinti* ***kongruencia relációt*** *az alábbi módon*

$$a \equiv b \mod (m) \qquad \Longleftrightarrow \qquad n \mid b-a.$$

Megjegyzés *Könnyű látni, hogy* $a \equiv b \mod (m)$ *akkor és csak akkor, ha* a *és* b m*-mel maradékosan osztva ugyanazt a nemnegatív maradékot adja.*

A következő tulajdonságok triviálisak, de nagyon fontosak.

Tétel ***A kongruencia reláció tulajdonságai***

1. $a \equiv a \mod (m)$;

2. $a \equiv b \mod (m) \qquad \Rightarrow \qquad b \equiv a \mod (m)$;

3. $a \equiv b \mod (m)$ *és* $b \equiv c \mod (m)$, *akkor* $a \equiv c \mod (m)$.

Az 1.,2.,3., tulajdonságok azt fejezik ki, hogy rögzített m *modulus esetén a kongruencia reláció egy ekvivalencia reláció* $\mathbb{Z}$*-n. Ennek hatására* $\mathbb{Z}$ *ekvivalencia osztályokra esik szét.*

4. $a \equiv b \mod (m)$ *és* $c \equiv d \mod (m)$, *akkor* $(a+c) \equiv (b+d) \mod (m)$.

5. $a \equiv b \mod (m)$ *és* $c \equiv d \mod (m)$, *akkor* $ac \equiv bd \mod (m)$.

A 4.,5. tulajdonságok azt fejezik ki, hogy az ekvivalencia reláció kompatibilis a műveletekkel, így a faktorhalmazon művelet definiálható a reprezentáns elemek segítségével.

Magyarázat *Az egész számok halmazán értelmezve van két művelet, az összeadás és a szorzás. A fenti tétel alapján a* $\mathbb{Z}$ *szétesik ekvivalencia osztályokra. Ez azt jelenti, hogy* $\mathbb{Z}$ *olyan halmazok uniója, melyekre teljesül, hogy*

1. *Egyikük sem üres;*

2. *Páronként diszjunktak, azaz közös elem nélküliek;*

3. *Uniójuk kiadja a* $\mathbb{Z}$ *halmazt.*

Ezek között az úgynevezett maradékosztályok között műveletet értelmezhetünk a reprezentáns elemek segítségével. A reprezentáns elem a maradékosztály tetszőleges elemét jelenti. Legyen $\underline{a}$, $\underline{b}$ *két maradékosztály. Ekkor* $a \in \underline{a}$ *és* $b \in \underline{b}$. *Ha* a_1, $a_2 \in \underline{a}$, b_1, $b_2 \in \underline{b}$, *akkor*

$$a_1 + b_1 \qquad \text{és} \qquad a_2 + b_2$$

ugyanabba a maradékosztályba esnek. Illetve analóg módon

$$a_1 \cdot b_1 \qquad \text{és} \qquad a_2 \cdot b_2$$

is ugyanabba a maradékosztályba esnek. Így a reprezentáns elemek segítségével műveletet definiálhatunk a maradékosztályok halmazán, azaz a faktorhalmazon:

$$\underline{a} + \underline{b} := \underline{a + b} \qquad \text{és} \underline{a} \cdot \underline{b} := \underline{a \cdot b}.$$

Példa *Legyen* $m = 6$. *Ekkor* $\mathbb{Z}_6 = \{\underline{0}, \underline{1}, \underline{2}, \underline{3}, \underline{4}, \underline{5}\}$, *ahol*

$$\begin{aligned}
\underline{0} &= \{\dots, -12, -6, 0,\ 6, 12, \dots\} = \{0 + 6k \mid k \in \mathbb{Z}\}, \\
\underline{1} &= \{\dots, -11, -5, 1,\ 7, 13, \dots\} = \{1 + 6k \mid k \in \mathbb{Z}\}, \\
\underline{2} &= \{\dots, -10, -4, 2,\ 8, 14, \dots\} = \{2 + 6k \mid k \in \mathbb{Z}\}, \\
\underline{3} &= \{\dots,\ -9, -3, 3,\ 9, 15, \dots\} = \{3 + 6k \mid k \in \mathbb{Z}\}, \\
\underline{4} &= \{\dots,\ -8, -2, 4, 10, 16, \dots\} = \{4 + 6k \mid k \in \mathbb{Z}\}, \\
\underline{5} &= \{\dots,\ -7, -1, 5, 11, 17, \dots\} = \{5 + 6k \mid k \in \mathbb{Z}\}.
\end{aligned}$$

A műveletek Cayley táblázata $\mathbb{Z}_6$**-ban**

$+$	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	2	3	4	5	0
2	2	3	4	5	0	1
3	3	4	5	0	1	2
4	4	5	0	1	2	3
5	5	0	1	2	3	4

$\cdot$	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	2	3	4	5
2	0	2	4	0	2	4
3	0	3	0	3	0	3
4	0	4	2	0	4	2
5	0	5	4	3	2	1

Mielőtt továbbhaladnánk, átismételünk néhány korábban tanult fogalmat.

0.4.4. Absztrakt algebrai fogalmak

A most következő fogalmakat már tanultuk Diszkrét Matematika tantárgy keretében és nem azért tanultuk, hogy ne használjuk. Ezek nélkül a fogalmak nélkül is elboldogulnánk, de azt gondolom, hogy egy egyetemi jegyzetnek nem kell egy középiskolai matematika szakkör stílusában íródnia, másrészt nem árt egy kis kitekintés. "Nagyobb általánosság, nagyobb tisztaság."

Legyen X egy nemüres halmaz, $*$ egy binér algebrai művelet X-en, azaz egy olyan függvény, amely minden (x, y) X-beli elemekből alkotott párhoz hozzárendel egy $x*y$ módon jelölt X-beli elemet. Ekkor azt mondjuk, hogy $X = (X, *)$ egy algebrai struktúra.

Most megismerkedünk néhány fontos egy, illetve kétműveletes algebrai struktúrával, melyek a következők: félcsoport, csoport, gyűrű, test.

Ezek definícióihoz szükségünk lesz bizonyos műveleti tulajdonságok ismeretére.

Műveleti tulajdonságok *Azt mondjuk, hogy a $*$ művelet*

- ***Kommutatív**, ha $x * y = y * x$ minden x, $y \in X$ esetén;*
- ***Asszociatív**, ha $x * (y * z) = (x * y) * z$ minden x, y, $z \in X$ esetén;*

Fontos eleme egy algebrai struktúrának az úgynevezett **egységelem**, illetve szükségünk lesz még egy elem inverzének a fogalmára.

Egységelem és elem inverze *Egy $X = (X, *)$ struktúra $e \in X$ elemét **egységelem**nek nevezzük, ha minden $x \in X$ esetén*

$$e * x = x * e = x.$$

Ha egy struktúrának van egységeleme, akkor az egyértelműen létezik.

Legyen $X = (X, *)$ egy egységelemes struktúra (e az egységelemmel). Ha egy $x \in X$ elemhez létezik olyan $y \in X$ elem, amelyre

$$x * y = y * x = e,$$

*akkor azt mondjuk, hogy az y elem az x **elem inverze**.*

Ha az $X = (X, *)$ struktúrában a $*$ művelet asszociatív és egy $x \in X$ elemnek van inverze, akkor az egyértelműen létezik.

A 4 fontos struktúratípus közül kettőt máris tudunk definiálni:

Félcsoport, csoport *Egy $X = (X, *)$ struktúrát*

- ***Félcsoport**nak nevezzük, ha a $*$ művelet asszociatív;*
- ***Csoport**nak nevezzük, ha asszociatív, létezik egységeleme és minden elemének van inverze.*

Egy egységelemes félcsoportot monoidnak, egy kommutatív csoportot Abel csoportnak nevezünk. Ha $M = M(*, 1)$ egy monoid, akkor az M invertálható elemei csoportot alkotnak, amit az M monoid egységcsoportjának nevezzük.

Ha a binér algebrai műveletet $+$ módon jelöljük (összeadás), akkor additív írásmódról beszélünk. Ha a műveletet $\cdot$ módon jelöljük (szorzás), akkor multiplikatív írásmódról beszélünk.

Additív írásmód esetén az egységet 0-val jelöljük és nullának vagy zérusnak nevezzük, illetve egy x elem additív inverzét $-x$ módon jelöljük.

Multiplikatív írásmód esetén az egységet 1-gyel jelöljük és egy-nek olvassuk. Egy x elem multiplikatív inverzét x^{-1} módon jelöljük.

További műveleti tulajdonságok

Ha $X = X(+, \cdot)$ egy kétműveletes algebrai struktúra, akkor a azt mondjuk, hogy

- ***A szorzás az összeadásra nézve balról disztributív**, ha $x(y + z) = xy + xz$ minden x, y, $z \in X$ esetén.*
- ***A szorzás az összeadásra nézve balról disztributív**, ha $(x + y)z = xz + yz$ minden x, y, $z \in X$ esetén.*

Mivel gyakran találkozunk olyan kétműveletes struktúrákkal, amelyekben a szorzás kommutatív, így ha az egyik irányú disztributivitás teljesül, akkor a másik is, így na kettő között nem teszünk különbséget és csak annyit mondunk, hogy a szorzás disztributív az összeadásra nézve.

Gyűrű

*Egy $R = R(+, \cdot)$ kétműveletes algebrai struktúrát **gyűrű**nek nevezünk, ha rendelkezik a következő tulajdonságokkal:*

- *$R(+)$ (kommutatív) csoport;*
- *$R(\cdot)$ egy félcsoport;*
- *A szorzás az összeadásra nézve disztributív.*

Ha a gyűrű további tulajdonságokkal rendelkezik, akkor különféle jelzőkkel illetjük. Egy gyűrűt kommutatívnak nevezzük, ha a szorzás kommutatív, egységelemesnek nevezzük, ha a szorzásnak van egységeleme, illetve zérusosztómentesnek nevezzük, ha nincs benne zérusosztó. (Az x, y zérustól különböző gyűrűbeli elemeket zérusosztóknak nevezzük, ha $x \neq 0$, $y \neq 0$, de $xy = 0$.) A kommutatív, egységelemes zérusosztómentes gyűrűket **integritástartomány**nak nevezzük. Integritástartományra szép példa az egész számok gyűrűje. (Az integritástartomány rendelkezik azokkal a tulajdonságokkal, amelyek szükségesek ahhoz, hogy benne számelméletet építsünk fel.)

Könnyű látni, hogyha R egy egységelemes gyűrű, és a szorzás az összeadásra nézve balról is és jobbról is kommutatív, akkor az összeadás szükségszerűen kommutatív (HF, szorgalmi).

Végül eljutottunk a test fogalmához:

Test *Egy $F = F(+, \cdot)$ kézműveletes algebrai struktúrát testnek nevezzük, ha*

- *F egy kommutatív egységelemes gyűrű;*
- *F minden nemzérus elemének van multiplikatív inverze.*

Testekre már ismerünk példákat: $\mathbb{Q}(+,\cdot)$ a racionális számok teste, $\mathbb{R}(+,\cdot)$ a valós számok teste, $\mathbb{C}(+,\cdot)$ a komplex számok teste mind olyan test, amelyekről már tanultunk. A Kvaterniók már nem testet, hanem úgynevezett ferdetestet alkotnak, mivel benne a szorzás már nem kommutatív. Érdemes megjegyezni, hogy rengeteg köztes test van a $\mathbb{Q}$ és az $\mathbb{R}$ között. Például a $\mathbb{Q}(\sqrt{2}) := \left\{a + b\sqrt{2} \mid a \in \mathbb{Q}, b \in \mathbb{Q}\right\}$ szintén test (biz: HF. szorgalmi).

0.4.5. Modulo m maradékosztálygyűrű

A $(\mathbb{Z}, +, \cdot)$ (egész számok gyűrűje) egy gyűrű. Ha $m \in \mathbb{Z}_+$, $m > 1$, akkor $(\mathbb{Z}_m, +, \cdot)$ szintén gyűrű, amit $\bmod(m)$ *maradékosztály gyűrűnek nevezünk.*

Ha az m modulus reducibilis (azaz összetett szám), akkor a $\mathbb{Z}_m$ maradékosztálygyűrű nem zérusosztómentes. Például $\mathbb{Z}_6$-ban a $\underline{2}$ és a $\underline{3}$ zérusosztók.

Feladat *Keressük meg az invertálható elemek $\mathbb{Z}_6$-ban és határozzuk meg az inverzeiket.*

x	0	1	2	3	4	5
x^{-1}	-	1	-	-	-	5

Két invertálható elem van, az 1 és az 5. $1^{-1} = 1$, $5^{-1} = 5$.

Általában elmondható, hogyha $\mathbb{Z}_m$ maradékosztálygyűrűben $\underline{a}$ elemnek pontosan akkor létezik multiplikatív inverze, ha $\mathrm{lnko}(a, m) = 1$. *A most bevezetett multiplikatív inverz fogalma azonos a azzal a multiplikatív inverz fogalommal, amelyet a későbbiekben a lineáris kongruenciaegyenlet segítségével fogunk bevezetni. A Zh-ban ez utóbbi fogalmat kérjük. (Némileg kevesebb előkészületet igényel.)*

A leírtakból következik az alábbi egyszerű tétel.

Ha p prím, akkor $\mathbb{Z}_p$ test.

A fenti tétel alkalmat ad arra, hogy véges testeket konstruáljunk. A legkisebb elemszámú test a kételemű $\{0, 1\}$ amelyben a műveletek $\bmod 2$ szerint vannak definiálva.

Példa $p = 5$. $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$.

A műveletek Cayley táblázata $\mathbb{Z}_5$-ben.

$+$	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

$\cdot$	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	3	1	4	2

A szorzás Cayley művelettáblázatából kapjuk, hogy $\mathbb{Z}_5$-ben minden nemzérus elemnek van multiplikatív inverze, tehát $\mathbb{Z}_5$ valóban test.

0.4.6. Lineáris kongruencia egyenletek

Feladat *Tekintsük a* $3x \equiv 2$ *egyenletet. Oldjuk meg ezt az egyenletet különböző maradékosztálygyűrűkben.*
$m = 4$*:* $3 \cdot 0 \equiv 0$, $3 \cdot 1 \equiv 3$, $3 \cdot 2 \equiv 2$, $3 \cdot 3 \equiv 1$, *tehát nincs megoldás.*
$m = 5$*:* $3 \cdot 0 \equiv 0$, $3 \cdot 1 \equiv 3$, $\boxed{3 \cdot 2 \equiv 2}$, $3 \cdot 3 \equiv 1$, $3 \cdot 4 \equiv 2$, *tehát megoldható a kongruencia egyenlet.*
$m = 5$*:* $3 \cdot 0 \equiv 0$, $3 \cdot 1 \equiv 3$, $3 \cdot 2 \equiv 2$, $3 \cdot 3 \equiv 1$, $3 \cdot 4 \equiv 2$, $3 \cdot 5 \equiv 3$, *tehát megint nincs megoldás. Nincs megoldás.*

Következzenek a formális definíciók és a tételek.

Lineáris kongruencia egyenlet *Legyenek* $a, b \in \mathbb{Z}$, $m \in \mathbb{Z}_+$ *úgy, hogy* $m > 1$. *Ekkor az*

$$ax \equiv b \mod (m) \tag{LKE}$$

egyenletet lineáris kongruencia egyenletnek nevezzük.

A lineáris kongruencia egyenletnek keressük az összes egész szám megoldását, ha egyáltalán megoldható. A megoldhatóság kritériuma viszonylag egyszerűen megoldható.

Másrészt, ha megoldható a lineáris kongruencia egyenlet és egy z_1 egy egész megoldása, a z_2 pedig olyan, hogy $z_1 \equiv z_2 \mod (m)$, akkor z_2 is megoldás. Tehát elég megkeresnünk az egymással páronként modulo m inkongruens megoldásokat. Azt is előírhatjuk, hogy minden egyes ilyen megoldás nemnegatív, de $m-1$-nél kisebb vagy egyenlő legyen. Ezeket a számokat nevezzük *alaprendszernek*. Az alaprendszer birtokában az összes megoldást ismerjük, mivel ha találunk egy tetszőleges megoldást, akkor az az alaprendszer valamelyik tagjával lesz kongruens modulo m.

A(z) (LKE) megoldhatósága és a megoldás szerkezete *Tekintsük az* $ax \equiv b \mod (m)$ *(LKE)-t.*

- ***A(z) (LKE) egyenlet megoldhatóságának a kritériuma:*** *A(z) (LKE) pontosan akkor oldható meg, ha* $\operatorname{lnko}(a, m) | b$.
- ***A megoldás szerkezete:*** *Megoldhatósága esetén a lineáris kongruencia egyenlet egy alaprendszere*

$$x_0 = x^* \cdot \frac{b}{d^*} \mod (m)$$
$$x_i = x_0 + i \cdot \frac{m}{d^*} \mod (m) \qquad (i = 1, 2, \ldots, d^* - 1),$$

ahol $d^* = \operatorname{lnko}(a, m)$ *és az* x^*, y^* *olyan egész számok, amelyekkel* $x^*a + y^*b = d^*$. *(Az* y^* *számot nem használjuk fel az alaprendszer felírásakor.)*

Ebből az is látszik, hogy az (LKE)-nek megoldhatósága esetén $d^* = \operatorname{lnko}(a, m)$ *elemű az alaprendszere (azaz az egymással modulo* m *inkongruens megoldásainak a száma.)*

A multiplikatív inverz *Legyen $a \in \mathbb{Z}$, $m \in \mathbb{Z}_+$, $m > 1$ olyanok, hogy* $\text{lnko}(a, m) = 1$ *(azaz relatív prímek). Ekkor az* $ax \equiv 1 \mod (m)$ *lineáris kongruencia egyenlet modulo m egyértelműen létező megoldását az a szám m modulusra vonatkozó* ***multiplikatív inverz****ének nevezzük és* a^{-1} *módon jelöljük.*

Érdemes megjegyezni, hogy:

- *Amikor multiplikatív inverzről beszélünk, mindig meg kell mondani, hogy a multiplikatív inverz milyen modulusra vonatkozik.*
- *Multiplikatív inverz keresésekor is mindig a legkisebb pozitív egész multiplikatív inverzet keressük.*
- *Ha egy a egész szám multiplikatív inverze adott m modulusra a b pozitív egész, az azt jelenti, hogy létezik olyan k egész szám, amelyre $ab = km + 1$.*

0.4.7. Példa lineáris kongruencia egyenlet megoldására

Feladat *Oldjuk meg a* $84x \equiv 16 \mod (44)$ *lineáris kongruencia egyenletet kibővített euklideszi algoritmussal.*

k	a	m	q	r	d	x^*	y^*
0	*84*	*44*	*1*	*40*	*4*	*-1*	$1 - 1 \cdot (-1) = 2$
1	*44*	*40*	*1*	*4*	*4*	*1*	$0 - 1 \cdot 1 = -1$
2	*40*	*4*	*10*	*0*	*4*	*0*	$1 - 10 \cdot 0 = 1$
3	*4*	*0*	-	-	*4*	*1*	*0*

Ellenőrzés: $(-1) \cdot 84 + 2 \cdot 44 = 4$.

Alaprendszer

$$x_0 = x^* \cdot \frac{b}{d^*} = (-1) \cdot \frac{16}{4} = -4 \equiv 40 \qquad \mod (44)$$
$$x_i = x_0 + i\frac{m}{d^*} = 40 + i\frac{44}{4} = 40 + i \cdot 11 \qquad \mod (44) \qquad (i = 1, 2, 3),$$

azaz $x_0 = 40$, $x_1 = 7$, $x_2 = 18$, $x_3 = 29$.

T *anultunk két hasonló fogalmat, a* $\text{mod}(\cdot, \cdot)$ *függvényt és az* $a \equiv b \mod (m)$ *relációt. Nyílván mást jelent a kettő, ugyanis a nevükben is benne van, hogy az első egy függvény, a második egy reláció. mégis van közöttük kapcsolat, mivel ha* $a \equiv b \mod (m)$ *és* $0 \leqslant b < m$, *akkor* $\text{mod}(a, m) = b$.

0.5. Az RSA algoritmus

RSA algoritmust Ron Rivest, Adi Shamir és Leonard Adleman által kidolgozott titkosítási eljárás. Az eljárás matematikai alapja az, hogy a prímfaktorizáció, azaz egy nagy egész szám prímszámok szorzataként való felbontása a jelenleg rendelkezésre álló számítógépekkel és algoritmusokkal nem megoldható.

A szükséges matematikai előismeret annak a ténynek az ismerete, hogy ha p, q prímszámok, a egy olyan egész szám, amelyre $\text{lnko}(a, pq) = 1$, akkor

$$a^{(p-1)(q-1)} \equiv 1 \qquad \mod (pq)$$

Ettől a képlettől működik az RSA algoritmus. Az RSA algoritmus az az algoritmus, amelyet ha megismerünk, némi nagyképűséggel mondhatjuk, hogy erre akár mi magunk is rájöhettünk volna. Utólag persze sok minden egyszerű. (Van ami utólag sem.) Az RSA algoritmus primitív, de éppen a primitívségében rejlik a szépsége és nagyszerűsége.

Ismét egy kicsit megkapargatjuk a felszínt, hogy megértsük a háttérben megbúvó matematikát. Enélkül csak annyi marad meg az egész algoritmusból, hogy megadtunk egy számot, csináltunk vele valami hókusz-pókuszt, lett belőle egy másik szám, azzal is csináltunk valami hókusz-pókuszt és visszakaptuk az eredeti számot. Az eredeti szám úgy jön ki a számolások végén, mint ahogy a bűvész húzza ki a nyulat a cilinderből.

Ebben a gyakorlatban egy kicsit körüljárjuk a matematikai hátteret, nem kötelező velem tartaniuk, de mint tudják, "gondolkodni menő". Akiket a matematikai háttér nem érdekel, azok elégedjenek meg az $a^{(p-1)(q-1)} \equiv 1 \mod (pq)$ összefüggéssel, és a MODHAT algoritmus működésével, és fogják tudni kézzel futtatni az RSA algoritmust.

Akiket érdekel, hogy mitől működik, azoknak adok egy kis segítséget a megéréshez, de ez nem lesz számon kérve, ez csak egy lehetőség érdeklődő hallgatók számára.

A következő felépítést követjük:

- A MODHAT (moduláris hatványozás algoritmus ismertetése;
- Az RSA algoritmus ismertetése;
- Az RSA algoritmus matematikai alapjai (kihagyható);
- Primtesztek, Carmichael számok;

0.5.1. A moduláris hatványozás algoritmusa

A MODHAT algoritmus az $\text{mod}(a^b, n)$ értékét számolja ki.
INPUT: $a \in \mathbb{Z}_+$, $b \in \mathbb{Z}_+$, $n \in \mathbb{Z}_+$. A b szám a 2-es számrendszerben

$$b = b_k b_{k-1}, \ldots, b_0 \qquad \text{azaz} \qquad b = \sum_{i=0}^{k} b_i 2^k.$$

OUTPUT: $c = \mathrm{mod}(a^b, n)$.

	MODHAT(a,b,m $\|\|$ c)
1.	$c \longleftarrow 1$
2.	FOR $i \longleftarrow k$ DOWNTO 0 DO
3.	$\quad c \longleftarrow \mathrm{mod}(c^2, n)$
4.	$\quad$ IF $b_i = 1$ THEN
5.	$\quad\quad c \longleftarrow \mathrm{mod}(ca, n)$
6.	RETURN(c)

1. Példa. *Számoljuk ki a* $\mathrm{mod}(63^{90}, 17)$ *értékét.*
először átírjuk a 90-et 2-es számrendszerbe. $90 = 1011010_{②}$. *Ezt követően alkalmazzuk a MODHAT algoritmust.*

b_i	$\mathrm{mod}(c^2, 17)$	$\mathrm{mod}(63c, 17)$
1	$\mathrm{mod}(1^2, 17) = 1$	$\mathrm{mod}(63 \cdot 1, 17) = 12$
0	$\mathrm{mod}(12^2, 17) = 8$	—
1	$\mathrm{mod}(8^2, 17) = 13$	$\mathrm{mod}(63 \cdot 13, 17) = 3$
1	$\mathrm{mod}(3^2, 17) = 9$	$\mathrm{mod}(63 \cdot 9, 17) = 6$
0	$\mathrm{mod}(6^2, 17) = 2$	—
1	$\mathrm{mod}(2^2, 17) = 4$	$\mathrm{mod}(63 \cdot 4, 17) = 14$
0	$\mathrm{mod}(14^2, 17) = 9$	—

Tehát $\mathrm{mod}(63^{90}, 17) = 9$.

2. Példa. *Számoljuk kia a* $\mathrm{mod}(2^{40}, 41)$ *értékét.*
először átírjuk a 40-et 2-es számrendszerbe. $40 = 1010000_{②}$. *Ezt követően alkalmazzuk a MODHAT algoritmust.*

b_i	$\mathrm{mod}(c^2, 41)$	$\mathrm{mod}(2c, 41)$
1	$\mathrm{mod}(1^2, 41) = 1$	$\mathrm{mod}(2 \cdot 1, 41) = 2$
0	$\mathrm{mod}(2^2, 41) = 4$	—
1	$\mathrm{mod}(4^2, 41) = 16$	$\mathrm{mod}(2 \cdot 16, 41) = 32$
0	$\mathrm{mod}(32^2, 41) = 40$	—
0	$\mathrm{mod}(40^2, 41) = 1$	—
0	$\mathrm{mod}(1^2, 41) = 1$	—

Tehát $\mathrm{mod}(2^{40}, 17) = 1$, *de ez számolgatás nélkül is látható a kis Fermat tétel alapján.*

0.5.2. Az RSA algoritmus

Alice üzenetet ír Bobnak.

- M az üzenet;

- C a titkosított üzenet;

ugyanis Alice azt szeretné, ha az üzenetét csak Bob tudná dekódolni. Titkosítani a P, dekódolni az S függvénnyel fogunk. Azaz

$$P(M) = C, \qquad S(C) = M.$$

A P és az S függvények egymás inverzei, ugyanis

$$M = S(C) = S(P(M)) = S \circ P(M).$$

A szöveges üzenetből először egy bitsorozatot készítünk, ami a kettes számrendszerben egy pozitív egész szám. Ez a szám az M. Titkosítani az n és e pozitív egészekkel fogunk. Ezek nyilvánosak, mindenki számára láthatóak, **tehát az n és e számok segítségével mindenki tud titkosítani**. Fontos még, hogy $\text{lnko}(M, n) = 1$ teljesüljön, azonban ez könnyen elérhető, (az n két nagy prímszám szorzata, ha ezeknél a prímeknél kisebb az M szám, akkor biztosan teljesül, hogy $\text{lnko}(M, n) = 1$. Ebből arra következtethetünk, hogy hosszú üzenet titkosítására nincs lehetőségünk, de ez nincs így, a hosszú üzenetet részekre bontjuk, a részek már titkosíthatók.) A titkosítás a következő módon történik:

$$C = P(M) = \text{mod}(M^e, n)$$

(szavakba öntve: az M et az e-edik hatványra emeljük és vesszük a kapott szám n modulus szerinti maradékát.) Ez a MODHAT algoritmussal könnyen elvégezhető, mivel a MODHAT algoritmus gondoskodik arról, hogy a hatvány "ne szálljon el", azaz végig kezelhető méretű maradjon.

Bob megkapja a titkosított üzenetet, azaz azt, hogy C. Most jön a csavar a történetben. Ahhoz, hogy **dekódolni** tudja az üzenetet, **szüksége van az n és d** számokra. A d az n és az e számokból számolható. Az n két prím szorzata, $n = pq$. A p és q prímeket csak Alice és Bob ismerik. A dekódoláshoz először ki kell számolni az f értékét:

$$f = (p-1)(q-1),$$

majd az d értékét. A d szám az e szám f modulusra vonatkozó multiplikatív inverze. Ehhez persze teljesülnie kell az $\text{lnko}(e, f) = 1$ feltételnek, ennek megfelelően az e számot eleve így kell megválasztanunk. A dekódolás a következő módon történik:

$$S(C) = \text{mod}(C^d, n).$$

A működés tényleg nagyon egyszerű, csak két dolgot kell megjegyezni hozzá:

- a d az e szám f-re vonatkozó multiplikatív inverze, azaz létezik olyan k egész szám, amellyel $de = 1 + kf$;

- $f = (p-1)(q-1)$, így alkalmazható a kis Fermat tételt követő következmény.

Ugorjon elő a nyúl a cilinderből: Idézzük fel, hogy $C = M^e$. Számoljuk ki a C^d n re vonatkozó osztási maradékát.

$$S(C) = C^d = (M^e)^d = M^{ed} = M^{1+kf} = M(M^{(p-1)(q-1)})^k \equiv M \mod (n)$$

és ezt kellett bizonyítani.

3. Példa. *Legyen $M = 65$ az üzenet, amit Alice küld a Bobnak, $n = 899$ és $e = 11$. Ezek nyilvánosak. Az n prímfaktorizációja szigorúan titkos, ezt csak Alice és Bob ismerik. $n = 899 = 29 \cdot 31$, azaz $p = 29$, $q = 31$. $f = (p-1)(q-1) = 28 \cdot 30 = 840$. Legyen $e = 11$. Ekkor $\text{lnko}(M, n) = \text{lnko}(65, 899) = 1$ és $\text{lnko}(e, f) = \text{lnko}(11, 840) = 1$, tehát teljesülnek a titkosítás és a dekódolás feltételei.*

Először titkosítsunk: $C = P(M) = (M^e, n)$, ami a moduláris hatványozás algoritmussal számolható.

b_i	$\text{mod}(c^2, 899)$	$\text{mod}(65c, 899)$
1	$\text{mod}(1^2, 899) = 1$	$\text{mod}(65 \cdot 1, 899) = 65$
0	$\text{mod}(65^2, 899) = 629$	—
1	$\text{mod}(629^2, 899) = 81$	$\text{mod}(65 \cdot 81, 899) = 770$
1	$\text{mod}(770^2, 899) = 459$	$\text{mod}(65 \cdot 459, 899) = 168$

Tehát a tikosított üzenet: $C = 168$.

Most nézzük meg, hogy Bob tudja-e dekódolni az üzenetet. Ehhez először számoljuk ki d-t, ami az e-nek az $f$re vonatkozó multiplikatív inverze. Ez kibővített euklideszi algoritmussal számolható.

h	e_k	f_k	q_k	r_k	d^*	x^*	y^*
0	11	840	0	11	1	-229	$3 - 0 \cdot (-229) = 3$
1	840	11	76	4	1	3	$-1 - 76 \cdot 3 = -229$
2	11	4	2	3	1	-1	$1 - (-1) \cdot 2 = 3$
3	4	3	1	1	1	1	$0 - 1 \cdot 1 = -1$
4	3	1	3	0	1	0	$1 - 3 \cdot 0 = 1$
5	1	0	–	–	*1*	*1*	*0*

Ellenőrzés: $11 \cdot (-229) + 840 \cdot 3 = 1$. *Tehát* $d = -229 + 840 = 611$.

A dekódoláshoz az $S(C) = \text{mod}(C^d, n)$, ami a moduláris hatványozás algoritmussal számolható.

b_i	$\text{mod}(c^2, 899)$	$\text{mod}(168c, 899)$
1	$\text{mod}(1^2, 899) = 1$	$\text{mod}(168 \cdot 1, 899) = 168$
0	$\text{mod}(168^2, 899) = 355$	—
0	$\text{mod}(355^2, 899) = 165$	—
1	$\text{mod}(165^2, 899) = 255$	$\text{mod}(168 \cdot 255, 899) = 587$
1	$\text{mod}(587^2, 899) = 252$	$\text{mod}(168 \cdot 252, 899) = 83$
0	$\text{mod}(83^2, 899) = 596$	—
0	$\text{mod}(596^2, 899) = 111$	—
0	$\text{mod}(111^2, 899) = 634$	—
1	$\text{mod}(634^2, 899) = 103$	$\text{mod}(168 \cdot 103, 899) = 223$
1	$\text{mod}(223^2, 899) = 284$	$\text{mod}(168 \cdot 284, 899) = 65$

Tehát az eredeti üzenet "$M = 65$" volt.

0.5.3. Matematikai alapok az RSA algoritmus működéséhez

Mint ahogy azt a korábbi részben említettük, az RSA algoritmus működése az

$$a^{(p-1)(q-1)} \equiv 1 \qquad \mod (pq)$$

képlet ismerete, ahol p, q prímszámok, a egy olyan egész szám, amelyre $\text{lnko}(a, pq) = 1$. Most két bizonyítást adunk az állítása. (Ez az a bizonyos kihagyható rész.) Az első bizonyítás a kis Fermat tételen keresztül vezet, a második esetén a bizonyítandó állítás az Euler-Fermat tétel speciális esete.

A kis Fermat tételben lévő kis jelző nem arra utal, hogy a felfedezés a gyermek Fermatnak tulajdonítható (hasonlatosan a kis Gauss összefüggéshez, ami nem más, mint a számtani sorozat összegképlete), hanem ezzel a jelzővel különbözteti meg a tudománytörténet a Fermat által talált és bizonyított összefüggést az úgynevezett nagy Fermat tételtől. Erről a tételről ennek a résznek a végén mondunk pár szót.

Pierre de Fermat (Beaumont-de-Lomagne, 1601. augusztus 17. – Castres, 1665. január 12.) francia jogász, műkedvelő matematikus. Jogászként dolgozott, nem akart kapcsolatba kerülni olyan emberekkel, akikkel később peres ügyekkel kapcsolatban találkozhat. Ezért hobbiként a matematikát választotta, amelyben maradandót alkotott.

Leonhard Euler (Bázel, 1707. április 15. – Szentpétervár, 1783. szeptember 18.) svájci származású matematikus és fizikus. Az egyik legnagyobb (talán a legnagyobb) matematikus. 1771-ben mindkét szemére megvakult, tudományos eredményeinek csaknem felét vakon hozta létre. Vajon Euler hallgathatta-e Haydn zenéjét? Igen, hiszen kortársak voltak.

Most átnézzük az $a^{(p-1)(q-1)} \equiv 1 \mod (pq)$ (p, q különböző prímek, $\text{lnko}(a, pq) = 1$) képlet igazolásához vezető két utat. Az első a kis Fermat tételen keresztül vezet. Ezt szokták javasolni műszaki egyetemeken. Kevés előismeretet igényel, azonban itt is kell egy kicsit számolgatni. Kezdjük ezzel az úttal. Pontokba szedtük a bizonyítás lépéseit:

1. Ha p prím, $a \in \mathbb{Z}_+$, akkor $a^p \equiv a$ modulo (p). (Tehet ehhez az állításhoz még arra sincs szükségünk, hogy az a és p számok relatív prímek legyenek.)

Útmutatás *Az a szám szerinti teljes indukcióval bizonyítunk. Az állítás $a = 0$, 1 -1 esetén nyilvánvaló. Innen akár fölfelé, akár lefelé teljes indukcióval bizonyítható. Az öröklődés részhez elegendő észrevenni, hogyha p prím, akkor*

$$p \Big| \binom{p}{k} = \frac{p(p-1)\dots(p-k+1)}{1 \cdot 2 \dots k} \qquad (k = 1, \dots, p-1).$$

Így a binomiális tételt is felhasználva kapjuk, hogy

$$(a+1)^p = a^p + \binom{p}{1} a^{p-1} + \dots + \binom{p}{p-1} a + 1 \equiv a + 1 \qquad \mod (p),$$

Ha lefelé megyünk, akkor a $p = 2$ esete külön kell vizsgálnunk. Illetve, ha p egy páratlan prím, akkor a fent leírtakkal analóg módon kapjuk, hogy $(a-1)^p \equiv a - 1$ $\mod (p)$.

2. Ha $ac \equiv bc \mod (n)$, akkor $a \equiv b \mod (\frac{n}{\text{lnko}(n,c)}$.

Útmutatás *Az állítás egyszerű számolás segítségével könnyen ellenőrizhető. (A bizonyítás* ***szorgalmi feladat****.)*

Ebből az állításból már könnyen kijön a kis Fermat tétel. A kis Fermat tételt azért hívjuk kis Fermat tételnek, hogy megkülönböztessük a nagy Fermat sejtéstő, ami mára már nagy Fermat tétel, erről később még írunk.

3. Kis Fermat tétel Ha p prím, $a \in \mathbb{Z}$ úgy, hogy $\text{lnko}(a,p) = 1$ (azaz a p prím nem osztója az a számnak), akkor

$$a^{p-1} \equiv 1 \mod (p).$$

Ismét szükségünk van egy kis állításra.

4. Ha $a \equiv b \mod (n_1)$, $a \equiv b \mod (n_2)$, akkor $a \equiv b \mod (\text{lkkt}(n_1, n_2))$.

Útmutatás *Egyszerű számolással könnyen kijön (****Szorgalmi feladat.****)*

5. Ha p, q prímek, $\text{lnko}(a, pq) = 1$, akkor $a^{(p-1)(q-1)} \equiv 1 \mod (pq)$.

most nézzünk egy kicsit elegánsabb utat, ami az Euler Fermat tételen alapul.

Az Euler Fermat tétel bizonyítása mindössze egy sor, azonban igényel egy kevés előismeretet. Most a szükséges fogalmakat és előismereteket szedjük pontokba:

1. Csoport rendje, elem rendje. Ha $(G, \cdot)$ egy csoport akkor a G csoport rendjének a G halmaz számosságát nevezzük és $|G|$ módon jelöljük. Csoport rendje lehet véges, vagy végtelen. Például a $(\mathbb{Z}, +)$, tehát az egész szók halmaza az összeadás műveletére nézve egy végtelen rendű csoport.

Az elem rendjének a fogalmához induljunk ki egy $(G, \cdot)$ legyen $g \in G$ egy egységelemtől különböző elem. Képezzük a g elem hatványaiból alkotott

$$g, \quad g^2, \quad g^3, \quad g^4, \quad g^5, \ldots$$

sorozatot. Ekkor két eset van. A sorozat tagjai páronként különbözőek. Ekkor azt mondjuk, hogy a g elem rendje végtelen.

a másik eset az, amikor ennek a sorozatnak van két egyenlő eleme, mondjuk léteznek olyan k, l pozitív egészek, amelyekre $k < l$ és $g^k = g^l$. Ekkor, $g^{l-k} = 1$, azaz van olyan pozitív egész z, amelyre $g^z = 1$. Ekkor a g elem rendjét

$$|g| := \min\{z \in \mathbb{Z}_+ | g^z = 1\}$$

módon értelmezünk.

Ha a G egy véges rendű csoport, akkor nem állhat fenn az az eset, hogy egy tetszőleges egységtől különböző elem pozitív egész kitevőjű hatványai páronként különbözők legyenek, így egy véges rendű csoport minden egységtől különböző eleme véges rendű.

Az is könnyű látni (a hatványok ciklikus ismétlődése miatt), hogyha valamely n-re $g^n = 1$, akkor a g rendje osztója az n-nek, bár erre az összefüggésre nem lesz szükségünk.

Most nézzünk példát véges rendű csoportra. Tudjuk, hogy rögzített m modulus esetén a $\mathbb{Z}_m$ a szorzásra nézve félcsoportot alkot. Ebből a félcsoportból vegyük ki az 1-et és azokat a

maradékosztályokat, amelyeknek van multiplikatív inverzük. Tanultuk, hogy egy a elemnek pontosan akkor van multiplikatív inverze az m modulusra nézve, ha a és m relatív prímek. Tehát egy $\underline{a}$ maradékosztálynak pontosan akkor van multiplikatív inverze, ha valamelyik (és ebben az esetben minden eleme) relatív prím m-mel.

Általában is igaz, ha egy tetszőleges $(S, \cdot)$ félcsoportban összegyűjtjük az 1-et és az invertálható elemeket, akkor a kapott elemek egy csoportot alkotnak, ugyanis, ha az a és b elemeknek van inverzük, akkor az ab elemeknek is van, mivel $(ab)^{-1} = b^{-1}a^{-1}$.

Tehát, ha rögzített m modulus esetén a $(\mathbb{Z}_m, \cdot)$ félcsoportból indulunk ki, akkor az invertálható elemek által alkotott csoportot **redukált maradékosztálycsoport**nak vevezzük. A kapott csoportnak annyi eleme van, ahány elem relatív prím m-hez a

$$0, \quad 1, \quad 2, \quad \ldots, \quad m-1$$

sorozat tagjai közül. Azt a függvényt, amely egy m pozitív egész számhoz a $0, 1, 2, \ldots, m-1$ sorozatban az m-hez relatív prímek számát rendeli Euler-féle φ függvénynek nevezzük és $\varphi(m)$ módon jelöljük.

Tekintsük az $m = 12$ elemet, és határozzuk meg a redukált maradékosztálycsoport elemeit.

$$0, \quad \boxed{1}, \quad 2, \quad 3, \quad 4 \quad \boxed{5}, \quad 6 \quad \boxed{7}, \quad 8 \quad 9, \quad 10, \quad \boxed{11}.$$

Nyilván szebb lett volna egy nagyobb alapszám, mondjuk a 42 használatával, de tovább tartana begépelni. Tehát a modulo 12 redukált maradékosztálycsoport elemei $\{1, 5, 7, 11\}$, a csoport Cayley táblázata:

$+$	1	5	7	11
1	1	5	7	11
5	5	1	11	7
7	7	11	1	5
11	11	7	5	1

Tehát a modulo 12 redukált maradékosztálycsoportnak $\varphi(12) = 4$ eleme van. Még két apró lépés kell a bizonyításhoz.

2. Legyen $(G, \cdot)$ egy csoport, H a G halmaz egy nemüres részhalmaza. Az mondjuk, hogy H egy részcsoportja a G csoportnak, ha H maga is csoport (G-beli csoportműveletre nézve), azaz rendelkeznek a következő tulajdonsággal:

1. $1 \in H$;
2. Ha h_1, $h_2 \in H$, akkor $h_1h_2 \in H$, azaz a H halmaz zárt a csoportműveletre nézve;
3. Minden $h_1 \in H$ elemhez létezik olyan $h_2 \in H$ elem, amelyekkel $h_1h_2 = h_2h_1 = 1$, azaz minden H-beli elem elem inverze is H-beli.

3. Lagrange tétele Legyen $(G, \cdot)$ egy véges csoport. Ekkor

a. Részcsoport rendje osztója a csoport rendjének;

b. Elem rendje osztója a csoport rendjének.

Útmutatás
a. Szorgalmi házi feladat. *Nem valószínű, hogy a bizonyításhoz szükséges ötletre bárki segítség nélkül rájön, azonban maga a bizonyítás nagyon egyszerű, minden elemi algebra könyv tartalmazza.*
b. *már nagyon könnyen kijön az* ***a.*** *állításból. Ugyanis, ha $g \in G$ egy tetszőleges elem, amelynek rendjét jelölje k, akkor*

$$\langle g \rangle >= \{1,\ g,\ \ldots, g^{k-1}\}$$

egy részcsoportja G. Ezt a részcsoportot a g elem által generált csoportnak nevezzük. Az egy elem által generált csoportot, tehát azt a legszűkebb csoportot, ami ezt az elemet tartalmazza ***ciklikus csoport****nak nevezzük. Ciklikus csoport rendje egyenlő a generáló elmének a rendjével, így alkalmazható a Lagrange tétel* ***a.*** *része.*

Joseph-Louis Lagrange gróf, eredeti olasz nevén Giuseppe Luigi Lagrangia (Torino, 1736. január 25. – Párizs, 1813. április 10.) olasz születésű francia matematikus; a számelmélet, a matematikai analízis és az égitestek mechanikája területén elért eredményeiről híres. Csendes, a tudománynak élő tudós volt. Egy, a nevét viselő tétellel (a Roll- , Lagrange-, Cauchy tételek középső tagja) már alkalmunk volt megismerkedni az Analízis I tárgy keretében.

Euler-Fermat tétel *Ha $a \in \mathbb{Z}$, $n \in \mathbb{Z}_+$, $n > 1$, $\text{lnko}(a, n) = 1$, akkor*

$$a^{\varphi(n)} \equiv 1 \qquad \mod (n)$$

Útmutatás *Legyenek a és m relatív prímek. Ekkor az a elemmel reprezentált maradékosztály eleme a $\varphi(m)$ rendű redukált maradékosztály csoportnak. Felhasználva Lagrange tételét, mely szerint elem rendje osztója csoport rendjének, bebizonyítottuk a tételt.*

Az Euler Fermat tétel speciális esete az általunk bizonyítandó állítás, mivel ha p és q két (különböző prímszám), a pdig egy pozitív egész, amelyre $\text{lnko}(a, pg) = 1$, akkor mivel $\varphi(pq) = (p-1)(q-1)$, így az Euler-Fermat tétel egyszerű következményeként kapjuk a bizonyítandó állítást.

A φ egy számelméleti függvény, amely tetszőleges n nemnegatív egész számhoz a 0, 1, $\ldots$, $n-1$ sorozatban n-hez relatív prímek számát rendeli. A φ függvény egy úgynevezett multiplikatív számelméleti függvény, ami azt jelenti, hogyha a és b relatív prímek, akkor $\varphi(ab) = \varphi(a)\varphi(b)$. Sajnos ez a tény nem segít a φ függvény értékének a kiszámításában, ugyanis először meg kell határoznunk a φ függvény értékét tetszőleges n helyen, és ebből vezethető le az a tény, hogy a φ függvény multiplikatív.

Tétel *Legyen $n = p_1^{\alpha_1} \ldots p_n^{\alpha_k}$, ekkor*

$$\varphi(n) = n\left(1 - \frac{1}{p_1}\right) \ldots \left(1 - \frac{1}{p_k}\right)$$

Útmutatás *Szita formulával némi számolással kijön. (****Szorgalmi házi feladat****)*

Végül néhány gondolat a nagy Fermat tételről. A nagy Fermat tétel Fermat sejtésként kezdte földi pályafutását. Fermat foglalkozott az alábbi problémával: határozzuk meg az $x^n + y^n = z^n$ egész megoldásainak a halmazát, ahol $n > 2$ rögzített egész. Fermat sejtése az volt, hogy ennek az egyenletnek csak triviális (azaz 0, vagy 1) megoldásai vannak és egy könyv margójára azt írta (szabad fordításban), hogy ennek a ténynek egy csodálatosan szép megoldását találtam, de a hely kevés, hogy a bizonyítást befogadja. Eltelt vagy 300 év, amíg Sir Andrew John Wiles (Cambridge, 1953. április 11.) az Amerikai Egyesült Államokban élő angol matematikus bizonyíttota a nagy Fermat tételt. Eredményét 1993-ban publikálta, ami azonban hibásnak bizonyult. Wiles két évre bezárkózott a házába, és 1995-ben publikálta a hibátlan bizonyítást.

0.5.4. A kis Fermat tételen alapuló prímteszt

A kis Fermat tétel azt mondja ki, hogyha p prím és $\text{lnko}(a, p) = 1$, akkor $a^{p-1} \equiv 1 \mod (p)$. Ebből kontrapozícióval kapjuk, hogyha $\text{lnko}(a, p) = 1$ de $a^{p-1} \not\equiv 1 \mod (p)$. Ha egy pozitív egész megbukik egy ilyen prímtesten, akkor az biztosan összetett szám, tehát nem prím. Ha azonban átmegy egy ilyen prímteszten, akkor az még nem biztos, hogy prím. azok a számok, amelyek az összes kis Fermat tételen alapuló prímteszten átmennek, mégsem prímek, az úgynevezett Carmichael számok. Ilyen szám például az 561, amelyet maga Carmichael talált meg 1910-ben.

Definíció *A $p \in \mathbb{Z}_+$, $p > 1$ számot **álprím**nek nevezzük, ha létezik $a \in \mathbb{Z}_+$ úgy, hogy* $\text{lnko}(a, p) = 1$ *és*

$$a^{p-1} \equiv 1 \mod (p).$$

Charmichael számok *A $p \in \mathbb{Z}$, $p > 1$ számot Carmichael számnak nevezzük, ha*

$$a^{p-1} \equiv 1 \mod (p).$$

minden olyan $a \in \mathbb{Z}_+$ esetén, amelyre $\text{lnko}(a, p) = 1$, *de p nem prím, azaz a **Charmichael számok** azok az 1-nél nagyobb pozitív egész számok, amelyek minden olyan prímteszten átmennek, amelyek a kis Fermat tételen alapulnak, mégsem prímek.*

Charmichael számok: 561, 1105, 1729, 2465, 2821, 6601, 8911, ... végtelen sok van belőlük.

0.6. Dinamikus halmazok: tömb és láncolt lista implementáció

0.6.1. Dinamikus halmazok

4. Definíció. *A **dinamikus halmaz** egy olyan halmaz, amely az őt felhasználó algoritmus során változik (bővül, szűkül, módosul).*

5. Definíció. *A **sorozat adatstruktúra** objektumok olyan tárolási módja, amikor az elemek a műveletek által kijelölt lineáris sorrendben követik egymást. Tipikus műveletek: keresés, beszúrás, törlés.*

Sorozat implementálása

- tömb
- láncolt lista
  - egyszeresen vagy kétszeresen láncolt, szentineles

6. Definíció. ***Speciális sorozat adatstruktúrák**: verem (STACK), sor (QUEUE)*

7. Definíció. *A **verem** (STACK) olyan dinamikus halmaz, amelyben előre meghatározott az az elem melyet a TÖRÖL eljárással eltávolítunk. Ez az elem mindig az időben legutoljára a struktúrába helyezett elem lesz. (LIFO last in first out adat szerkezet). Műveletei: beszúrás: PUSH, törlés POP.*

8. Definíció. *A **sor** (QUEUE) olyan dinamikus halmaz, amelyben előre meghatározott az az elem, melyet TÖRÖL eljárással eltávolítunk, és az az elem is, amelyet a BESZÚR eljárással a halmazba teszünk. Törlésre mindig az elemek közül a legrégebben beszúrt kerül. A beszúrt elem lesz a legfrissebb. (FIFO szerkezet). Műveletei: beszúrás, törlés.*

0.6.2. Tömb

Legyen A egy tömb.
Attribútumok:
fej $[A]$ = az első elem indexe.
vége $[A]$ = az utolsó elem indexe.
hossz $[A]$ = a tömbelemek száma (a tárolt elemek száma).
tömbméret $[A]$ = annak a memóriaterületnek a nagysága tömbelem egységben mérve, ahová a tömböt elhelyeztük.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

Beszúrás tömbbe algoritmus: BESZ$(A, x, r,$ hibajelzés)
INPUT: A a tömb, x a tömb indexe, amely elem elé történik a beszúrás, r a beszúrandó elem.
OUTPUT: hibajelzés - a beszúrás eredményességét jelzi.

	BESZ$(A, x, r,$ hibajelzés)
1.	IF hossz $[A] \neq 0$
2.	THEN IF (fej $[A] \leqslant x \leqslant$ vége $[A]$) és (tömbméret $[A] >$ hossz $[A]$)
3.	THEN FOR $i \longleftarrow$ vége $[A]$ DOWNTO x DO
4.	$A[i+1] \longleftarrow A[i]$
5.	$A[x] \longleftarrow r$, INC(hossz$[A]$), INC(vége$[A]$)
6.	hibajelzés $\longleftarrow$ "sikeres beszúrás"
6.	ELSE hibajelzés $\longleftarrow$ "sikertelen beszúrás"
7.	ELSE fej $[A] \longleftarrow 1$, vége $[A] \longleftarrow 1$ hossz $[A] \longleftarrow 1$, $A[x] \longleftarrow r$
8.	RETURN(hibajelzés)

Példa:
TÖMB $A = [5, 9, 10, 23, 92, ., ., .]$
fej$[A] = 1$, vége$[A] = 5$, hossz$[A] = 5$, tömbméret$[A] = 8$.
Szúrjuk be az $x = 3$ helyre az $r = 8$-at. (BESZÚRÁS, $T(n) = \Theta(n)$)

1. lépés: van-e hely?
2. lépés: ciklus $i = 5 \ldots 3$ $A[i+1] \longleftarrow A[i]$
$A[6] \longleftarrow A[5]$ $A[5, 9, 10, 23, 92, 92, ., .]$
$A[5] \longleftarrow A[4]$ $A[5, 9, 10, 23, 23, 92, ., .]$
$A[4] \longleftarrow A[3]$ $A[5, 9, 10, 10, 23, 92, ., .]$
3. lépés: $A[3] \longleftarrow x$ $A = [5, 9, 8, 10, 23, 92, ., .]$
INC(vége$[A]$): vége$[A] = 6$.
INC(hossz$[A]$): hossz$[A] = 6$.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

Törlés tömbből algoritmus: TÖRL$(A, x,$ hibajelzés)
INPUT: A a tömb, x a törlendő elem indexe.
OUTPUT: hibajelzés - a törlés eredményességét jelzi.

	TÖRL$(A, x,$ hibajelzés)
1.	IF hossz $[A] \neq 0$
2.	THEN IF fej $[A] \leqslant x \leqslant$ vége $[A]$
3.	THEN FOR $i \longleftarrow x$ TO vége $[A] - 1$ DO
4.	$A[i] \longleftarrow A[i+1]$
5.	DEC(hossz$[A]$), DEC(vége$[A]$)
6.	hibajelzés $\longleftarrow$ "sikeres törlés"
7.	ELSE hibajelzés $\longleftarrow$ "nem létező elem"
8.	ELSE hibajelzés $\longleftarrow$ "üres tömb"
9.	RETURN(hibajelzés)

Példa:
TÖMB $A = [5, 9, 10, 23, 92, ., ., .]$
fej$[A] = 1$, vége$[A] = 5$, hossz$[A] = 5$, tömbméret$[A] = 8$.
Töröljük az eredeti tömb 3. elemét. (TÖRLÉS, $T(n) = \Theta(n)$)

1. lépés: fej$[A] \leqslant 3 \leqslant$ vége$[A]$

2. lépés: ciklus $i = 3 \ldots 4$ $A[i] \longleftarrow A[i+1]$
$A[3] \longleftarrow A[4]$ $A[5, 9, 23, 23, 92, ., ., .]$
$A[4] \longleftarrow A[5]$ $A[5, 9, 23, 92, 92, ., ., .]$

3. lépés:
DEC(vége$[A]$): vége$[A] = 4$.
DEC(hossz$[A]$): hossz$[A] = 4$.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

Lineáris keresés tömbben algoritmus: LINKER(A, k, x)
INPUT: A a tömb, k a keresett kulcs.
OUTPUT: x a k kulcsú elem indexe, vagy NIL, ha nincs ilyen elem..

	LINKER(A, k, x)
1.	IF Hossz[A]$\neq 0$
2.	THEN $x \longleftarrow$ fej $[A]$
3.	WHILE ($x \leqslant$ vége $[A]$) és $A[x] \neq k$) DO
4.	INC(x)
4.	IF ($x >$ vége $[A]$)
5.	THEN $x \longleftarrow$ NIL
6.	ELSE $x \longleftarrow$ NIL
7.	RETURN(x)

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

Bináris keresés rekurzív változata: BINKER(A, i, j, k, x)
INPUT: A a rendezett tömb, i a keresés kezdőindexe, j a keresés végindexe (i és j nem vesznek részt a keresésben).
OUTPUT: x, ahol x a k kulcsú elem indexe, ha $A[x] = k$, vagy NIL, ha k kulcsú elem nincs a tömbben.

	BINKER(A, i, j, k, x)
1.	$l \longleftarrow \lfloor \frac{i+j}{2} \rfloor$
2.	IF $i = l$ THEN $x \longleftarrow$ NIL, RETURN(x)
3.	ELSE IF $A[l] = k$
4.	THEN $x \longleftarrow l$, RETURN(x)
5.	ELSE IF $A[l] < k$
6.	THEN BINREK(A, l, j, k, x)
7.	ELSE BINREK(A, i, l, k, x)

Példa: (Megtalálja)
TÖMB $A = [5, 9, 10, 23, 92, ., ., .]$
fej$[A] = 1$, vége$[A] = 5$, hossz$[A] = 5$, tömbméret$[A] = 8$.
Keressük a $k = 23$ kulcsú elemet a tömbben.
Bin$(A, 0, 6, k = 23, x)$
$l \longleftarrow \lfloor \frac{0+6}{2} \rfloor = 3$
$i \stackrel{?}{=} l \quad (0 \stackrel{?}{=} 3) \quad$ hamis
$A[3] \stackrel{?}{=} 23 \quad$ hamis
$A[3] < 23$ igaz $\Rightarrow$ Bin$(A, 3, 6, 23, x)$
$l \longleftarrow \lfloor \frac{3+6}{2} \rfloor = 4.$
$i \stackrel{?}{=} l \quad (3 \stackrel{?}{=} 4) \quad$ hamis
$A[4] \stackrel{?}{=} 23$ igaz $\Rightarrow$ RETURN(4).
Példa: (Kisebb, mint a tömb elemei)
TÖMB $A = [5, 9, 10, 23, 92, ., ., .]$
fej$[A] = 1$, vége$[A] = 5$, hossz$[A] = 5$, tömbméret$[A] = 8$.
Keressük a $k = 1$ kulcsú elemet a tömbben.
Bin$(A, 0, 6, k = 1, x)$
$l \longleftarrow \lfloor \frac{0+6}{2} \rfloor = 3$
$i \stackrel{?}{=} l \quad (0 \stackrel{?}{=} 3) \quad$ hamis
$A[3] \stackrel{?}{=} 1 \quad$ hamis
$A[3] < 1$ hamis $\Rightarrow$ Bin$(A, 0, 3, 1, x)$
$l \longleftarrow \lfloor \frac{0+3}{2} \rfloor = 1$
$i \stackrel{?}{=} l \quad (0 \stackrel{?}{=} 1) \quad$ hamis
$A[1] = 1 \quad$ hamis
$A[1] < 1$ hamis $\Rightarrow$ Bin$(A, 0, 1, 1, x)$
$l \longleftarrow \lfloor \frac{0+1}{2} \rfloor = 0.$
$i \stackrel{?}{=} l \quad (0 \stackrel{?}{=} 0)$ igaz $\Rightarrow$ RETURN(NIL).
Példa:
TÖMB $A = [5, 9, 10, 23, 92, ., ., .]$ (Nagyobb, mint a tömb elemei)
fej$[A] = 1$, vége$[A] = 5$, hossz$[A] = 5$, tömbméret$[A] = 8$.
Keressük a $k = 100$ kulcsú elemet a tömbben.
Bin$(A, 0, 6, k = 100, x)$
$l \longleftarrow \lfloor \frac{0+6}{2} \rfloor = 3$
$i \stackrel{?}{=} l \quad (0 \stackrel{?}{=} 3) \quad$ hamis
$A[3] \stackrel{?}{=} 100 \quad$ hamis
$A[3] < 100$ hamis $\Rightarrow$ Bin$(A, 3, 6, 100, x)$
$l \longleftarrow \lfloor \frac{3+6}{2} \rfloor = 4$
$i \stackrel{?}{=} l \quad (3 \stackrel{?}{=} 4) \quad$ hamis
$A[4] = 100 \quad$ hamis
$A[4] < 100$ igaz $\Rightarrow$ Bin$(A, 4, 6, 100, x)$

$l \longleftarrow \lfloor \frac{4+6}{2} \rfloor = 5.$

$i \stackrel{?}{=} j \qquad (4 \stackrel{?}{=} 5) \qquad$ hamis

$A[5] = 100 \qquad$ hamis

$A[5] < 100$ igaz $\Rightarrow \mathrm{Bin}(A, 5, 6, 100, x)$

$l \longleftarrow \lfloor \frac{5+6}{2} \rfloor = 5.$

$i \stackrel{?}{=} j \qquad (5 \stackrel{?}{=} 5)$ igaz $\Rightarrow$ RETURN(NIL).

Példa:

TÖMB $A = [5, 9, 10, 23, 92, ., ., .]$ (Van kisebb elem is és nagyobb elem is a tömbben, de a keresett elem nincs a tömbben.)

fej$[A] = 1$, vége$[A] = 5$, hossz$[A] = 5$, tömbméret$[A] = 8$.

Keressük a $k = 8$ kulcsú elemet a tömbben.

$\mathrm{Bin}(A, 0, 6, k = 8, x)$

$l \longleftarrow \lfloor \frac{0+6}{2} \rfloor = 3$

$i \stackrel{?}{=} l \qquad (0 \stackrel{?}{=} 3) \qquad$ hamis

$A[3] \stackrel{?}{=} 8 \qquad$ hamis

$A[3] < 8$ hamis $\Rightarrow \mathrm{Bin}(A, 0, 3, 8, x)$

$l \longleftarrow \lfloor \frac{0+3}{2} \rfloor = 1$

$i \stackrel{?}{=} l \qquad (0 \stackrel{?}{=} 1) \qquad$ hamis

$A[1] \stackrel{?}{=} 8 \qquad$ hamis

$A[1] < 8$ igaz $\Rightarrow \mathrm{Bin}(A, 1, 3, 8, x)$

$l \longleftarrow \lfloor \frac{1+3}{2} \rfloor = 2.$

$i \stackrel{?}{=} l \qquad (1 \stackrel{?}{=} 2) \qquad$ hamis

$A[2] = 8 \qquad$ hamis

$A[2] < 8$ hamis $\Rightarrow \mathrm{Bin}(A, 1, 2, 8, x)$

$l \longleftarrow \lfloor \frac{1+2}{2} \rfloor = 1.$

$i \stackrel{?}{=} l \qquad (1 \stackrel{?}{=} 1)$ igaz $\Rightarrow$ RETURN(NIL).

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

Bináris keresés iteratív változata: $\mathrm{BINIT}(A, i, j, k, x)$

INPUT: A a rendezett tömb, i a keresés kezdőindexe, j a keresés végindexe (i és j nem vesznek részt a keresésben).

OUTPUT: x, ahol $x = k$ kulcsú elem indexe, ha k-t megtaláltuk, illetve NIL, ha k nincs benne

a tömbben.

	BINIT(A, i, j, k, x)
1.	$l \longleftarrow \lfloor \frac{i+j}{2} \rfloor$
2.	WHILE $i < l$ DO
3.	IF $A[l] = k$, THEN $x \longleftarrow l$, RETURN(x)
4.	IF $A[l] < k$ THEN $i \longleftarrow l$
5.	ELSE $j \longleftarrow l$
6.	$l \longleftarrow \lfloor \frac{i+j}{2} \rfloor$
7.	$x \longleftarrow$ NIL
8.	RETURN(x)

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.6.3. Láncolt lista

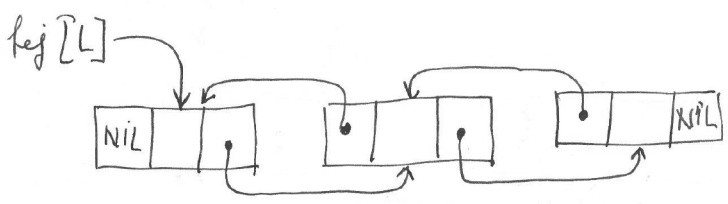
Kétirányban láncolt lista:

elő	kulcs	köv

Az egyirányban láncolt lista hasonló, de hiányzik belőle az előre mutató mutató, tehát csk egy irányban tudunk lépkedni az elemeken.

Először foglalkozunk a láncolt listával, de nem írjuk le részletesen a műveleteket (beszúrás, keresés, törlés), ezeket a műveleteket csak a szentineles láncolt lista esetén ismertetjük.

Jelölje $L =$ a láncolt listát. Egyetlen **attribútuma** van a fej$[L] =$, ami a legelső listaelemre mutató mutató.



Tehát:

fej$[L]$: az első listaelemre mutat (vagy NIL, ha a lista üres).

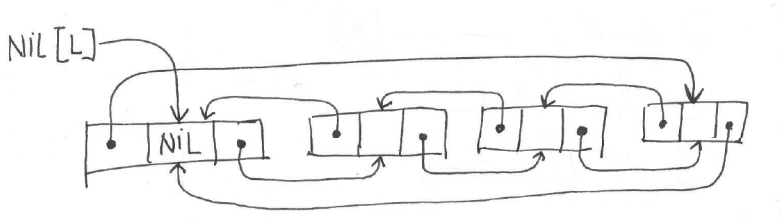
elő: az előző listaelemre mutat, illetve a lista első eleme esetén NIL

köv: a következő listaelemre mutat, illetve az utolsó listaelem esetén NIL

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

A szentineles láncolt lista. **Szentinel:** strázsa vagy őrszem, amely zárt láncolt listává teszi

a listát.



A szentineles láncolt listának egyetlen attribútuma van a NIL(L), ami a szentinelre mutató mutató. A szentinelnek NIL a kulcsa. A szentinel is rendelkezik elő, illetve köv mutatókkal, amelyek jelentése:

- elő [NIL[L]] az utolsó elemre mutat, illetve NIL, ha a lista üres;
- köv [NIL[L]] az első elemre mutat, illetve NIL, ha a lista üres.

A következő műveletekkel ismerkedünk meg kétszeresen láncolt szentineles listában:

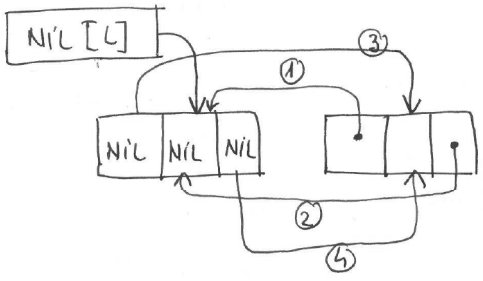
1. Keresés;
2. Beszúrás első elemként. (Külön vízsgáljuk azokat az eseteket, amikor a lista üres, illetve nem üres);
3. Beszúrás utolsó elemként (**Szorgalmi házi feladat**);
4. Beszúrás adott mutatójú elem elé;
5. Beszúrás adott mutatójú elem után (Szorgalmi házi feladat);
6. Adott mutatójú elem törlése.

1. Keresés:KERES(L, k, x)
INPUT: L a lista, k a keresett kulcs.
OUTPUT: x, ami a kulcs mutatója, ha a kulcs a listában van, vagy NIL, ha nincs a listában.

	KERES(L, k, x)
1.	$x \longleftarrow$ köv [NIL[L]]
2.	WHILE ($x \neq$ NIL[L] és kulcs[x] $\neq k$) DO
3.	$x \longleftarrow$ köv[x]
4.	IF $x =$ NIL[L] THEN $x \longleftarrow$ NIL
5.	RETURN(x)

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

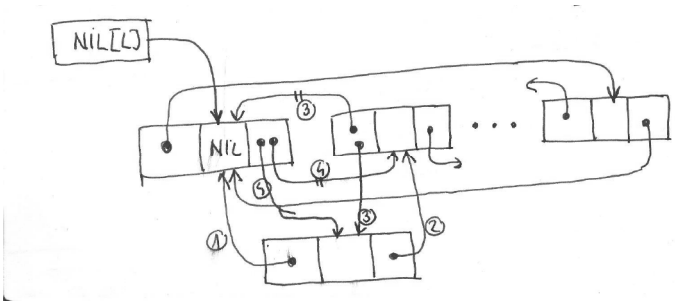
2. Beszúrás első elemként üres listába Az x mutatójú elemet szúrjuk be.



1. elő$[x] \longleftarrow$ NIL$[L]$
2. köv$[x] \longleftarrow$ NIL$[L]$
3. elő [NIL$[L]$] $\longleftarrow x$
4. köv [NIL$[L]$] $\longleftarrow x$

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

Beszúrás első elemként nemüres listába Az x mutatójú elemet szúrjuk be.



1. elő$[x] \longleftarrow$ NIL$[L]$
2. köv$[x] \longleftarrow$ köv [NIL$[L]$]
3. elő [köv [NIL$[L]$]] $\longleftarrow x$
4. köv [NIL$[L]$] $\longleftarrow x$

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

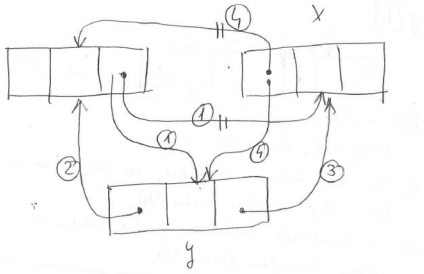
Mindkét fenti esetet magába foglaló pszeudokód: **Beszúrás első elemként**: BESZÚR(L, x)

INPUT: L a lista, x a beszúrandó elem mutatója.

	BESZÚR(L, x)
1.	IF (köv [NIL[L]] = NIL
2.	THEN elő[x] $\longleftarrow$ NIL[L]
3.	köv[x] $\longleftarrow$ NIL[L]
4.	elő [NIL[L]] $\longleftarrow$ x
5.	köv [NIL[L]] $\longleftarrow$ x
6.	ELSE elő[x] $\longleftarrow$ NIL[L]
7.	köv[x] $\longleftarrow$ köv [NIL[L]]
8.	elő [köv [NIL[L]]] $\longleftarrow$ x
9.	köv [NIL[L]] $\longleftarrow$ x
10.	RETURN

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

3. Beszúrás adott mutatójú elem elé: x mutatójú elem elé szúrjuk be az y mutatójú elemet.



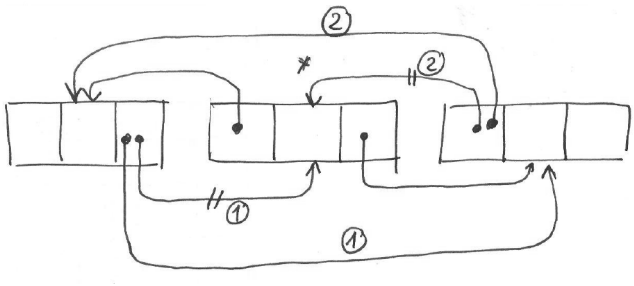
1. köv [elő[x]] $\longleftarrow$ y
2. elő[y] $\longleftarrow$ elő[x]
3. köv[y] $\longleftarrow$ x
4. elő[x] $\longleftarrow$ y

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

4. Beszúrás adott mutatójú elem után elem után: Szorgalmi házi feladat. Könnyen visszavezethető az előző feladatra.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

5. Adott mutatójú elem törlése: TÖRÖL(L, x)



INPUT: L a lista, x a törlendő elemre mutató mutató.

	TÖRÖL(L, x)
1.	köv [elő$[x]$] $\longleftarrow$ köv$[x]$
2.	elő [köv$[x]$] $\longleftarrow$ elő$[x]$
3.	RETURN

Példa: Végezzük el a következő műveleteket szentineles listában. (a beszúrás mindig első elemként történő beszúrást jelent)

$236, 546, 223, T546$

NIL$[L] = 1$ szabad=$[2, 3, 4, 5]$

1	NIL	NIL	NIL
2			
3			
4			
5			

236

NIL$[L] = 1$ szabad=$[3, 4, 5]$

1	2	NIL	2
2	1	NIL	1
3			
4			
5			

256

NIL$[L] = 1$ szabad=$[4, 5]$

1	2	NIL	3
2	3	236	1
3	1	256	2
4			
5			

223
NIL$[L] = 1$ szabad=[5]

1	2	NIL	4
2	3	236	1
3	4	546	2
4	1	223	3
5			

T546
NIL$[L] = 1$ szabad=$[5, 3]$

1	2	NIL	4
2	4	236	1
3	4	546	2
4	1	223	2
5			

0.7. Hasítótáblák, feloszt algoritmus

0.7.1. Hasítótábla

A hasítótábla azt a szituációt kezeli, amikor nagy a kulcstartomány, de csak kevés kulcsot kell kezelnünk. Erre példa egy kis válalkozás, amely 20 embert alkalmaz. Minden alkalmazottjának a kulcsa a személyi igazolvány száma. Ha minden lehetséges személyi igazolványszámhoz tárterületet rendelnénk, akkor túl nagy tárterületet vennénk igénybe. Számunkra elegendő, ha csak akkora tárterületet foglalunk, amelyben 25-30 dolgozó adatait lehet tárolni. Természetesen egy dolgozóhoz sok mező van rendelve, pl név, lakcím, beosztás, stb. ezek közül csak egy mező a kulcsmező.

Ekkor szükségünk van egy hasítófüggvényre. A hasítófüggvény kulcshoz memóriacímet rendel. A példánkban a kulcs a személyi igazolvány szám.

Mivel nagy a kulcstartomány (azaz sok lehetséges kulcs van), de csak kevés a tárterület, így szükségszerűen bekövetkezik az a probléma, hogy a hasítófüggvény több kulcshoz ugyanazt a memóriacímet rendeli. Ezt a jelenséget nevezzük ütközésnek. Tehát:

9. Definíció. ***Hasítófüggvény*** *kulcshoz memóriacímet rendel.*
Ütközés*: a hasítófüggvény több kulcshoz ugyanazt a címet rendeli.*

Az ütközésfeloldás típusai:

- **Közvetlen címzés**: ütközés feloldása: láncolt lista alkalmazásával történik. Tehát a táblázat rései memóriacímeket tartalmaznak. Ezek láncolt listák fejei. A dolgozó adataihoz a megfelelő láncolt listán keresztül lehet eljutni. Kulcsot törölni a láncolt listáknál tanult módon lehet.

- **Nyíltcímzéses hasítás** Tömbös implementációt használunk. Háromféle módszert tanulunk, ezek a következők:

  1. **lineáris kipróbálás**
  2. **négyzetes kipróbálás**
  3. **dupla hasítás**

Műveletek: beszúrás, keresés, törlés.
Állapotok: Sz (szabad), F (foglalt), T (törölt).

Bármelyik módszert alkalmazzuk a fenti három közül (lineáris kipróbálás, négyzetes kipróbálás, dupla hasítás) van két függvényünk: a $h_0(k)$ amely a k kulcshoz memóriacímet rendel, illetve a $h(k, i)$, amelynek $i = 0$ esetén az értéke azonos a $h_0(k)$ értékével, míg $i = 1, 2, \ldots, m-1$ esetén a $h(k, i)$ függvény segítségével végig lehet vele ugrálni a k kulcs által meghatározott sorrendben a táblázat összes résén.

Beszúrás: Kezdetben minden rés állapota Sz. Ekkor a $h_0(k)$ és a $h(k, 0)$ ad egy címet, ahová a k kulcsot beszúrjuk és a rés állapotát Sz-ről F-re állítjuk. Általában a beszúrás úgy történik, hogy a $h(k, i)$ függvény által előírt sorrendben addig ugrálunk résről résre, amíg Sz, vagy T állapotú résre találunk, ide beszúrjuk a kulcsot, majd a rés állapotát F-re állítjuk.

Keresés: A k kulcsú rekordot keressük. A $h_0(k)$ és a $h(k,i)$ függvények által meghatározott sorrendben F, vagy T állapotú réseken ugrálunk. Az F állapotú rések esetén ellenőrizni kell, hogy megtaláltuk-e az k kulcsú rekordot. T állapot esetén tovább kell folytatni a keresést. A keresés kétféle eredménnyel zárulhat:
Valamely F állapotú rés esetén megtaláljuk a k kulcsú rekordot.
Sz állapotú rést kapunk, ami azt jelenti, hogy a táblázatban nem szerepel az adott kulcsú rekord. A keresés művelete rávilágít arra, hogy miért van szükségünk az T állapotra. Ha a törlés alkalmával a rés állapotát F-ről Sz-re állítanánk, akkor ez az Sz állapotú rés olyan kulcs esetén is leállítaná a keresést "nincs a táblázatban" eredménnyel, amely szerepel a táblázatban.

Törlés: Megkeressük az adott kulcsú rekordot és az állapotát F-ről T-re állítjuk. Ez tulajdonképpen egy logikai törlés, az adatok fizikailag még jelen vannak, de az adott kulcsú rekord már nem elérhető. Fizikai törlésre akkor kerül sor, amikor rekordot szúrunk be erre a résre, ekkor a rés fizikailag felülíródik az új rekorddal, és a rés állapota T-ről F-re állítódik.

Most következzenek a numerikus példák. A táblázat a végeredményt tartalmazza, de leírtuk a számolást is. m a táblázat sorainak a számát jelöli, ezek 0-tól $m-1$-g vannak indexelve. A k a kulcsot (nemnegatív egész), a kulcs mögötti T betű az adott kulcsú elem törlését jelöli. A következő 3 numerikus feladatban az m értéke mindig 7 lesz.

1. **Lineáris kipróbálás**

$$\begin{array}{lcll} h_0(k) & = & k \mod m, & \\ h(k,i) & = & (h_0(k)+i) \mod m & (i=0,1,\ldots,m-1). \end{array}$$

$m = 7$ (a táblázat sorainak a száma).

Feladat. $k = 17, 127, 20, 45, 17T, 81, 127T, 3, 10$ *(T=törlés). Végezzük el a kijelölt műveleteket lineáris kipróbálás esetén.*

Az üres táblázat:

0	*Sz*	
1	*Sz*	
2	*Sz*	
3	*Sz*	
4	*Sz*	
5	*Sz*	
6	*Sz*	

Beszúrjuk a $k = 17$*-et.*
$h_0(17) = 17 \mod 7 = 3$
$h(17,0) = (3+0) \mod 7 = 3$

0	*Sz*	
1	*Sz*	
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	*Sz*	

Beszúrjuk a $k = 127$-et.
$h_0(127) = 127 \mod 7 = 1$
$h(127, 0) = (1 + 0) \mod 7 = 1$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	*Sz*	

Beszúrjuk a $k = 20$-at.
$h_0(20) = 20 \mod 7 = 6$
$h(20, 0) = (6 + 0) \mod 7 = 6$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 45$-öt.
$h_0(45) = 45 \mod 7 = 3$
$h(45, 0) = (3 + 0) \mod 7 = 3$
$h(45, 1) = (3 + 1) \mod 7 = 4$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	~~*Sz*~~*F*	*45*
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Töröljük a $k = 17$-et.
17T $h_0(17) = 17 \mod 7 = 3$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*SzF*~~*T*	*17*
4	~~*Sz*~~*F*	*45*
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 81$-et.
$h_0(81) = 81 \mod 7 = 4$
$h(81, 0) = (4 + 0) \mod 7 = 4$
$h(81, 1) = (4 + 1) \mod 7 = 5$

0	*Sz*	
1	~~*SzF*~~	*127*
2	*Sz*	
3	~~*SzF*~~*T*	*17*
4	~~*Sz*~~*F*	*45*
5	~~*Sz*~~*F*	*81*
6	~~*Sz*~~*F*	*20*

Töröljük a $k = 127$-et.
127T $h_0(127) = 127 \mod 7 = 1$

0	*Sz*	
1	~~*SzF*~~*T*	*127*
2	*Sz*	
3	~~*SzF*~~*T*	*17*
4	~~*Sz*~~*F*	*45*
5	~~*Sz*~~*F*	*81*
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 3$-at.
$h_0(3) = 3 \mod 7 = 3$
$h(3, 0) = (3 + 0) \mod 7 = 3$

0	*Sz*	
1	~~*SzF*~~*T*	*127*
2	*Sz*	
3	~~*SzFTF*~~	~~*173*~~
4	~~*Sz*~~*F*	*45*
5	~~*Sz*~~*F*	*81*
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 10$-et.
$h_0(10) = 10 \mod 7 = 3$
$h(10, 0) = (3 + 0) \mod 7 = 3$
$h(10, 1) = (3 + 1) \mod 7 = 4$
$h(10, 2) = (3 + 2) \mod 7 = 5$
$h(10, 3) = (3 + 3) \mod 7 = 6$
$h(10, 4) = (3 + 4) \mod 7 = 0$

0	~~*Sz*~~ *F*	10
1	~~*Sz F T*~~	127
2	*Sz*	
3	~~*Sz F T*~~ *F*	~~*17*~~ 3
4	~~*Sz*~~ *F*	45
5	~~*Sz*~~ *F*	81
6	~~*Sz*~~ *F*	20

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

2. **Négyzetes kipróbálás**

$$\begin{aligned} h_0(k) &= k \mod m, \\ h(k,i) &= \left(h_0(k) + \frac{i(i+1)}{2}\right) \mod m \qquad (i = 0, 1, \ldots, m-1). \end{aligned}$$

Állapotok: Sz (szabad), F (foglalt), T (törölt).
$m = 7$ (a táblázat sorainak a száma).

Feladat. $k = 17, 127, 20, 45, 17T, 81, 127T, 3, 10$. *Végezzük el a kijelölt műveleteket négyzetes kipróbálás esetén.*

0	*Sz*	
1	*Sz*	
2	*Sz*	
3	*Sz*	
4	*Sz*	
5	*Sz*	
6	*Sz*	

Beszúrjuk a $k = 17$*-et.*
$h_0(17) = 17 \mod 7 = 3$
$h(17, 0) = \left(3 + \frac{0 \cdot 1}{2}\right) \mod 7 = 3$

0	*Sz*	
1	*Sz*	
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	*Sz*	

Beszúrjuk a $k = 127$*-et.*
$h_0(127) = 127 \mod 7 = 1$

$h(127,0) = \left(1 + \frac{0 \cdot 1}{2}\right) \mod 7 = 1$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	*Sz*	

Beszúrjuk a $k = 20$*-at.*
$h_0(20) = 20 \mod 7 = 6$
$h(20,0) = \left(6 + \frac{0 \cdot 1}{2}\right) \mod 7 = 6$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 45$*-öt.*
$h_0(45) = 45 \mod 7 = 3$
$h(45,0) = \left(3 + \frac{0 \cdot 1}{2}\right) \mod 7 = 3$
$h(45,1) = \left(3 + \frac{1 \cdot 2}{2}\right) \mod 7 = 4$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	~~*Sz*~~*F*	*45*
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Töröljük a $k = 17$*-et.*
17T $h_0(17) = 17 \mod 7 = 3$
$h(17,0) = \left(3 + \frac{0 \cdot 1}{2}\right) \mod 7 = 3$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*SzF*~~*T*	*17*
4	~~*Sz*~~*F*	*45*
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 81$-et.
$h_0(81) = 81 \mod 7 = 4$
$h(81, 0) = \left(4 + \frac{0 \cdot 1}{2}\right) \mod 7 = 4$
$h(81, 1) = \left(4 + \frac{1 \cdot 2}{2}\right) \mod 7 = 5$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*SzF*~~*T*	*17*
4	~~*Sz*~~*F*	*45*
5	~~*Sz*~~*F*	*81*
6	~~*Sz*~~*F*	*20*

Töröljük a $k = 127$-et.
$\boxed{127T}$ $h_0(127) = 127 \mod 7 = 1$
$h(127, 0) = \left(1 + \frac{0 \cdot 1}{2}\right) \mod 7 = 1$

0	*Sz*	
1	~~*SzF*~~*T*	*127*
2	*Sz*	
3	~~*SzF*~~*T*	*17*
4	~~*Sz*~~*F*	*45*
5	~~*Sz*~~*F*	*81*
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 3$-at.
$h_0(3) = 3 \mod 7 = 3$
$h(3, 0) = \left(3 + \frac{0 \cdot 1}{2}\right) \mod 7 = 3$

0	*Sz*	
1	~~*SzF*~~*T*	*127*
2	*Sz*	
3	~~*SzFTF*~~	~~*173*~~
4	~~*Sz*~~*F*	*45*
5	~~*Sz*~~*F*	*81*
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 10$-et.
$h_0(10) = 10 \mod 7 = 3$
$h(10, 0) = \left(3 + \frac{0 \cdot 1}{2}\right) \mod 7 = 3$
$h(10, 1) = \left(3 + \frac{1 \cdot 2}{2}\right) \mod 7 = 4$
$h(10, 2) = \left(3 + \frac{2 \cdot 3}{2}\right) \mod 7 = 6$

$h(10,3) = \left(3 + \frac{3\cdot 4}{2}\right) \mod 7 = 2$

0	*Sz*	
1	~~*Sz*~~ ~~*F*~~ *T*	127
2	~~*Sz*~~ *F*	10
3	~~*Sz*~~ ~~*F*~~ ~~*T*~~ *F*	~~*17*~~ 3
4	~~*Sz*~~ *F*	45
5	~~*Sz*~~ *F*	81
6	~~*Sz*~~ *F*	20

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

3. **Dupla hasítás**

$$\begin{array}{lcll} h_0(k) & = & k \mod m, & \\ h_1(k) & = & 1 + (k \mod (m-1)) & \\ h(k,i) & = & (h_0(k) + ih_1(k)) \mod m & (i = 0, 1, \ldots, m-1). \end{array}$$

$m = 7$ (a táblázat sorainak a száma).

Feladat. $k = 17, 127, 20, 45, 17T, 81, 127T, 3, 10$. *Végezzük el a kijelölt műveleteket dupla hasítás segítségével.*

0	*Sz*	
1	*Sz*	
2	*Sz*	
3	*Sz*	
4	*Sz*	
5	*Sz*	
6	*Sz*	

Beszúrjuk a $k = 17$*-et.*

$$\left.\begin{array}{l} h_0(17) = 17 \mod 7 = 3 \\ h_1(17) = 1 + (17 \mod 6) = 6 \end{array}\right\} \Rightarrow \quad h(17,0) = (3 + 0 \cdot 6) \mod 7 = 3$$

0	*Sz*	
1	*Sz*	
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	*Sz*	

Beszúrjuk a $k = 127$-et.

$$\left.\begin{array}{l} h_0(127) = 127 \mod 7 = 1 \\ h_1(127) = 1 + (127 \mod 6) = 2 \end{array}\right\} \quad \Rightarrow \quad h(127, 0) = (1 + 0 \cdot 2) \mod 7 = 1$$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	*Sz*	

Beszúrjuk a $k = 20$-at.

$$\left.\begin{array}{l} h_0(20) = 20 \mod 7 = 6 \\ h_1(20) = 1 + (20 \mod 6) = 3 \end{array}\right\} \quad \Rightarrow \quad h(20, 0) = (6 + 0 \cdot 3) \mod 7 = 6$$

0	*Sz*	
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 45$-öt.

$$\left.\begin{array}{l} h_0(45) = 45 \mod 7 = 3 \\ h_1(45) = 1 + (45 \mod 6) = 4 \end{array}\right\} \quad \Rightarrow \quad \begin{array}{l} h(45, 0) = (3 + 0 \cdot 4) \mod 7 = 3 \\ h(45, 1) = (3 + 1 \cdot 4) \mod 7 = 0 \end{array}$$

0	~~*Sz*~~*F*	*45*
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*Sz*~~*F*	*17*
4	*Sz*	
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Töröljük a $k = 17$-et.

$$17T \quad \left.\begin{array}{l} h_0(17) = 17 \mod 7 = 3 \\ h_1(17) = 1 + (17 \mod 6) = 6 \end{array}\right\} \quad \Rightarrow \quad h(17, 0) = (3 + 0 \cdot 6) \mod 7 = 3$$

0	~~*Sz*~~*F*	*45*
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*SzF*~~*T*	*17*
4	*Sz*	
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 81$-et.

$$\left.\begin{array}{l} h_0(81) = 81 \mod 7 = 4 \\ h_1(81) = 1 + (81 \mod 6) = 4 \end{array}\right\} \quad \Rightarrow \quad h(81, 0) = (4 + 0 \cdot 4) \mod 7 = 4$$

0	~~*Sz*~~*F*	*45*
1	~~*Sz*~~*F*	*127*
2	*Sz*	
3	~~*SzF*~~*T*	*17*
4	~~*Sz*~~*F*	*81*
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Töröljük a $k = 127$-et.

$$127T \quad \left.\begin{array}{l} h_0(127) = 127 \mod 7 = 1 \\ h_1(127) = 1 + (127 \mod 6) = 2 \end{array}\right\} \quad \Rightarrow \quad h(127, 0) = (1 + 0 \cdot 2) \mod 7 = 1$$

0	~~*Sz*~~*F*	*45*
1	~~*SzF*~~*T*	*127*
2	*Sz*	
3	~~*SzF*~~*T*	*17*
4	~~*Sz*~~*F*	*81*
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 3$-at.

$$\left.\begin{array}{l} h_0(3) = 3 \mod 7 = 3 \\ h_1(3) = 1 + (3 \mod 6) = 4 \end{array}\right\} \quad \Rightarrow \quad h(3, 0) = (3 + 0 \cdot 4) \mod 7 = 3$$

0	~~*Sz*~~*F*	*45*
1	~~*SzF*~~*T*	*127*
2	*Sz*	
3	~~*SzFTF*~~	~~*173*~~
4	~~*Sz*~~*F*	*81*
5	*Sz*	
6	~~*Sz*~~*F*	*20*

Beszúrjuk a $k = 10$-et.

$$\left.\begin{array}{l} h_0(10) = 10 \mod 7 = 3 \\ h_1(10) = 1 + (10 \mod 6) = 5 \end{array}\right\} \Rightarrow \begin{array}{l} h(10,0) = (3 + 0 \cdot 5) \mod 7 = 3 \\ h(10,1) = (3 + 1 \cdot 5) \mod 7 = 1 \end{array}$$

0	~~*Sz*~~ *F*	45
1	~~*Sz F T*~~ *F*	~~*127*~~ *10*
2	*Sz*	
3	~~*Sz F T*~~ *F*	~~*17*~~ 3
4	~~*Sz*~~ *F*	81
5	*Sz*	
6	~~*Sz*~~ *F*	20

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.7.2. A FELOSZT algoritmus

A FELOSZT algoritmus *Egy A tömb $A[p \dots r]$ résztömbjét egy adott $x \in \mathbb{R}$ szám körül felosztja az $A[p \dots q]$, $A[q \dots r]$ két résztömbre olyan módon, hogy az $A[p \dots q]$ résztömbbe az A tömbnek azok az elemei kerülnek, amelyek kisebbek vagy egyenlőek x-nél, az $A[q+1 \dots r]$ résztömbbe azok az elemek kerülnek, amelyek nagyobbak vagy egyenlőek x-nél.*

Feloszt algoritmus: FELOSZT(A, p, r, x, q)
INPUT: A a tömb, amelynek az $A[p \dots r]$ résztömbjét felosztjuk, $x \in \mathbb{R}$ az előre megadott érték, amely a felosztást szabályozza.
OUTPUT: A a megváltozott tömb, q a felosztás határa, $A[p \dots q]$, $A[q+1 \dots r]$ a kapott résztömbök.

	FELOSZT(A, p, r, x, q)
1.	$i \longleftarrow p-1$, $j \longleftarrow r+1$
2.	WHILE IGAZ DO
3.	REPEAT $j \longleftarrow j-1$
4.	UNTIL $A[j] \leqslant x$
5.	REPEAT $i \longleftarrow i+1$
6.	UNTIL $x \leqslant A[i]$
7.	IF $i < j$
8.	THEN CSERE$(A[i], A[j])$
9.	ELSE $q \longleftarrow j$, RETURN(A, q)

A feloszt algoritmus működése:

Gondolatban két nyilat mozgatunk. A nyilak indexeket jelölnek, a nyilak a vizsgált indexű elemre mutatnak. A bal oldali nyilacska az i (index), a jobb oldali a j. A nyilakat kiinduló helyzetben tartományon kívülre, a tartomány szélére állítjuk. Mindkét nyilat befelé mozgatjuk.

A j indexet jelölő nyilacskát mindaddig mozgatjuk balra, (azaz csökkentjük), amíg az elem amire mutat nagyobb x-nél. Amint olyan elemre mutat, ami egyenlő x-el, vagy kisebb nála, megállunk.

Az i indexet jelölő nyilacskát mindaddig mozgatjuk jobbra, (azaz növeljük), amíg az elem amire mutat kisebb x-nél. Amint olyan elemre mutat, ami egyenlő x-szel, vagy nagyobb nála, megállunk.

Ekkor megvizsgáljuk a két indexet. Ha i kisebb j-nél, akkor megcseréljük azokat az elemeket, amelyekre az i illetve a j mutat és tovább mozgatjuk befelé a nyilacskákat. Ha azonban az i és a j átfedik egymást (i=j), vagy j megelőzi az i-t (j<i), akkor leáll az algoritmus, a q megkapja a j értékét és visszatérünk q-val és a megváltozott tömbbel. A q jelöli a felosztás határát.

Érdemes megtanulni a feloszt algoritmust, hiszen ez működteti a gyors rendezést, ami fontos, illetve a kiválasztás lineáris időben algoritmust, ami érdekes.

Példa. *Legyen* $A = [17, 127, 20, 45, 81, 3, 10, 49, 50, 28, 12, 81, 9]$. *Osszuk fel A-t az* $x = 20$ *körül.*

	17	127	20	45	81	3	10	49	50	28	12	81	9	
↑														↑
		↑											↑	
		9	20	45	81	3	10	49	50	28	12	81	127	
			↑								↑			
			12	45	81	3	10	49	50	28	20			
				↑			↑							
				10	81	3	45							
					↑	↑								
					3	81								
					↑ j	↑ i								

$q \longleftarrow j = 5$, *RETURN*$(A, 5)$, $A = [17, 9, 12, 10, 3 \mid 81, 45, 49, 50, 28, 20, 81, 127]$

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

Kiválasztás lineáris időben

10. Definíció. ***Kiválasztási probléma:*** *Legyen adott egy A halmaz (n db páronként különböző szám). Adott továbbá* $1 \leqslant i \leqslant n$ *pozitív egész. Keressük az A halmaznak azt az x elemét, amelynél pontosan* $i - 1$ *db kisebb eleme van a halmaznak.*

A kiválasztás lineáris időben algoritmushoz szükségünk lesz az (alsó) medián fogalmára.

11. Definíció. ***Medián*** *páratlan elemszámú A halmaz esetén a rendezett A halmaz középső eleme, míg páros elemszám esetén a két középső közül a kisebb vagy egyenlő (alsó medián).* $A = \{x_1, \ldots x_n\} \rightsquigarrow$ *rendezzük:* $x_1^* \leqslant x_2^* \leqslant \cdots \leqslant x_n^*$.

$$\text{med} = x^*_{\lceil \frac{n}{2} \rceil}$$

0.8. Rendezések I

Ezen a gyakorlaton elkezdjük a rendezési algoritmusok ismertetését. Rakjuk sorrendbe a rendezéseket. Logikailag először kellene tárgyalni a buborék rendezést, utána talán a minimumkereséseset, majd a beszúrásosat, utána következhetne a shell, mert az a végén átmegy beszúrásosba, majd jöhetne a négyzetes. Folytathatnánk a sort két jópofa rekurzívval, úgymint gyors és összefésüléses rendezésekkel. Ezeket követhetnék azok az algoritmusok, amelyek nem összehasonlításon alapulnak, hanem a rendezendő elemek valamilyen speciális tulajdonságát használják ki. Ilyenek a leszámláló és a radix. Végül a teljesség igénye nélkül befejezhetnénk a sorozatot a kupacrendezéssel.

Ezt a szép tervet felborítja az a tény, hogy a kiválasztást előbb tárgyaljuk, mint a rendezést és még adósak vagyunk a kiválasztás lineáris időben algoritmussal. Ez az algoritmus alkalmazza a feloszt algoritmust, melynek futtatásában már rutint szereztünk az előző gyakorlaton.

A kiválasztási probléma józan paraszti ésszel úgy oldható meg, hogy megkeressük a tömb legkisebb elemét, ezt egy cserével a tömb elejére rakjuk, majd megkeressük a második legkisebbet, ezt egy cserével a második helyre rakjuk, és ez így megy az i-edik elemig. Sajnos ez egy négyzetes futási idejű algoritmus, szemben a kiválasztás lineáris időben algoritmussal, ami a nevének megfelelően lineáris idejű.

A leírtaknak megfelelően először vesszük a minimumkeresést, aztán ha már itt vagyunk, a minimumkeresésre épülő négyzetes futási idejű **minimumkereséses rendezés**t.

Ezt követően bepótoljuk a hiányosságunkat, átnézzük a **kiválasztás lineáris időben** algoritmust, majd ezt követi a **gyorsrendezés**, mert ez az algoritmus is a feloszt algoritmusra épül. A gyorsrendezés valóban gyors, bár a futási ideje négyzetes, de az átlagos futási ideje $n \log(n)$, ami egy elméleti alsó korlát az összehasonlításon alapuló rendezésekre. Az algoritmus rekurzív. A rendezendő tömböt az első eleme körül felosztjuk, majd két résztömbre bontjuk a felosztás határa szerint. Ezt követően a résztömbökön futtatjuk az algoritmust mindaddig, amíg egy elemű résztömböket nem kapunk. Ezáltal a tömb rendezetté válik.

Ezt követően vegyük előre a leszámláló rendezést, mert a gyors, a leszámláló és a kupacrendezés talán a legkevésbé triviálisak, viszont a kupacrendezésre nem mindig marad idő.

A **Leszámláló rendezés** egy lineáris idejű rendezési algoritmus, ami nagyon kecsegtetően hangzik, azonban nagy ára van. Csak kis felső korláttal rendelkező pozitív (vagy akár nemnegatív) számok rendezésére alkalmas. Rendkívül ötletes, (mármint ha szabad ilyet mondanom). Hatalmas előnye, hogy stabil rendezés, azaz az azonos kulcsú elemek eredeti sorrendjét megőrzi. Ezért ezt a rendező algoritmust felhasználja más rendező algoritmus is.

A buborékrendezés szintén négyzetes idejű. Rendkívül egyszerű. Tömb elemeinek a rendezésére alkalmas. Vagy lentről fölfelé vagy fentről lefelé haladunk végig a tömb elemein. Ha az egymást követő elemek nem a megfelelő sorrendben vannak, akkor megcseréljük őket. Így ha lentről felfelé haladunk, akkor a legnagyobb elem a tömb "tetejére" míg ha fentről lefelé haladunk, akkor a legkisebb elem a tömb legaljára kerül. Ez a buborék, amely vagy lentről felfelé, vagy fentről lefelé halad. Ezt az eljárást kell ismételni mindaddig, amíg rendezetté nem válik a tömb. Van lehetőség az algoritmus finomítására. Az egyik egy logikai típusú változó bevezetése, ami azt figyeli, hogy volt-e csere. Ha nem volt csere, akkor már rendezett a tömb. A másik módszer az, hogy váltogatjuk az összehasonlítás irányát. Ez utóbbi algoritmus a

koktélrendezés, amivel azonban már nem foglalkozunk.

A **beszúró rendezés** egy négyzetes futási idejű rendezés. El sem tudok képzelni jobb rendezést, ha Zh-kat kell rendeznünk betűrend szerint.

A **Shell rendezés** egy $n^{1,5}$ futásidejű algoritmus, amely gyorsabb, mint a négyzetes, de lassabb, mint az $n\log(n)$ idejű. Teljes neve: Shell rendezés csökkenő növekményekkel. Az algoritmus először a rendezendő tömb távoli elemeit mozgatja, ez a távolság a növekmény. Amikor a növekmény eléri az egyet, akkor egy beszúró rendezés valósul meg, de eddigre a tömb már majdnem rendezetté válik.

A négyzetes rendezés szintén $n^{1,5}$ futási idejű.

Összefésülő rendezés. Ehhez az algoritmushoz szükségünk van egy összefésül algoritmusra. Az összefésül algoritmus rendezett tömbök rendezett összefésülésére szolgál. Tehát van két rendezett tömbünk, ezeket egy többé fésüljük össze úgy, hogy megint csak rendezett tömböt kapjunk. A Batcher-féle páros-páratlan összefűzés "csupán" egy gyors és ötletes módszer a rendezett összefésülés kivitelezésére. Az összefésülő rendezés egy rekurzív, $n\log(n)$ futási idejű algoritmus. Úgy működik, hogy a tömböt (valahol középen) kettévágjuk. A kettévágást mindaddig folytatjuk, amíg egy elemű résztömböket nem kapunk. Az egyelemű résztömbök rendezettek. Ezt követően a rendezett résztömböket rendezett módon összefűzzük. Ezáltal a tömb rendezetté válik.

A **számjegyes, vagy radix rendezés** szintén lineáris futási idejű. Ez a rendezés alkalmas ugyanannyi számjegyből álló számok rendezésére. A radix rendezés stabil rendezés, tehát az azonos kulcsú elemek eredeti sorrendjét megőrzi. A radix rendezéshez szükség van stabil rendezésre, például a leszámláló rendezésre. Az algoritmus roppant egyszerű. A stabil rendezéssel a rendezendő számokat a legkisebb helyiértékű (tehát utolsó jegyük) szerint rendezzük. Ezt követően a számjegyeket a második, harmadik, legkisebb, végül a legnagyobb helyiértékű jegyük alapján rendezzük. Ezáltal a számok sorozata rendezetté válik.

Az edényrendezés is lineáris idejű. Ez az algoritmus is akkor fut gyorsan, ha a rendezendő elemek (matematikailag valós számok) egyenletes eloszlásúak egy $[a, b[$ intervallumban. (Az egyenletes eloszlás azt jelenti, hogy nagy elemszám esetén az adatelemek részintervallumba esésének valószínűsége arányos a részintervallum hosszával.) Mivel egy $[a, b[$ intervallum mindig áttranszformálható a $[0, 1[$ intervallumba, így csak a $[0, 1[$ intervallum esetét tárgyaljuk. A $[0, 1[$ intervallumot osszuk fel n egyenlő hosszúságú részintervallumra. Ezek a kis intervallumok az edények. Létrehozunk egy n elemű tömböt, amely láncolt listák fejeit tartalmazza. Van egy ügyes függvényünk, amely egy tetszőleges $[0, 1[$-beli valós száról kapásból megmondja, hogy melyik $\frac{1}{n}$ hosszúságú részintervallumba tartozik, majd beszúrjuk a megfelelő láncolt listába. Ezután a láncolt listákat rendezzük, majd egymás után fűzzük.

A **kupac rendezés** egy $n\log(n)$ futási idejű rendezés, amely a kupac adatstruktúrán alapul. a kupac adatstruktúra egy fagráf, amely sok érdekes tulajdonsággal rendelkezik, például a gráf gyökere a legnagyobb elem. A gyökérelem lesz a tömb legutolsó eleme. Ezáltal elromlik a kupac struktúra. Helyreállítjuk a kupac struktúrát és iteráltan alkalmazzuk az eljárást.

Most ismételjük át a rendezési algoritmusokat és a futási idejüket.

Rendező algoritmusok:

1. Minimum kiválasztásos rendezés: $T(n) = \Theta(n^2)$.

2. Gyorsrendezés: $T(n) = \Theta(n^2)$, de átlagosan $T(n) = \Theta(n \log(n))$.
3. Leszámláló rendezés: $T(n) = \Theta(n)$.
4. Buborék rendezés: $T(n) = \Theta(n^2)$.
5. Beszúró rendezés: $T(n) = \Theta(n^2)$.
6. Shell rendezés: $T(n) = \Theta(n^{1,5})$.
7. Összefésülő rendezés: $T(n) = \Theta(n \log(n))$.
8. Négyzetes rendezés: $T(n) = \Theta(n^{1,5})$.
9. Számjegyes (radix) rendezés: $T(n) = \Theta(n)$.
10. Edény rendzés: átlagosan $T(n) = \Theta(n)$.
11. Kupac rendezés: átlagosan $T(n) = \Theta(n \log(n))$.

0.8.1. A rendezés absztrakt fogalma

Most elevenítsük fel a rendezés és a rendezett halmaz fogalmát.

Definíció. *Legyen X egy halmaz és $\leqslant$ egy reláció X-en. Az $\leqslant$ relációt **rendezés**nek nevezzük, ha rendelkezik a következő tulajdonságokkal:*

1. ***reflexív**, azaz $x \leqslant x$ minden $x \in X$ esetén.*
2. ***antiszimmetrikus**, azaz ha $x \leqslant y$ és $y \leqslant x$ akkor $x = y$ minden $x, y \in X$ esetén.*
3. ***tranzitív**, azaz $x \leqslant y$ és $y \leqslant z$, akkor $x \leqslant z$ minden $x, y, z \in X$ esetén.*
4. ***dichotom**, azaz $x \leqslant y$ vagy $y \leqslant x$ minden $x, y \in X$ esetén.*

Azt mondjuk, hogy az $(X, \leqslant)$ egy rendezett halmaz, ha $\leqslant$ egy rendezés az X-en.

Példa. *A valós számok $\mathbb{R}$ halmazán a szokásos $\leqslant$ reláció egy rendezés. Ez a rendezés úgy magyarázható meg, hogy $x \leqslant y$ pontosan akkor teljesül, ha a azámegyenesen az x pont azonos az y ponttal, vagy az y pont az x pont jobb oldalán található. Persze ezt a fogalmat az Analízis I tárgyban egy jobban részleteztük.*

Ha X egy halmaz, $\leqslant$ egy olyan reláció X-en, amelyre a fenti definíció tulajdonságai közül csak az első három tulajdonság teljesül, akkor a $\leqslant$ relációt **parciális rendezés**nek, illetve az X halmazt parciálisan rendezett halmaznak nevezzük.

Ha ki akarjuk hangsúlyozni, hogy a $\leqslant$ reláció a fenti definícióban felsorolt tulajdonságok közül mind a négy tulajdonsággal rendelkezik, akkor azt mondjuk, hogy a $\leqslant$ reláció linearis

rendezés X-en. A (4) tulajdonság, azaz a rendezés dichotom tulajdonsága azt jelenti, hogy az X halmaz bármely két eleme összehasonlítható.

A továbbiakban az esetek többségében valamilyen számhalmaz elemeit rendezzük és a szokásos lineáris rendezést használjuk. Azonban az olyan parciális rendezés, amely nem lineáris, sem egy valóságtól elrugaszkodott fogalom. Erre vonatkozik a következő egyszerű példa.

Példa. *Jelölje X az $\{1,2,3\}$ halmaz összes részhalmazának a halmazát. Ekkor*

$$X = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}.$$

Az X halmazon a rendezés legyen a szokásos halmazelméleti tartalmazás, jele: $\subseteq$. A tartalmazás reflexív, szimmetrikus, tranzitív, de nem dichotom, mert például a $\{1,2\}$ és a $\{2,3\}$ a tartalmazás szempontjából nem összehasonlíthatók.

A továbbiakban számokat tartalmazó tömbök elemeit rendezzük. Ez alól csak a számjegyes, vagy más néven radix rendezés lesz a kivétel, mert akkor egy ábécé betűiből alkotott n hosszúságú szavakat is fogjuk tudni rendezni. A lexikografikus rendezés az, ami a neve. Ha az ábécé betűi rendezettek, akkor a betűk rendezése segítségével rendezés definiálható az az n hosszúságú szavak halmazán. A lexikografikus rendezés definíciójában van egy finom trükk, de ezt majd látni fogjuk a radix rendezésnél.

Most minden tanult rendezésnek megadjuk a pszeudo-kódját és az algoritmus működését numerikus példák segítségével illusztráljuk.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.8.2. Minimumkeresés tömbben

Minimum keresés tömbben: *Egy A tömb $A[p\ldots,r]$ résztömbében keressük a minimumot. Az algoritmus a minimum indexével tér vissza.*
MINKER$(A,p,r,minindex)$
INPUT: A, p, r.
OUTPUT: minindex.

	MINKER$(A,p,r,minindex)$
1.	*minindex* $\longleftarrow p$
2.	*FOR* $j \longleftarrow p+1$ *TO* r *DO*
3.	*IF* $[A[j] < A[minindex]$ *THEN minindex* $\longleftarrow j$
4.	*RETURN(minindex)*

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

A minimumkeresés algoritmusa elvileg alkalmas a kiválasztási probléma megoldására, de ennél sokkal jobb lesz a kiválasztás lineáris időben algoritmus. A minimumkeresés algoritmus működteti a minimumkiválasztásos rendezést, ami egy négyzetes futási idejű rendezés.

0.8.3. Minimumkiválasztásos rendezés

1. Minimumkiválasztásos rendezés:
MINREND(A)
INPUT: $A[1,\dots,n]$ *a rendezendő tömb.*
OUTPUT: $A[1,\dots,n]$ *a rendezett tömb.*

	MINREND(A)
1.	*FOR* $i \longleftarrow 1$ *TO* $n-1$ *DO*
2.	*MINKER*$(A, i, n, minindex)$
3.	*IF* $i <$ *minindex THEN CSERE* $(A[i], A[minindex])$
4.	*RETURN(A)*

Feladat. *Minimumkiválasztásos rendezés segítségével rendezzük az*

$$A = [13, 22, 8, 315, 3]$$

tömböt.

$i = 1$ *MINKER*$(A, 1, 5)$ $minindex = 4$ *CSERE* $(A[1], A[4])$ $A = [3||22, 8, 315, 13]$
$i = 2$ *MINKER*$(A, 2, 5)$ $minindex = 3$ *CSERE* $(A[2], A[3])$ $A = [3, 8||22, 315, 13]$
$i = 3$ *MINKER*$(A, 3, 5)$ $minindex = 5$ *CSERE* $(A[3], A[5])$ $A = [3, 8, 13||315, 22]$
$i = 4$ *MINKER*$(A, 4, 5)$ $minindex = 5$ *CSERE* $(A[4], A[5])$ $A = [3, 8, 13, 22, 315]$

0.8.4. Kiválasztás lineáris időben algoritmus

Kiválasztási probléma *Adott egy A halmaz, amelynek n darab páronként különböző eleme van, továbbá adott egy* $i \in \mathbb{Z}_+$ *szám (nem index, hanem darabszám), amelyre* $1 \leqslant i \leqslant n$. *Keressük az A halmaznak azt az x elemét, amelyre teljesül, hogy pontosan* $(i-1)$ *darab nála kisebb kisebb eleme van a halmaznak.*

Az algoritmus leírásához szükségünk lesz a medián fogalmára.

A medián *páratlan elemszám esetén a rendezett adathalmaz középső eleme. Páros elemszám esetén a két középső közül a kisebb vagy egyenlő. Azaz alsó mediánt keresünk.*

Kissé részletesebben: Legyen a minta $x_1, \dots, x_n$ *a rendezett minta:* $x_1^* \leqslant \dots \leqslant x_n^*$. *Ekkor*

$$\text{med} = x^*_{\lceil \frac{n}{2} \rceil}.$$

Kiválasztás lineáris időben
KIVÁLASZT(A, p, r, i, x)
INPUT: A a tömb, amelynek az $A[p \dots r]$ résztömbjének keressük az i-edik legkisebb elemét. $i \in \mathbb{Z}_+$, $1 \leqslant i \leqslant r - p + 1$..

OUTPUT: x a keresett elem.

	KIVÁLASZT(A, p, r, i, x)
1.	IF $p = r$ THEN $x \longleftarrow A[p]$
2.	ELSE Az $A[p..r]$ tömb elemeit 5 elemből álló csoportokba soroljuk. A kapott csoportok mediánjait elhelyezzük egy M tömb $M[1..k]$ résztömbjébe, ahol $k = \left\lceil \frac{r-p+1}{5} \right\rceil$
3.	medmed $\longleftarrow$ med $(M[1], \ldots, M[k])$. Ennek a mediánnak a számára $k \leqslant 5$ esetén közvetlenül történik, ellenkező esetben a KIVÁLASZT $\left(M, 1, k, \lceil \frac{k}{2} \rceil, \text{medmed}\right)$ módon, tehát rekurzívan.
4.	FELOSZT$(A, p, r, \text{medmed}, q)$. IF $i \leqslant q - p + 1$ THEN KIVÁLASZT(A, p, q, i, x) ELSE KIVÁLASZT$(A, q + 1, r, i - (q - p + 1), x)$
5.	RETURN(x)

Magyarázat *Az $A[p..r]$ tömb i-edik eleme a $p + i - 1$ elem. Azt kell megvizsgálnunk, hogy ez az elem a q indexű elemhez képest hol van a tömbben. Nézzük meg a 4. pontot.*
4. IF $p + i - 1 \leqslant q$ THEN KIVÁLASZT(A, p, q, i, x)
ELSE KIVÁLASZT$(A, q + 1, r, i - (q - p + 1), x)$.
Azonban a $p + i - 1 \leqslant q$ feltétel ekvivalens az $i \leqslant q - p + 1$ feltétellel.

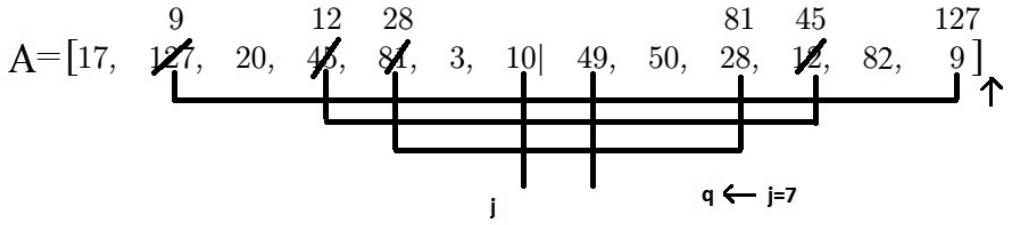
Feladat *Válasszuk ki az*

$$A = [17, 127, 20, 45, 81, 3, 10, 49, 50, 28, 12, 82, 9]$$

tömb 9. elemét.
1. *KIVÁLASZT$(A, p = 1, r = 13, i = 9, x)$*
Felosztjuk a tömböket 5-ös csoportokra, med, medmed.

	$17, 127, 20, 45, 81 \mid$	$3, 10, 49, 50, 28 \mid$	$12, 82, 9$
med	45	28	12
medmed		28	

FELOSZT$(A, 1, 13, 28, q)$ $x = 28$



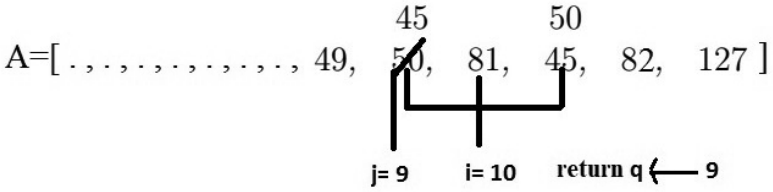
$\boxed{q=7}$

$i = 9 \overset{?}{\leqslant} q - p + 1 = 7 - 1 + 1 = 7$ *(nem)* $\Longrightarrow$

2.*KIVÁLASZT*(A, 8, 13, $9\text{-}(7\text{-}1\text{+}1)\text{=}2$, x)
p r i

		49,	50,	81,	45,	82 | 127
med				50		127
medmed					50	

FELOSZT$(A, p = 8, r = 13, x = 50, q)$



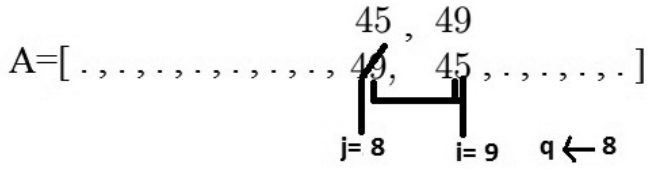
$\boxed{q\text{=}9}$

$i = 2 \overset{?}{\leqslant} q - p + 1 = 9 - 8 + 1 = 2$ *(igaz)*, $\Longrightarrow$

3.*KIVÁLASZT*(A, 8, 9, $9\text{-}8\text{+}1\text{=}2$, x)
p r i

	49,	45
med		45
medmed		45

FELOSZT$(A, p = 8, r = 9, x = 45, q)$



$\boxed{q\text{=}8}$

IF $i = 2 \overset{?}{\leqslant} p - q + 1 = 8 - 8 + 1 = 1$ *(nem)* $\Longrightarrow$

4.*KIVÁLASZT*(A, $q + 1 = 8 + 1 = 9$, $r = 9$, $2\text{-}(8\text{-}8\text{+}1)\text{=}1$, x)
p r i

IF $p = r$ *THEN* $x \longleftarrow A[p] = 49$ *RETURN*$(q = 49)$.
Tehát az A tömb 9. eleme a 49.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

Ellenőrzésképpen minimumkiválasztás rendezés segítségével megkeressük az A tömb 9. elemét.

$$A = \Big[\boxed{17}, 127, 20, 45, 81, \boxed{3}, 10, 49, 50, 28, 12, 82, 9\Big]$$

$$A = \Big[3 || \boxed{127}, 20, 45, 81, 17, 10, 49, 50, 28, 12, 82, \boxed{9}\Big]$$

$$A = \Big[3, 9 || \boxed{20}, 45, 81, 17, \boxed{10}, 49, 50, 28, 12, 82, 127\Big]$$

$$A = \Big[3, 9, 10 || \boxed{45}, 81, 17, 20, 49, 50, 28, \boxed{12}, 82, 127\Big]$$

$$A = \Big[3, 9, 10, 12 || \boxed{81}, \boxed{17}, 20, 49, 50, 28, 45, 82, 127\Big]$$

$$A = \Big[3, 9, 10, 12, 17 || \boxed{81}, \boxed{20}, 49, 50, 28, 45, 82, 127\Big]$$

$$A = \Big[3, 9, 10, 12, 17, 20 || \boxed{81}, 49, 50, \boxed{28}, 45, 82, 127\Big]$$

$$A = \Big[3, 9, 10, 12, 17, 20, 28 || \boxed{49}, 50, 81, \boxed{45}, 82, 127\Big]$$

$$A = \Big[3, 9, 10, 12, 17, 20, 28, 45 || \boxed{50}, 81, \boxed{49}, 82, 127\Big]$$

$$A = \Big[3, 9, 10, 12, 17, 20, 28, 45, \boxed{49}, 81, 50, 82, 127\Big]$$

Látható, hogy $A[9] = 49$, azonban ehhez az eredményhez majdnem rendezni kellett az egész tömböt.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.8.5. A gyorsrendezés

Talán a legtrükkösebb rendezés a gyorsrendezés, ami a nevéhez híven valóban gyors a maga $n\log(n)$-es átlagos futási idejével. Ez a rendezés is használja a feloszt algoritmust.

2. Gyorsrendezés: $T(n) = \Theta(n^2)$, *átlagosan* $T(n) = \Theta(n\log(n))$.
$GY(A, p, r)$
INPUT: Az A *tömb* $A[p, \ldots, r]$ *résztömbjét rendezzük.*

OUTPUT: Az A tömb a rendezett $A[p,\dots,r]$ résztömbökkel.

	$GY(A,p,r)$
1.	*IF* $p < r$
2.	*THEN* $FELOSZT(A,p,r,A[p],q)$
3.	$GY(A,p,q)$
4.	$GY(A,q+1,r)$
5.	*RETURN*(A)

Feladat. *A gyorsrendezés segítségével rendezzük az alábbi A tömböt.*

$A = [91, 117, 52, 11, 63, 49]$

$GY(A,1,6)$

① $Felosz(A,1,6, x=A[1]=91, q)$

1.) $FELOSZT(A,1,6,x=A[1]=91,q)$

		49				117	91		
$A=[$	~~91~~,	~~117~~,	52,	11,	~~63~~,	49	$]$	
$i\uparrow$	$\uparrow$						$\uparrow$	$\uparrow j$	$x=91$
		$\uparrow$				$\uparrow$			
				$\uparrow j$	$\uparrow i$				

$q \longleftarrow j = 4$

$A = [91, 117, 52, 11, 63, 49]$

$GY(A,1,6)$

① $Felosz(A,1,6, x=A[1]=91, q)$

$A = [49, 63, 52, 11, 117, 91], q = 4$

$GY(A,1,4)$ $\qquad$ $GY(A,5,6)$

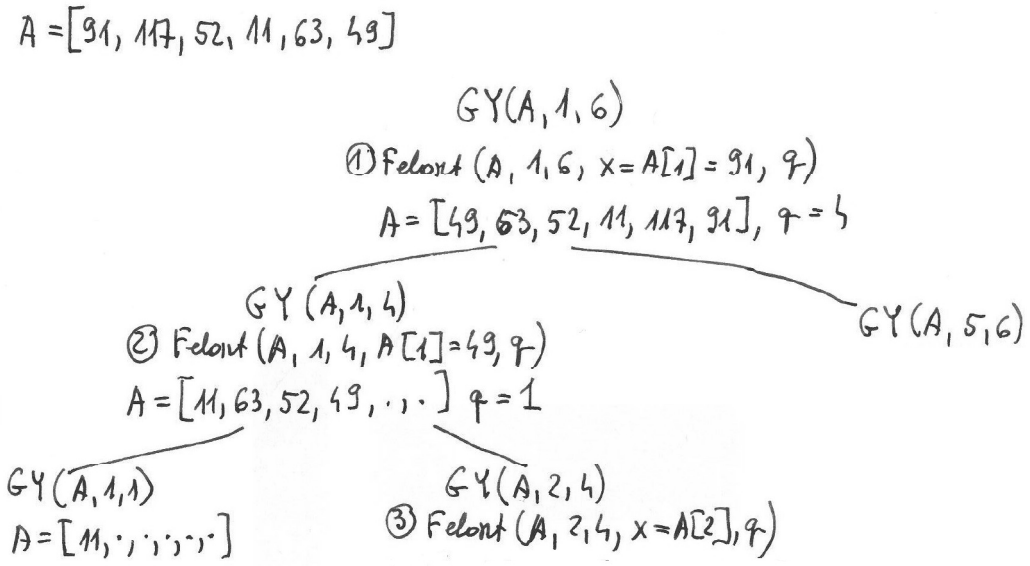
② $Felosz(A,1,4, A[1]=49, q)$

2.) $FELOSZT(A,1,4,x=A[1]=49,q)$

	11			49	
$A=[$	~~49~~,	63,	52,	~~11~~,	$]$
$\uparrow$					$\uparrow$
	$\uparrow$			$\uparrow$	
$\uparrow j=1$	$\uparrow$				

$x = 49$

$$RETURN\,(q \longleftarrow j = 1)\,,$$

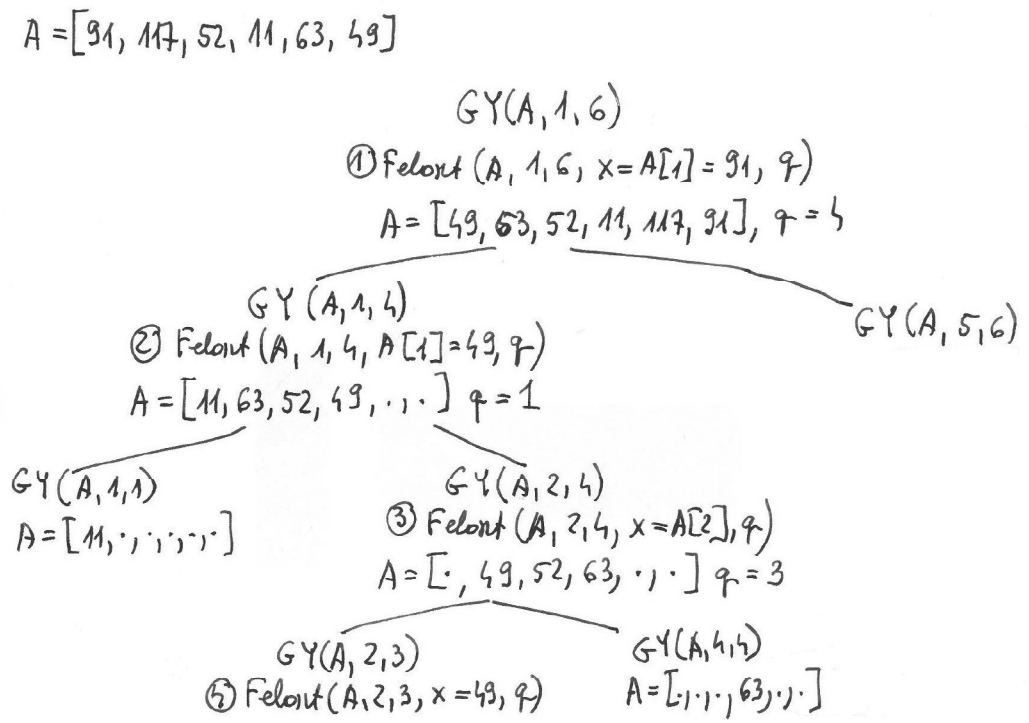


3.) $FELOSZT(A, 2, 4, x = A[2] = 63, q)$

$$\begin{array}{lccccccc} & & & 49 & & 63 & & \\ A = [& \cdot, & \underline{63}, & 52, & \underline{49}, & \cdot, & \cdot &] \\ & \uparrow & & & & \uparrow & & \end{array} \qquad x = 63$$

$$\begin{array}{ccccc} & & \uparrow & & \uparrow \\ & & \uparrow j & & \uparrow i \end{array}$$

$$RETURN\,(q \longleftarrow j = 3)\,,$$



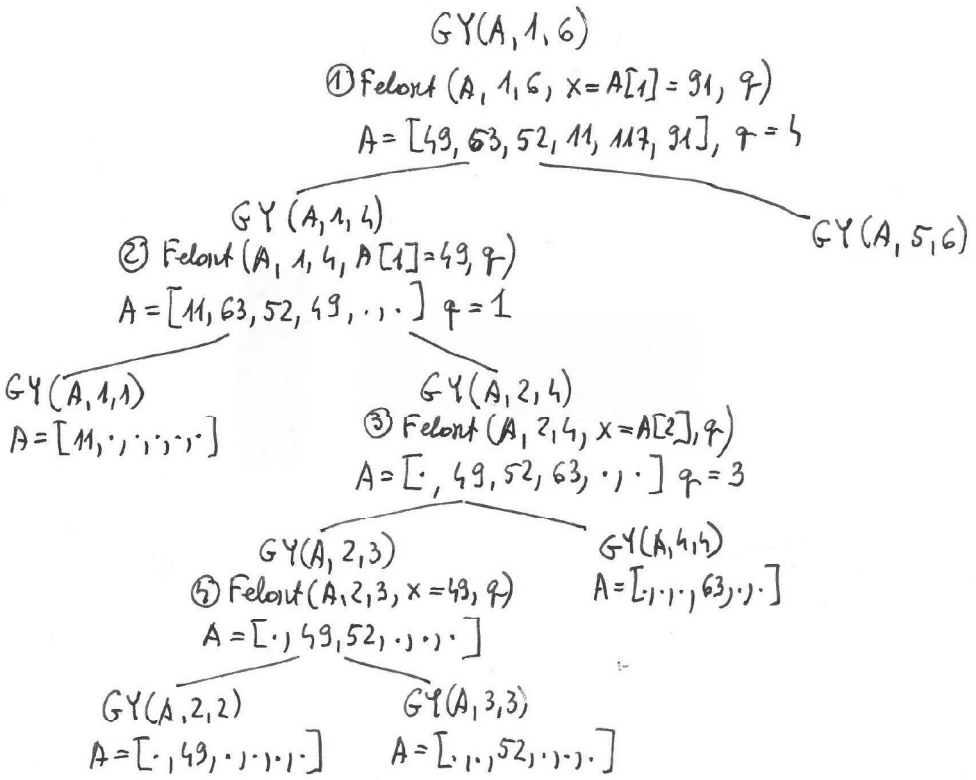
4.) $FELOSZT(A, 2, 3, x = 49, q)$

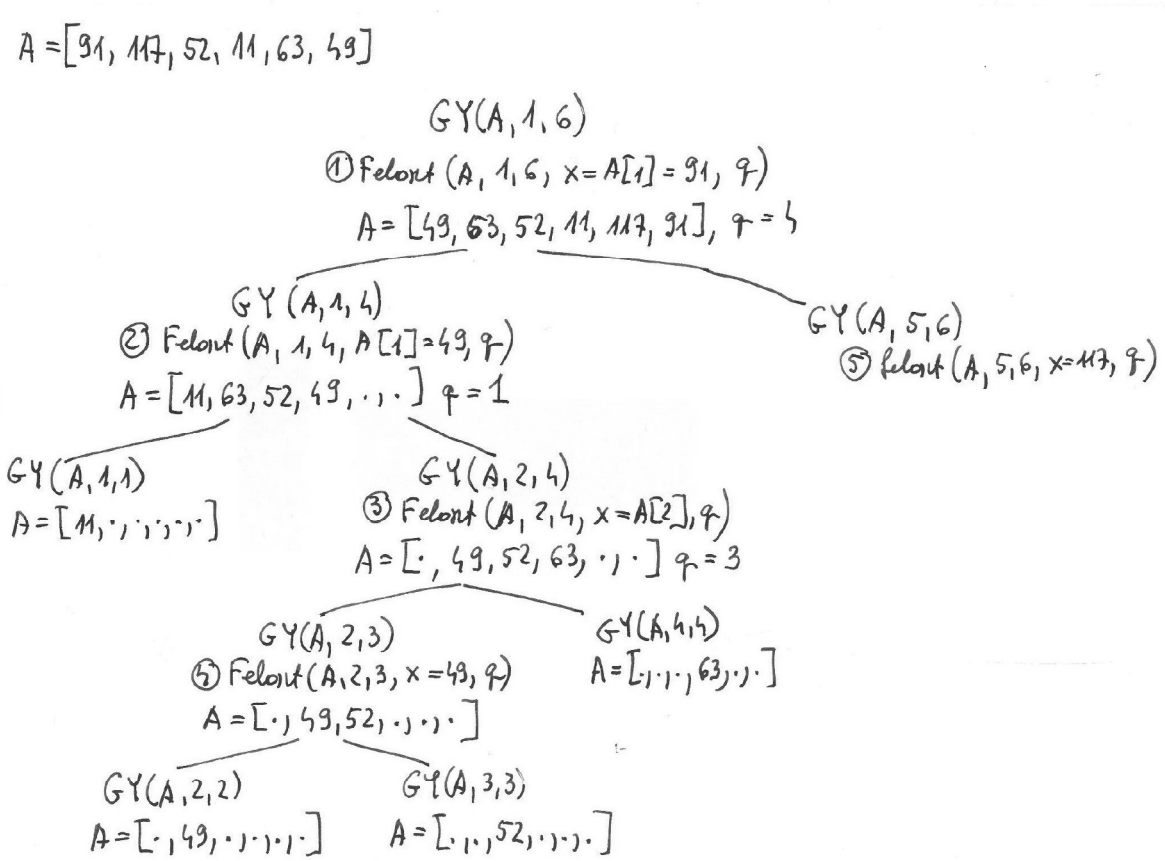
$$A = [\ \cdot,\ 49,\ 52,\ \cdot,\ \cdot,\ \cdot\ ] \qquad x = 49$$

$\uparrow j \quad \uparrow i$

$$RETURN\,(q \longleftarrow j = 2)\,,$$

A = [91, 117, 52, 11, 63, 49]

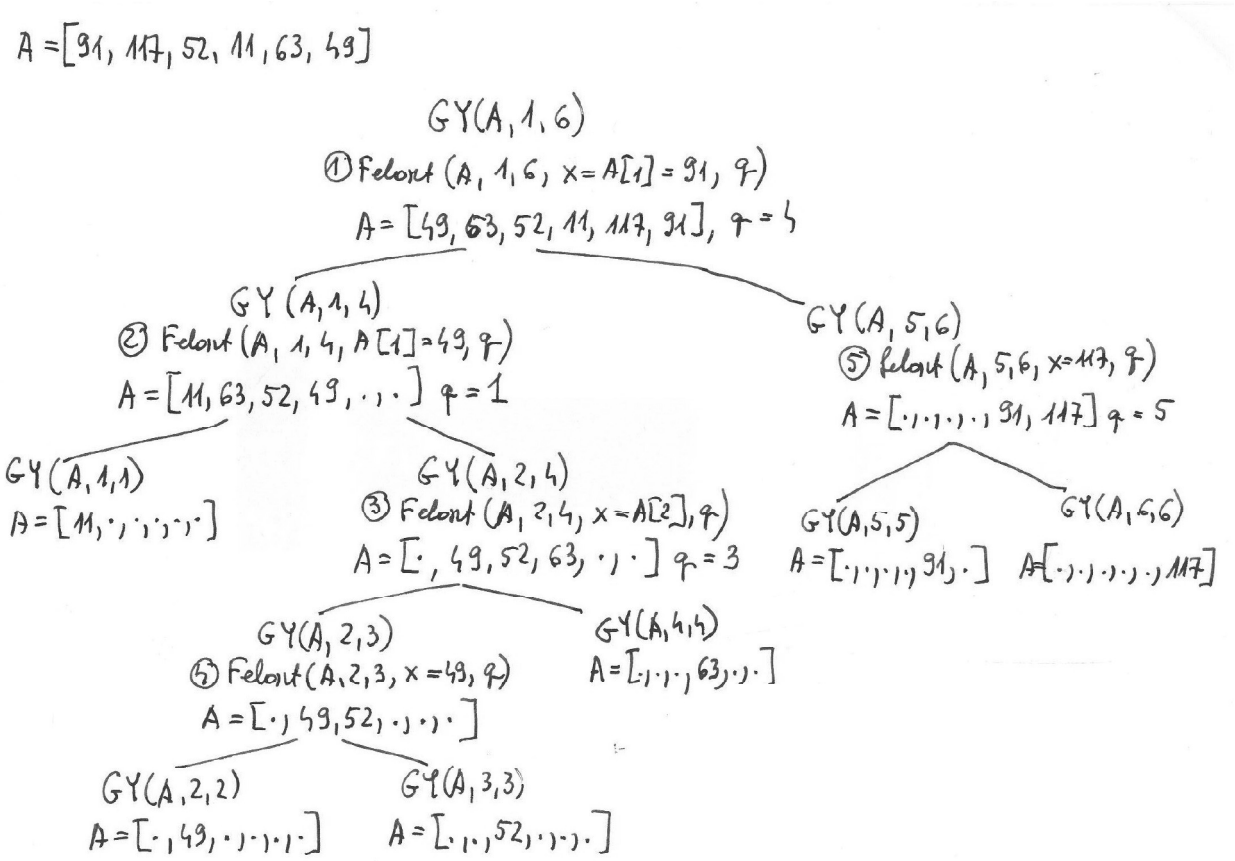




5.) $FELOSZT(A, 5, 6, x = 117, q)$

						91	117		
$A = [$	$\cdot,$	$\cdot,$	$\cdot$	$\cdot,$		~~117~~,	~~91~~	$]$	$x = 117$
				↑				↑	
						↑	↑		
						↑ j	↑ i		

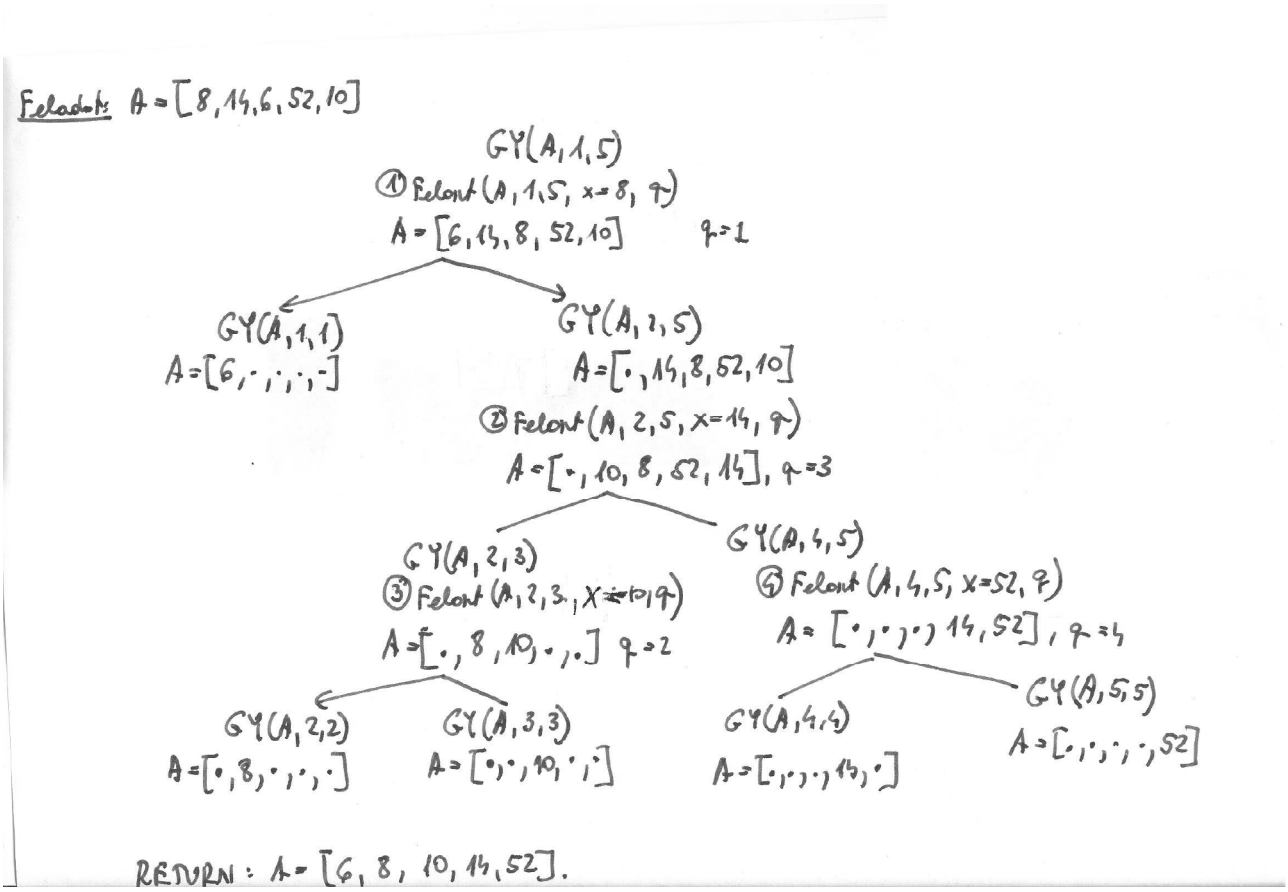
$$RETURN\,(q \longleftarrow j = 5)\,,$$



RETURN

A = [11, 49, 52, 63, 91, 117]

Feladat. *A gyorsrendezés segítségével rendezzük az alábbi A tömböt.*



1.) $FELOSZT(A, 1, 5, x = 8, q)$

$$A = [\ \overset{6}{\cancel{8}},\ 14,\ \overset{8}{\cancel{6}},\ 52,\ 10,\ ] \qquad x = 8$$

$$\uparrow j = 1$$

$$RETURN\,(q \longleftarrow j = 1)\,,$$

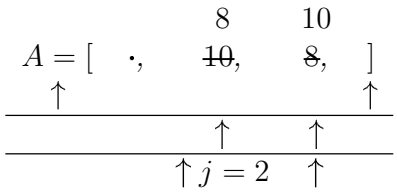
2.) $FELOSZT(A, 2, 5, x = 14, q)$

$$A = [\ \cdot,\ \overset{10}{\cancel{14}},\ 8,\ 52,\ \overset{14}{\cancel{10}},\ ] \qquad x = 14$$

$$\uparrow j = 1$$

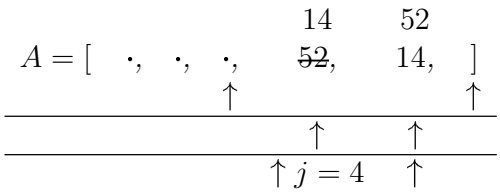
$$RETURN\,(q \longleftarrow j = 1)\,,$$

3.) $FELOSZT(A, 2, 3, x = 10, q)$



$$RETURN\,(q \longleftarrow j = 2)\,,$$

4.) $FELOSZT(A, 4, 5, x = 52, q)$



$$RETURN\,(q \longleftarrow j = 4)\,,$$

0.9. Rendezések II

0.9.1. A leszámláló rendezés

Először megismerkedünk a leszámláló rendezéssel. A "bicaj, decaj" mondóka segít a pszeudokód memorizálásában. A leszámláló rendezés egy lineáris idejű stabil rendezés. Ennek persze ára van. Olyan pozitív egészek rendezésére alkalmas, amelyek kis felső korláttal rendelkeznek. Bár át lehet írni úgy az algoritmust, hogy egy intervallumból származó egészeket rendezze, feltéve, hogy az intervallum nem túl hosszú a rendezendő tömb elemszámához képest. Az algoritmus erősen kihasználja azt a tényt, hogy pozitív egészeket rendez, mert egy pozitív egész számról kapásból meg lehet mondani, hogy hányadik a pozitív egészek rendezett sorozatában. Ugyanez például nem mondható el a pozitív racionális számok rendezett halmazáról, mivel ez utóbbi halmaz (matematikai értelemben) sűrű.

3. Leszámláló rendezés *LESZREND*(A, B)
INPUT: $A[1..n]$ *a rendezendő tömb,* $k = \max\{A[i] \mid i = 1, 2, \ldots, n\}$.
OUTPUT: $B[1..n]$*, ami az* A *elemeit tartalmazza rendezve.*

(Az algoritmus használ egy $C[1..k]$ *segédtömböt, ahol* k *a rendezendő elemek egy felső korlátja.)*

	LESZREND(A, B)
1.	*FOR* $i \longleftarrow 1$, *TO* k *DO*
2.	$\quad C[i] \longleftarrow 0$
3.	*FOR* $j \longleftarrow 1$ *TO* n *DO*
4.	$\quad \mathrm{INC}\,(C\,[A[j]])$
5.	*FOR* $i \longleftarrow 2$ *TO* k *DO*
6.	$\quad C[i] \longleftarrow C[i-1] + C[i]$
7.	*FOR* $j \longleftarrow n$ *DOWNTO* 1 *DO*
8.	$\quad B\,[C\,[A[j]]] \longleftarrow A[j]$
9.	$\quad \mathrm{DEC}\,(C\,[A[j]])$
10.	*RETURN(B)*

Az algoritmus működését egy konkrét példa segítségével magyarázzuk el.

1. feladat. *Rendezzük leszámláló rendezéssel az alábbi tömböt*

$$A = [4, 2, 4, 1, 4, 6, 1, 4]\,.$$

$$B = [\cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot]$$
$$C = [\cdot, \cdot, \cdot, \cdot, \cdot, \cdot]$$

1. lépés. *A* C *tömb inicializálása.*

FOR $i \longleftarrow 1$ *TO* hossz$[C]$ *DO*
$C[i] \longleftarrow 0$

$$C = [0, 0, 0, 0, 0, 0,]$$

2. lépés.

$$\boxed{\begin{array}{l} FOR\ j \longleftarrow 1\ TO\ \text{hossz}[A]\ DO \\ \text{INC}\,(C\,[A[j]]) \end{array}}$$

$$\begin{array}{lll} j=1 & \text{INC}\,(C\,[A[1]]): & C=[0,0,0,1,0,0] \\ j=2 & \text{INC}\,(C\,[A[2]]): & C=[0,1,0,1,0,0] \\ j=3 & \text{INC}\,(C\,[A[3]]): & C=[0,1,0,2,0,0] \\ j=4 & \text{INC}\,(C\,[A[4]]): & C=[1,1,0,2,0,0] \\ j=5 & \text{INC}\,(C\,[A[5]]): & C=[1,1,0,3,0,0] \\ j=6 & \text{INC}\,(C\,[A[6]]): & C=[1,1,0,3,0,1] \\ j=7 & \text{INC}\,(C\,[A[7]]): & C=[2,1,0,3,0,1] \\ j=8 & \text{INC}\,(C\,[A[8]]): & C=[2,1,0,4,0,1] \end{array}$$

A 2. lépés hatására a $C[i]$ azt mondja meg, hogy hány darab i szám van az A többen, azaz képletekkel $C[i] = |\{k \mid A[k] = i\}|$.

3. lépés.

$$\boxed{\begin{array}{l} FOR\ i \longleftarrow 2\ TO\ \text{hossz}[C]\ DO \\ C[i] \longleftarrow C[i] + C[i-1] \end{array}}$$

Ez a lépés egy részletösszeg-sorozat képzést valósít meg. A felső sor mutatja a C tömb elemeit a részletösszeg sorozat képzés előtt, az alsó utána:

$$C = [2,1,0,4,0,1]$$

$$C = \left[2, \underbrace{2+1}_{3}, \underbrace{2+1+0}_{3}, \underbrace{2+1+0+4}_{7}, \underbrace{2+1+0+4+0}_{7}, \underbrace{2+1+0+4+0+1}_{8}\right]$$

$$\begin{array}{ll} i=2 & C[2] \longleftarrow C[2] + C[1] = 1+2 = 3 \\ i=3 & C[3] \longleftarrow C[3] + C[2] = 0+3 = 3 \\ i=4 & C[4] \longleftarrow C[4] + C[3] = 4+3 = 7 \\ i=5 & C[5] \longleftarrow C[5] + C[4] = 0+7 = 7 \\ i=6 & C[6] \longleftarrow C[6] + C[5] = 1+7 = 8 \end{array}$$

Tehát a megváltozott C tömb:

$$C = [2,3,7,7,8]$$

A 3. lépés hatására most $C[i] = |\{k \mid A[k] \leqslant i\}|$. Ez azt jelenti, hogy ha $A[j] = i$ és ez az j a legutolsó index az A tömbben, amelyre $A[j] = i$, akkor $C[i]$ megmondja azt az indexet, hogy az $A[j]$ elemnek a B tömbben hová kell kerülnie.

Az első 3 lépést nem kell ennyire részletezni a Z.H.-ban, persze a pszeudokód és a futás eredménye kell. A most következő részt viszont valamilyen módon kérem kiírni.

4. lépés. *(a bicaj decaj rész)*

```
FOR j ⟵ Hossz[A] DOWTO 1 DO
B [C [A[j]]] ⟵ A[j]
DEC (C [A[j]])
```

$$j = 8 \qquad \underbrace{B[\underbrace{C[\underbrace{A[8]}_{4}]}_{7}]}_{} \longleftarrow \underbrace{A[8]}_{4} \qquad C = [2, 3, 3, \overset{6}{\not{7}}, 7, 8]$$

Ezt a lépést most egy kicsit részletezzük. $j = 8$, $A[8] = 4$. *A 2. lépés végén* $C[4] = 4$ *volt, ami azt jelentette, hogy a 4-es számból (index) 4 darab (érték) van az A tömbben. A 3. lépés végére* $C[4] = 7$, *ami azt jelenti, hogy a legnagyobb indexű 4-es az A tömbben, ami az* $A[8]$ *a B-ben - ami az A tömb elemeit tartalmazza - a 7. helyre kell kerülnie. Úgyhogy az* $A[8] = 4$*-et beszúrjuk a* $B[7]$*-be, majd ezt követően a* $C[4]$ *értékét 7-ről 6-ra csökkentjük, hogy a második legnagyobb indexű 4-es a* $B[6]$*-ba tudjon kerülni. Futtassuk tovább az algoritmust.*

$$j = 7 \qquad \underbrace{B[\underbrace{C[\underbrace{A[7]}_{1}]}_{2}]}_{} \longleftarrow \underbrace{A[7]}_{1} \qquad C = [\overset{1}{\not{2}}, 3, 3, 6, 7, 8]$$

$$j = 6 \qquad \underbrace{B[\underbrace{C[\underbrace{A[6]}_{6}]}_{8}]}_{} \longleftarrow \underbrace{A[6]}_{6} \qquad C = [1, 3, 3, 6, 7, \overset{7}{\not{8}}]$$

$$j = 5 \qquad \underbrace{B[\underbrace{C[\underbrace{A[5]}_{4}]}_{6}]}_{} \longleftarrow \underbrace{A[5]}_{4} \qquad C = [1, 3, 3, \overset{5}{\not{6}}, 7, 7]$$

$$j = 4 \qquad \underbrace{B[\underbrace{C[\underbrace{A[4]}_{1}]}_{1}]}_{} \longleftarrow \underbrace{A[4]}_{1} \qquad C = [\overset{0}{\not{1}}, 3, 3, 5, 7, 7]$$

$$j = 3 \qquad \underbrace{B[\underbrace{C[\underbrace{A[3]}_{4}]}_{5}]}_{} \longleftarrow \underbrace{A[3]}_{4} \qquad C = [0, 3, 3, \overset{4}{\not{5}}, 7, 7]$$

$$j = 2 \qquad \underbrace{B[\underbrace{C[\underbrace{A[2]}_{2}]}_{3}]}_{} \longleftarrow \underbrace{A[2]}_{2} \qquad C = [0, \overset{2}{\not{3}}, 3, 4, 7, 7]$$

$$j = 1 \qquad \underbrace{B[\underbrace{C[\underbrace{A[1]}_{4}]}_{4}]}_{} \longleftarrow \underbrace{A[1]}_{4} \qquad C = [0, 2, 3, \overset{3}{\not{4}}, 7, 7]$$

Az eredeti tömb: $A = [4, 2, 4, 1, 4, 6, 1, 4]$.
A rendezett tömb: $B = [1, 1, 2, 4, 4, 4, 4, 6]$.

2. feladat. *Rendezzük leszámláló rendezéssel az*

$$A = [2, 1, 2, 4, 3, 3]$$

tömböt.

$$B = [\cdot, \cdot, \cdot, \cdot, \cdot, \cdot]$$
$$C = [\cdot, \cdot, \cdot, \cdot]$$

1. lépés. *A C tömb inicializálása.*

$$\begin{array}{l} FOR\ i \longleftarrow 1\ TO\ \text{hossz}[C]\ DO \\ C[i] \longleftarrow 0 \end{array} \qquad C = [0, 0, 0, 0]$$

2. lépés.

$$\begin{array}{l} FOR\ j \longleftarrow 1\ TO\ \text{hossz}[A]\ DO \\ \text{INC}\,(C\,[A[j]]) \end{array}$$

$C = [1, 2, 2, 1]$
$C[i]$*: az i értékből hány darab van.*

3. lépés.

$$\begin{array}{l} FOR\ i \longleftarrow 2\ TO\ \text{hossz}[C]\ DO \\ C[i] \longleftarrow C[i] + C[i-1] \end{array}$$

$C = [1, 3, 5, 6]$
$C[i] = \{k | A[k] \leqslant i\}$

4. lépés.

$$\begin{array}{l} FOR\ j \longleftarrow \text{Hossz}[A]\ DOWTO\ 1\ DO \\ B\,[C\,[A[j]]] \longleftarrow A[j] \\ \text{DEC}\,(C\,[A[j]]) \end{array}$$

$$j = 6 \qquad \underbrace{\underbrace{B[C[\underbrace{A[6]}_{3}]]}_{5} \longleftarrow \underbrace{A[6]}_{3}}_{} \qquad C = [1, 3, \overset{4}{\cancel{5}}, 6]$$

$$j = 5 \qquad \underbrace{\underbrace{B[C[\underbrace{A[5]}_{3}]]}_{4} \longleftarrow \underbrace{A[5]}_{3}}_{} \qquad C = [1, 3, \overset{3}{\cancel{4}}, 6]$$

$$j = 4 \qquad \underbrace{\underbrace{B[C[\underbrace{A[4]}_{4}]]}_{6} \longleftarrow \underbrace{A[4]}_{4}}_{} \qquad C = [1, 3, 3, \overset{5}{\cancel{6}}]$$

$$j = 3 \qquad \underbrace{\underbrace{B[C[\underbrace{A[3]}_{2}]]}_{3} \longleftarrow \underbrace{A[3]}_{2}}_{} \qquad C = [1, \overset{2}{\cancel{3}}, 3, 5]$$

$$j = 2 \qquad \underbrace{\underbrace{B[C[\underbrace{A[2]}_{1}]]}_{1} \longleftarrow \underbrace{A[2]}_{1}}_{} \qquad C = [\overset{0}{\cancel{1}}, 2, 3, 5]$$

$$j = 1 \qquad \underbrace{\underbrace{B[C[\underbrace{A[1]}_{2}]]}_{2} \longleftarrow \underbrace{A[1]}_{2}}_{} \qquad C = [0, \overset{0}{\cancel{1}}, 3, 5]$$

Az eredeti tömb: $A = [2, 1, 2, 4, 3, 3]$.

A rendezett tömb: $B = [1, 2, 2, 3, 3, 4]$.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.9.2. 4. A buborék rendezés

4. Buborékrendezés *: BUBIREND*(A), $T(n) = \Theta(n^2)$ *helyben rendezés, stabil rendezés. A buborékrendezés szomszédos cserékkel rendezi a tömb elemeit.*

INPUT: $A[1..n]$ *a rendezendő tömb.*

OUTPUT: $A[1..n]$ *a rendezett tömb.*

	BUBIREND(A)
1.	$i \longleftarrow 2$
2.	*REPEAT*
3.	$\quad$ *nemvoltcsere* $\longleftarrow$ *igaz*
4.	$\quad$ *FOR* $j \longleftarrow n$ *DOWTO* i *DO*
5.	$\quad\quad$ *IF* $A[j] < A[j-1]$
6.	$\quad\quad\quad$ *THEN CSERE* $(A[j], A[j-1])$
7.	$\quad\quad\quad$ *nemvoltcsere* $\longleftarrow$ *hamis*
8.	$\quad$ *INC* (i)
9.	*UNTIL nemvoltcsere*
10.	*RETURN(A)*

Feladat. *Rendezzük buborékrendezéssel az alábbi tömböt:*

$$A = [315, 22, 8, 3, 13].$$

$i = 2$

$$A = [315, 22, \overset{3}{\cancel{8}}, \overset{8}{\cancel{3}}, 13]$$

$$A = [315, \overset{3}{\cancel{22}}, \overset{22}{\cancel{3}}, 8, 13]$$

$$A = [\overset{3}{\cancel{315}}, \overset{315}{\cancel{3}}, 22, 3, 13]$$

$i = 3$

$$A = [3||315, \overset{8}{\cancel{22}}, \overset{22}{\cancel{8}}, 13]$$

$$A = [3|| \overset{8}{\cancel{315}}, \overset{315}{\cancel{8}}, 22, 13]$$

$i = 4$

$$A = [3, 8||315, \overset{13}{\cancel{22}}, \overset{22}{\cancel{13}}]$$

$$A = [3, 8|| \overset{13}{\cancel{315}}, \overset{315}{\cancel{13}}, 22]$$

$i = 5$

$$A = [3, 8, 13|| \overset{22}{\cancel{315}}, \overset{315}{\cancel{22}}]$$

$i = 6$ *A for ciklus nem indul be, a nemvoltcsere igaz marad, így kilép az algoritmus a repeat-until ciklusból és visszatér a rendezett tömbbel.*
RETURN(A)

Az előző feladat *egy kicsit részletesebben leírva:*

$i = 2$ $\quad A = [315, 22, 8, 3, 13]$ $\quad$ *nemvoltcsere* $\longleftarrow$ *igaz*

$j = 5$ $\quad A[5] < A[4]$ $\quad$ *hamis*
$j = 4$ $\quad A[4] < A[3]$ $\quad$ *igaz* $\quad \Longrightarrow \quad$ CSERE$(A[4], A[3])$
$\quad A = [315, 22, \overleftrightarrow{8, 3}, 13],$ $\quad$ *nemvoltcsere* $\longleftarrow$ *hamis*
$j = 3$ $\quad A[3] < A[2]$ $\quad$ *igaz* $\quad \Longrightarrow \quad$ CSERE$(A[3], A[2])$
$\quad A = [315, \overleftarrow{22, 3}, 8, 13],$ $\quad$ *nemvoltcsere* $\longleftarrow$ *hamis*
$j = 2$ $\quad A[2] < A[1]$ $\quad$ *igaz* $\quad \Longrightarrow \quad$ CSERE$(A[2], A[1])$
$\quad A = [\overleftarrow{315, 3}, 22, 8, 13],$ $\quad$ *nemvoltcsere* $\longleftarrow$ *hamis*

$i = 3$ $\quad A = [3, 315, 22, 8, 13]$ $\quad$ *nemvoltcsere* $\longleftarrow$ *igaz*

$j = 5$ $\quad A[5] < A[4]$ $\quad$ *hamis*
$j = 4$ $\quad A[4] < A[3]$ $\quad$ *igaz* $\quad \Longrightarrow \quad$ CSERE$(A[4], A[3])$
$\quad A[3, 315, \overleftarrow{22, 8}, 13],$ $\quad$ *nemvoltcsere* $\longleftarrow$ *hamis*
$j = 3$ $\quad A[3] < A[2]$ $\quad$ *igaz* $\quad \Longrightarrow \quad$ CSERE$(A[3], A[2])$
$\quad A = [3, \overleftarrow{315, 8}, 22, 13],$ $\quad$ *nemvoltcsere* $\longleftarrow$ *hamis*

$i = 4$ $\quad A = [3, 8, 315, 22, 13]$ $\quad$ *, nemvoltcsere* $\longleftarrow$ *igaz*

$j = 5$ $\quad A[5] < A[4]$ $\quad$ *igaz* $\quad \Longrightarrow \quad$ CSERE$(A[5], A[4])$
$\quad A[3, 8, 315, \overleftarrow{22, 13}],$ $\quad$ *nemvoltcsere* $\longleftarrow$ *hamis*
$j = 4$ $\quad A[4] < A[3]$ $\quad$ *igaz* $\quad \Longrightarrow \quad$ CSERE$(A[4], A[3])$
$\quad A = [3, 8, \overleftarrow{315, 13}, 22],$ $\quad$ *nemvoltcsere* $\longleftarrow$ *hamis*

$i = 5$ $\quad A = [3, 8, 13, 315, 22]$ $\quad$ *nemvoltcsere* $\longleftarrow$ *igaz*

$j = 5$ $\quad A[5] < A[4]$ $\quad$ *igaz* $\quad \Longrightarrow \quad$ CSERE$(A[5], A[4])$
$\quad A[3, 8, 13, \overleftarrow{315, 22}],$ $\quad$ *nemvoltcsere* $\longleftarrow$ *hamis*

$i = 6$ $\quad A = [3, 8, 13, 22, 315]$ $\quad$ *A for ciklus már nem indul be.*

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.9.3. A beszúró rendezés

5. Beszúró rendezés *: BESZREND*(A), $T(n) = \Theta(n^2)$ *helyben rendezés, stabil rendezés. INPUT:* $A[1..n]$ *a rendezendő tömb.*

OUTPUT: $A[1..n]$ *a rendezett tömb.*

	BESZREND(A)
1.	*FOR* $i \longleftarrow 2$ *TO* n *DO*
2.	$\quad x \longleftarrow A[i]$
3.	$\quad j \longleftarrow i-1$
4.	$\quad$ *WHILE* $(j > 0$ *és* $x < A[j])$ *DO*
5.	$\quad\quad A[j+1] \longleftarrow A[j]$
6.	$\quad\quad$ *DEC*(j)
7.	$\quad A[j+1] \longleftarrow x$
8.	*RETURN(A)*

Feladat. *Beszúró rendezéssel rendezzük a*

$$A = [315, 22, 8, 3, 13]$$

tömböt.

Először csak nagy vonalakban átfutjuk, hogy hogyan működik az algoritmus.
$A = [315||22, 8, 3, 13]$

$$\begin{array}{lll} i=2 & x \longleftarrow A[2] = 22 & A = [22, 315||8, 3, 13] \\ i=3 & x \longleftarrow A[3] = 8 & A = [8, 22, 315||3, 13] \\ i=4 & x \longleftarrow A[4] = 3 & A = [3, 8, 22, 315||13] \\ i=5 & x \longleftarrow A[5] = 13 & A = [3, 8, 13, 22, 315] \end{array}$$

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

Most nézzük meg egy kicsit részletesebben:

$i = 2$

$$\begin{array}{lccccccc} & & 22 & 315 & & & & \\ A = [& \cancel{315}, & \cancel{22}, & 8, & 3, & 13] & & \\ & \uparrow & \downarrow & & & & & \\ & j & x & & & & & \end{array}$$

$i = 3$

$$\begin{array}{lccccccc} & & 8 & 22 & 315 & & & \\ A = [& \cancel{22}, & \cancel{315}|| & \cancel{8}, & 3, & 13] & & \\ & \uparrow & \uparrow & \uparrow & \downarrow & & & \\ & j & j & j & x & & & \end{array}$$

$i = 4$

$$\begin{array}{lccccccc} & & 3 & 8 & 22 & 315 & & \\ A = [& \cancel{8}, & \cancel{22}, & \cancel{315}||, & \cancel{3}, & 13] & & \\ & \uparrow & \uparrow & \uparrow & \uparrow & \downarrow & & \\ & j & j & j & j & x & & \end{array}$$

$i = 5$

$$\begin{array}{rccccccc} & & & & 13 & 22 & 315 & \\ A = [& 3 & 8, & \overline{22}, & \overline{315}||, & \overline{13}] & & \\ & & \uparrow & \uparrow & \uparrow & \downarrow & & \\ & & j & j & j & x & & \end{array}$$

RETURN $A = [3, 8, 13, 22, 315]$

Ugyanez még részletesebben:

$A = [315, 22, 8, 3, 13]$

$A = [315||22, 8, 3, 13]$

$i = 2 \qquad x \longleftarrow A[2] = 22$

$$\begin{array}{llcll} j = 1 & (1 > 0 \text{ és } A[1] > x) \text{ igaz} & \Longrightarrow & A[2] \longleftarrow A[1] & (A[315, 315, 8, 3, 13]) \\ j = 0 & (0 > 0) \text{ hamis} & \Longrightarrow & A[1] \longleftarrow x & (A[22, 315||8, 3, 13]) \end{array}$$

$i = 3 \qquad x \longleftarrow A[3] = 8$

$$\begin{array}{llcll} j = 2 & (2 > 0 \text{ és } A[2] > x) \text{ igaz} & \Longrightarrow & A[3] \longleftarrow A[2] & (A[22, 315, 315, 3, 13]) \\ j = 1 & (1 > 0 \text{ és } A[1] > x) \text{ igaz} & \Longrightarrow & A[2] \longleftarrow A[1] & (A[22, 22, 315, 3, 13]) \\ j = 0 & (0 > 0) \text{ hamis} & \Longrightarrow & A[1] \longleftarrow x & (A[8, 22, 315||3, 13]) \end{array}$$

$i = 4 \qquad x \longleftarrow A[4] = 3$

$$\begin{array}{llcll} j = 3 & (3 > 0 \text{ és } A[3] > x) \text{ igaz} & \Longrightarrow & A[4] \longleftarrow A[3] & (A[8, 22, 315, 315, 13]) \\ j = 2 & (2 > 0 \text{ és } A[2] > x) \text{ igaz} & \Longrightarrow & A[3] \longleftarrow A[2] & (A[8, 22, 22, 315, 13]) \\ j = 1 & (1 > 0 \text{ és } A[1] > x) \text{ igaz} & \Longrightarrow & A[2] \longleftarrow A[1] & (A[8, 8, 22, 315, 13]) \\ j = 0 & (0 > 0) \text{ hamis} & \Longrightarrow & A[1] \longleftarrow x & (A[3, 8, 22, 315||13]) \end{array}$$

$i = 5 \qquad x \longleftarrow A[5] = 13$

$$\begin{array}{llcll} j = 4 & (4 > 0 \text{ és } A[4] > x) \text{ igaz} & \Longrightarrow & A[5] \longleftarrow A[4] & (A[3, 8, 22, 315, 315]) \\ j = 3 & (3 > 0 \text{ és } A[3] > x) \text{ igaz} & \Longrightarrow & A[4] \longleftarrow A[3] & (A[3, 8, 22, 22, 315]) \\ j = 2 & (2 > 0 \text{ és } A[2] > x) \text{ hamis} & \Longrightarrow & A[3] \longleftarrow x & (A[3, 8, 13, 22, 315]) \end{array}$$

$A = [3, 8, 13, 22, 315]$ RETURN

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.9.4. A Shell rendezés

6. Shell rendezés *fogyó növekményekkel:*

Egymástól távoli elemeket mozgat. (Hasonlítsuk össze a Shell rendezés pszeudokódját a beszúró rendezés pszeudokódjával.)

***Növekmények**: $h[1..t]$ $(t = \lceil \log(n) \rceil - 1)$*

$h[1] < h[2] < \cdots < h[t]$ növekmények tömbjei csökkenő módon járjuk be.
Összegyűjtöttünk néhány tipikus módszert, amelynek segítségével a növekmények tömbjét fel tudjuk tölteni.

$$h[1] = 1, \quad h[i+1] = 2h[i] \quad \Longrightarrow \quad h = [1, 2, 4]$$
$$h[1] = 1, \quad h[i+1] = 3h[i] + 1 \quad \Longrightarrow \quad h = [1, 4, 13]$$
$$h[1] = 1, \quad h[i+1] = 2h[i] - 1 \quad \Longrightarrow \quad h = [1, 3, 7]$$

Az algoritmus:
SHELLREND(A, h)
INPUT: $A[1..n]$ *a rendezendő tömb,* $h[1..t]$ *növekmények.*
OUTPUT: $A[1..n]$ *a rendezett tömb.*

	SHELLREND(A, h)
1.	*FOR* $s \longleftarrow t$ *DOWNTO* 1 *DO*
2.	$m \longleftarrow h[s]$
3.	*FOR* $i \longleftarrow m+1$ *TO* n *DO*
4.	$x \longleftarrow A[i]$
5.	$j \longleftarrow i - m$
6.	*WHILE* $(j > 0$ *és* $x < A[j])$ *DO*
7.	$A[j+m] \longleftarrow A[j]$
8.	$j \longleftarrow j - m$
9.	$A[j+m] \longleftarrow x$
10.	*RETURN(A)*

Feladat. *Rendezzük Shell rendezéssel az alábbi tömböt:*

$$A = [315, 22, 8, 3, 13, 4, 29]$$

legyen továbbá $h := [1, 2, 4]$, *a rendezendő tömb hossza:* $n = 7$.
$\boxed{s{=}3}$ $m = h[3] = 4$ *FOR* $i = 5$ *TO* 7 *DO*

$$i = 5: \quad A = [\ \underset{\uparrow\, j}{\overset{13}{\cancel{315}}},\ 22,\ 8,\ 3,\ \underset{\downarrow\, x}{\overset{315}{\cancel{13}}},\ 4,\ 29\ ] \quad \textit{(WHILE 1x)}$$

$$i = 6: \quad A = [\ 13\ \underset{\uparrow\, j}{\overset{4}{\cancel{22}}},\ 8,\ 3,\ 315,\ \underset{\downarrow\, x}{\overset{22}{\cancel{4}}},\ 29\ ] \quad \textit{(WHILE 1x)}$$

$$i = 7: \quad A = [\ 13\ 4,\ \underset{\uparrow\, j}{8},\ 3,\ 315,\ 22,\ \underset{\downarrow\, x}{\overset{29}{\cancel{29}}}] \quad \textit{(WHILE 0x)}$$

$\boxed{s{=}2}$ $m = h[2] = 2$ *FOR* $i = 3$ *TO* 7 *DO*

$$\begin{array}{lll} i=3: & A=[& \overset{}{\overset{8}{\overset{\uparrow}{\overset{j}{\cancel{13}}}}},\ 4,\ \overset{}{\overset{13}{\overset{\downarrow}{\overset{x}{\cancel{8}}}}},\ 3,\ 315,\ 22,\ 29\ ] \quad \textit{(WHILE 1x)} \\ i=4: & A=[& 8,\ \overset{}{\overset{3}{\overset{\uparrow}{\overset{j}{\cancel{4}}}}},\ 13,\ \overset{}{\overset{4}{\overset{\downarrow}{\overset{x}{\cancel{3}}}}},\ 315,\ 22,\ 29\ ] \quad \textit{(WHILE 1x)} \\ i=5: & A=[& 8,\ 3,\ \overset{}{\overset{13}{\overset{\uparrow}{\overset{j}{\cancel{13}}}}},\ 4,\ \overset{}{\overset{315}{\overset{\downarrow}{\overset{x}{\cancel{315}}}}},\ 22,\ 29\ ] \quad \textit{(WHILE 0x)} \\ i=6: & A=[& 8,\ 3,\ 13,\ \overset{}{\overset{4}{\overset{\uparrow}{\overset{j}{\cancel{4}}}}},\ 315,\ \overset{}{\overset{22}{\overset{\downarrow}{\overset{x}{\cancel{22}}}}},\ 29\ ] \quad \textit{(WHILE 0x)} \\ i=7: & A=[& 8,\ 3,\ 13,\ 4,\ \overset{}{\overset{29}{\overset{\uparrow}{\overset{j}{\cancel{315}}}}},\ 22,\ \overset{}{\overset{315}{\overset{\downarrow}{\overset{x}{\cancel{29}}}}}] \quad \textit{(WHILE 1x)} \end{array}$$

$\boxed{s{=}1}$ $m = h[1] = 1$ *FOR* $i = 2$ *TO* 7 *DO (Innen egy beszúrásos rendezés valósul meg.)*

$$\begin{array}{lll} i=2: & A=[& \overset{}{\overset{3}{\overset{\uparrow}{\overset{j}{\cancel{8}}}}},\ \overset{}{\overset{8}{\overset{\downarrow}{\overset{x}{\cancel{3}}}}},\ 13,\ 4,\ 29,\ 22,\ 315\ ] \quad \textit{(WHILE 1x)} \\ i=3: & A=[& 3,\ \overset{}{\overset{8}{\overset{\uparrow}{\overset{j}{\cancel{8}}}}},\ \overset{}{\overset{13}{\overset{\downarrow}{\overset{x}{\cancel{13}}}}},\ 4,\ 29,\ 22,\ 315\ ] \quad \textit{(WHILE 0x)} \end{array}$$

$$\begin{array}{lll} & & \overset{}{\overset{4}{\overset{\uparrow}{\overset{j}{\cancel{8}}}}}\ \ \overset{}{\overset{8}{\overset{\uparrow}{\overset{j}{\cancel{13}}}}}\ \ \overset{}{\overset{13}{\overset{\downarrow}{\overset{x}{\cancel{4}}}}} \\ i=4: & A=[& 3,\ \ 29,\ \ 22,\ \ 315\ ] \end{array} \quad \begin{array}{l} j=3\ A=[3,8,13,13,29,22,315] \\ j=2\ A=[3,8,8,13,29,22,315] \\ j=1\ \textit{nem indul be} \end{array}$$

$$\begin{array}{lll} i=5: & A=[& 3,\ 4,\ 8,\ \overset{}{\overset{13}{\overset{\uparrow}{\overset{j}{\cancel{13}}}}},\ \overset{}{\overset{29}{\overset{\downarrow}{\overset{x}{\cancel{29}}}}},\ 22,\ 315\ ] \quad \textit{(WHILE 0x)} \end{array}$$

$$
\begin{array}{lllllllllll}
 & & & & & & & 22 & 29 & & \\
i=6: & A=[& 3, & 4, & 8, & 13, & \cancel{29}, & \cancel{22}, & 315\ ] & \textit{(WHILE 0x)} \\
 & & & & & \uparrow & \uparrow & \downarrow & & & \\
 & & & & & j & j & x & & &
\end{array}
$$

$$
\begin{array}{lllllllll}
 & & & & & & & & 315 & \\
i=7: & A=[& 3, & 4, & 8, & 13, & 22, & 29, & \cancel{315}] & \textit{(WHILE 0x)} \\
 & & & & & & & \uparrow & \downarrow & \\
 & & & & & & & j & x &
\end{array}
$$

RETURN $A = [3, 4, 8, 22, 29, 315]$

0.10. Rendezések III.

0.10.1. Az összefésülő rendezés

Az Összefésülő rendezés az ÖSSZEFÉSÜL algoritmuson alapul, amely két rendezett tömböt rendezett tömbbé fésül össze. Ennek az algoritmusnak most két változatát írjuk le. Az első változatban egy A és egy B rendezett tömböt rendezett módon összefésüljük. A kapott tömb a C tömb.

A második változatban ezt az algoritmust átírjuk úgy, hogy alkalmas legyen egy tömb két egymáshoz csatlakozó rendezett résztömbjének rendezett módon történő összefűzésére. A kapott rendezett tömb visszaíródik az eredeti tömb megfelelő helyére.

ÖSSZEFÉSÜL1(A, B, C) algoritmus.
INPUT: $A[1..n]$, $B[1..m]$ a rendezendő tömbök.
OUTPUT: $C[1..n+m]$ rendezett tömb, az A és B tömbök rendezett összefésülése.

	ÖSSZEFÉSÜL1(A, B, C)
1.	$i \longleftarrow 1$, $j \longleftarrow 1$, $k \longleftarrow 1$
2.	REPEAT
3.	IF $A[i] < B[j]$
4.	THEN $C[k] \longleftarrow A[i]$, INC(i), INC(k),
5.	ELSE $C[k] \longleftarrow B[j]$, INC(j), INC(k),
6.	UNTIL $i = n+1$ vagy $j = m+1$
7.	IF $i = n+1$
8.	THEN FOR $l \longleftarrow j$ TO m DO
9.	$C[k] \longleftarrow B[l]$ INC(k)
10.	ELSE FOR $l \longleftarrow i$ TO n DO
11.	$C[k] \longleftarrow A[i]$ INC(k)
12.	RETURN(C)

A fenti algoritmusnak csak az 1.-6. sora érdekes, a 7.-től 11.-ig csak a "szálak elvarrása történik."

Feladat. *Az ÖSSZEFÉSÜL1 algoritmus alkalmazásával fésüljük össze az*

$$A = [3, 5, 11, 14, 17, 22, 25, 33], \qquad \textit{és} \qquad B = [1, 2, 6, 8, 9, 15, 16]$$

tömböket. Az eredmény jelenjen meg egy C tömbben.

```
IF A[i] < B[j]
   THEN C[k] ⟵ A[i] INC(i),INC(k)
   ELSE C[k] ⟵ B[j] INC(j),INC(k)
```

i	j	k	$A[i] < B[j]$	$C[k] \longleftarrow$
1	1	1	$A[1] < B[1]$ *hamis*	$C[1] \longleftarrow B[1] = 1$
1	2	2	$A[1] < B[2]$ *hamis*	$C[2] \longleftarrow B[2] = 2$
1	3	3	$A[1] < B[3]$ *hamis*	$C[3] \longleftarrow A[1] = 3$
2	3	4	$A[2] < B[3]$ *hamis*	$C[4] \longleftarrow A[2] = 5$
3	3	5	$A[3] < B[3]$ *hamis*	$C[5] \longleftarrow B[3] = 6$
3	4	6	$A[3] < B[4]$ *hamis*	$C[6] \longleftarrow B[4] = 8$
3	5	7	$A[3] < B[5]$ *hamis*	$C[7] \longleftarrow B[5] = 9$
3	6	8	$A[3] < B[6]$ *hamis*	$C[8] \longleftarrow A[3] = 11$
4	6	9	$A[4] < B[6]$ *hamis*	$C[9] \longleftarrow A[4] = 14$
5	6	10	$A[5] < B[6]$ *hamis*	$C[10] \longleftarrow B[6] = 15$
5	7	11	$A[5] < B[7]$ *hamis*	$C[11] \longleftarrow B[7] = 16$
5	8	12		

$$C[12] \longleftarrow A[5] = 17$$
$$C[13] \longleftarrow A[6] = 22$$
$$C[14] \longleftarrow A[7] = 25$$
$$C[15] \longleftarrow A[8] = 33$$

A kapott rendezett tömb: $C = [1, 2, 3, 5, 6, 8, 9, 11, 14, 15, 16]$.

ÖSSZEFÉSÜL2 (A, p, q, r) *algoritmus.*
INPUT: $A[1, \ldots, n]$ *tömb, amelynek az* $A[p, \ldots, q]$, $A[q+1, \ldots, r]$ *résztömbjei rendezettek. Ezeket a rendezett résztömböket rendezett módon összefésüljük. Az eredmény átmenetileg bekerül egy* $C[1, \ldots, r - p + 2]$ *tömbbe, ahonnan az elemeket visszaírjuk az* $A[p, \ldots, r]$ *résztömbbe.*
OUTPUT: $C[1, \ldots, n]$ *rendezett tömb, az* $A[p, \ldots, r]$ *rendezett résztömbökkel.*

	ÖSSZEFÉSÜL2(A, p, q, r)
1.	$i \longleftarrow p$, $j \longleftarrow q + 1$, $k \longleftarrow 1$
2.	*REPEAT*
3.	*IF* $A[i] < A[j]$
4.	*THEN* $C[k] \longleftarrow A[i]$, *INC*(i), *INC*(k),
5.	*ELSE* $C[k] \longleftarrow A[j]$, *INC*(j), *INC*(k),
6.	*UNTIL* $i = q + 1$ *vagy* $j = r + 1$
7.	*IF* $i = q + 1$
8.	*THEN FOR* $l \longleftarrow j - 1$ *TO* r *DO*
9.	$C[k] \longleftarrow A[l]$ *INC*(k)
10.	*ELSE FOR* $l \longleftarrow i - 1$ *TO* q *DO*
11.	$C[k] \longleftarrow A[l]$ *INC*(k)
12.	*FOR* $l \longleftarrow 1$ *TO* $r - p + 1$ *DO*
13.	$A[p - 1 + l \longleftarrow C[l]$
14.	*RETURN(A)*

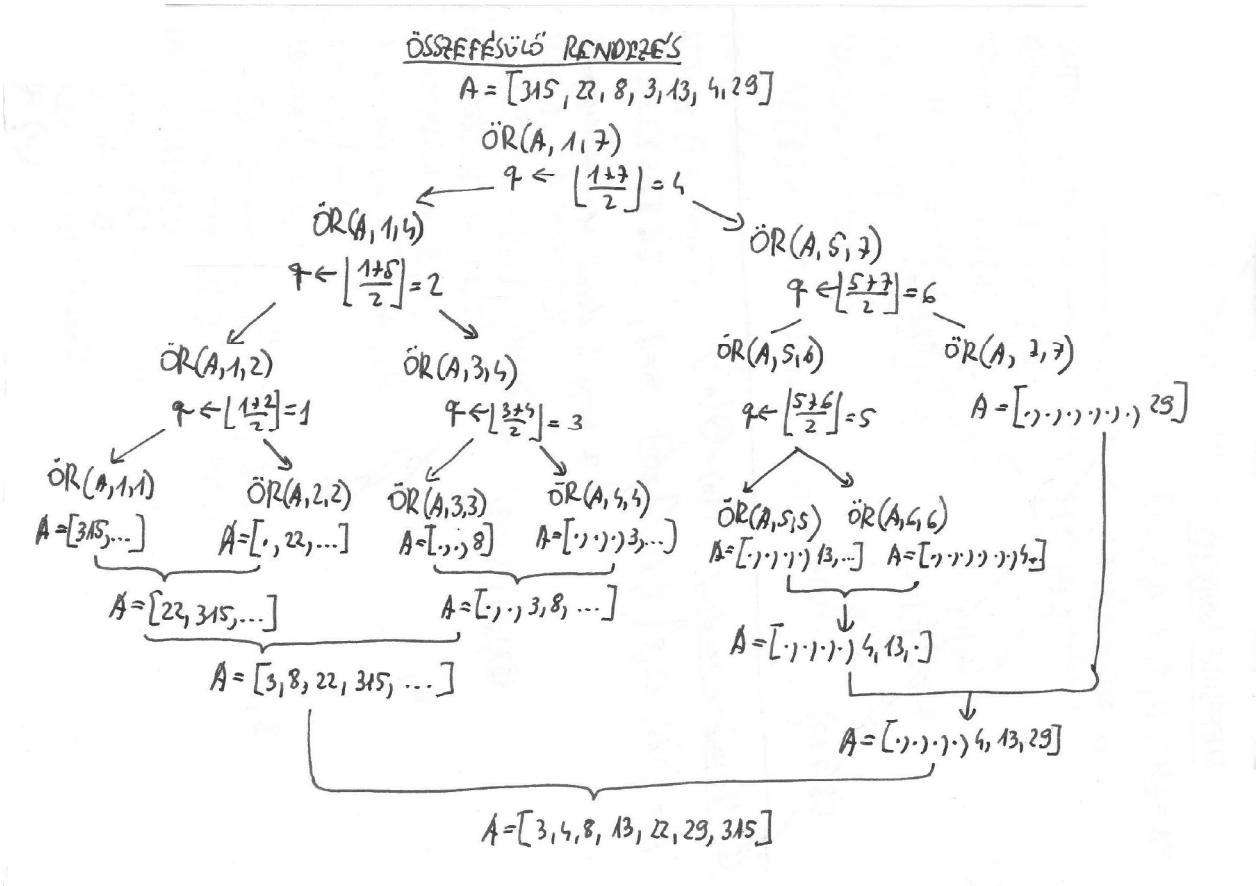
7. Összefésülő rendezés (MERGE SORT)

$T(n) = \Theta(n \log(n))$ $\ddot{O}R(A, p, r)$

INPUT: A egy tömb, amelynek az $A[p, \ldots, r]$ résztömbjét kell rendezni.

OUTPUT: A tömb, a $A[p, \ldots, q]$ rendezett résztömbökkel.

	$\ddot{O}R(A, p, r)$
1.	*IF* $p < r$
2.	*THEN* $q \longleftarrow \boxed{\frac{p+r}{2}}$
3.	$\ddot{O}R(A, p, q)$
4.	$\ddot{O}R(A, q+1, r)$
5.	$\ddot{O}SSZEFÉSÜL2(A, p, q, r)$
6.	*RETURN(A)*



0.10.2. Butcher-féle páros páratlan összefésülés

Butcher-féle páros-páratlan összefűzés

$BATCHER(A, B, C)$

INPUT: A, B rendezett tömbök.

OUTPUT: C az A, B rendezett tömbök rendezett összefűzése.

Az algoritmus úgy működik, hogy az A tömb elemeiből létrehozzuk az A_1 és A_2 tömböket úgy, hogy az A tömb páratlan indexű elemei az A_1, páros indexű elemei az A_2 tömbbe kerülnek. Hasonló módon előállítjuk a B tömb elemeiből a B_1 és B_2 tömböket.

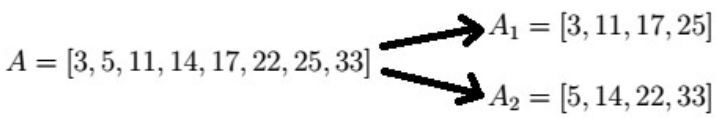
Ezután az U tömbbe rendezett módon összefésüljük a A_1 és B_2 rendezett tömböket, illetve a V tömbbe rendezett módon összefésüljük az A_2 és B_1 rendezett tömböket.

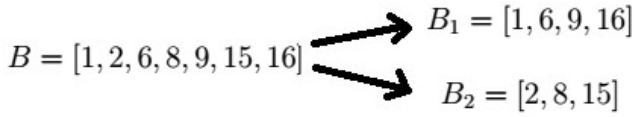
Ezután jön a trükk: A C tömbbe kell rendezett módon összefésülnünk az U és a V tömböket. Ez az összefésülés azonban már könnyen megtehető az alábbi módon:

$$C[2i-1] := \min(U[i], V[i]),$$
$$C[2i] := \max(U[i], V[i]).$$

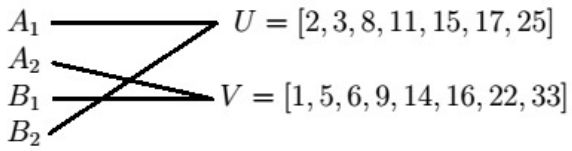
	$BATCHER(A, B, C)$
1.	$i \longleftarrow 1$, $k \longleftarrow 1$
2.	*WHILE* $(i \leqslant \text{Hossz}[A])$
3.	$\quad A1[k] \longleftarrow A[i]$, $INC(k)$, $i \longleftarrow i+2$
4.	$i \longleftarrow 2$, $k \longleftarrow 1$
5.	*WHILE* $(i \leqslant \text{Hossz}[A])$
6.	$\quad A2[k] \longleftarrow A[i]$, $INC(k)$, $i \longleftarrow i+2$
7.	$i \longleftarrow 1$, $k \longleftarrow 1$
8.	*WHILE* $(i \leqslant \text{Hossz}[B])$
9.	$\quad B1[k] \longleftarrow B[i]$, $INC(k)$, $i \longleftarrow i+2$
10.	$i \longleftarrow 2$, $k \longleftarrow 1$
11.	*WHILE* $(i \leqslant \text{Hossz}[B])$
12.	$\quad B2[k] \longleftarrow B[i]$, $INC(k)$, $i \longleftarrow i+2$
13.	$ÖSSZEFÉSÜL1(A1, B2, U)$
14.	$ÖSSZEFÉSÜL1(A2, B1, V)$
15.	$n \longleftarrow \min(\text{Hossz}[U], \text{Hossz}[V])$
16.	$i \longleftarrow 1$, $k \longleftarrow 1$
17.	*WHILE* $(i \leqslant n)$
18.	$\quad C[k] \longleftarrow \min(U[i], V[i])$, $INC(k)$
19.	$\quad C[k] \longleftarrow \max(U[i], V[i])$, $INC(k)$
20.	*IF* $n < \text{Hossz}[U]$
21.	$\quad$ *THEN* $C[k] \longleftarrow U[n+1]$
22.	*IF* $n < \text{Hossz}[V]$
23.	$\quad$ *THEN* $C[k] \longleftarrow V[n+1]$
24.	*RETURN(C)*

Feladat a Batcher-féle páros-páratlan összefésülésre





$$\ddot{O}SSZEFÉSÜL1(A1, B2, U),$$
$$\ddot{O}SSZEFÉSÜL1(A2, B1, V).$$



$$C[2i-1] = \min(U[i], V[i]),$$
$$C[2i] = \max(U[i], V[i]).$$

U	2	3	8	11	15	17	25	
V	1	5	6	9	14	16	22	33

Tehát a rendezett tömb: $C = [1, 2, 3, 5, 6, 8, 9, 11, 14, 15, 16, 17, 22, 25, 33]$.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.10.3. A Négyzetes rendezés

8. Négyzetes rendezés

$T(n) = \Theta\left(n^{\frac{3}{2}}\right)$. *A tömb elemeit alcsoportokra osztjuk úgy, hogy minden alcsoportnak körülbelül annyi eleme legyen, amennyi az alcsoportok száma. Ez úgy érhető el, hogy* n *hosszúságú tömb esetén a* $\lceil\sqrt{n}\rceil$ *elemenként alkossanak a tömb elemei egy-egy alcsoportot.*

Ezt követően minden alcsoportnak kiválasztjuk a legkisebb elemét. Ezeket az elemeket egy segédtömbben átmenetileg tároljuk, ezek alkotják a főcsoportot. A kiválasztott minimális elemeket el is távolítjuk az eredeti tömbből.

Ezt követően megkeressük a főcsoport legkisebb elemét, amelyet eltávolítunk a főcsoportból és beírjuk az eredménytömbbe. Az eltávolított elemet abból az alcsoportból pótoljuk, amelyikből az eltávolított elem származik. Az alcsoport sorszámát (tehát, hogy hányadik alcsoportról van szó), a főcsoportból eltávolított elem indexe adja. Illetve, ha az alcsoportnak már nincs eleme, akkor nem pótoljuk. Az eljárást iteráltan alkalmazzuk mindaddig, ameddig a főcsoportból el nem fogynak az elemek.

Az eredménytömb az eredeti tömb elemeit tartalmazza rendezett módon.

Rendezzük a négyzetes rendezés alkalmazásával a

$$A = [315, 22, 8, 3, 13, 4, 29]$$

tömböt. Mivel $n = 7$ *és* $\lceil \sqrt{n} \rceil = 3$*, így 3 elemenként fognak alcsoportotokat alkotni az* A*. tömb elemei és 3 alcsoportot kapunk. Ezek a következők:*

$$A = [315, 22, 8|3, 13, 4|29].$$

Az alcsoportok minimális elemeiből megalkotjuk a főcsoportot. A főcsoportnak annyi eleme van, amennyi az alcsoportok száma, tehát szintén 3.

A főcsoportból kiválasztjuk és el is távolítjuk a minimális elemet. Ez az elem bekerül az eredménytömbbe (ami most egy B *tömb), majd a megfelelő alcsoportból visszapótoljuk, amennyiben a megfelelő alcsoportnak van eleme, ha nincs, akkor nem pótoljuk. Ezt az iterációt tartalmazza a következő táblázat.*

8	~~3~~	29	$A = [315, 22, \cdot\|\cdot, 13, \sout{4}\|\cdot]$
8	4	29	$A = [315, 22, \cdot\|\cdot, \sout{13}, \cdot\|\cdot]$
~~8~~	13	29	$A = [315, \sout{22}, \cdot\|\cdot, \cdot, \cdot\|\cdot]$
22	~~13~~	29	$A = [315, \cdot, \cdot\|\cdot, \cdot, \cdot\|\cdot]$
~~22~~	$\cdot$	29	$A = [\sout{315}, \cdot, \cdot\|\cdot, \cdot, \cdot\|\cdot]$
315	$\cdot$	~~29~~	$A = [\cdot, \cdot, \cdot\|\cdot, \cdot, \cdot\|\cdot]$
~~315~~	$\cdot$	$\cdot$	
$\cdot$	$\cdot$	$\cdot$	

Tehát a rendezett tömb: $B = [3, 4, 8, 13, 22, 29, 315]$.

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.10.4. A lexikografikus rendezés

Lexikografikus rendezés: de előtte elevenítsük fel a rendezés fogalmát.

Ismétlés *Legyen* X *egy halmaz,* $\leqslant$ *egy reláció* X*-en. Azt mondjuk, hogy a* $\leqslant$ *reláció egy* ***rendezés*** X*-en, ha*

1. ***Reflexív****, azaz* $x \leqslant x$ *minden* $x \in X$ *esetén.*
2. ***Antiszimmetrikus****, azaz ha* $x \leqslant y$ *és* $y \leqslant x$*, akkor* $x = y$ *minden* $x, y \in X$ *esetén.*
3. ***Tranzitív****, azaz ha* $x \leqslant y$ *és* $y \leqslant z$*, akkor* $x \leqslant z$ *minden* $x, y, z \in X$ *esetén.*
4. ***Dichotóm****, azaz* $x \leqslant y$ *vagy* $y \leqslant x$ *minden* $x, y \in X$ *esetén.*

Most megismerkedünk a lexikografikus rendezés fogalmával.

Legyen Σ *egy véges ábécé, amelyen értelmezve van egy* $\leqslant$ *rendezési reláció. Jelölje* Σ^n *a* Σ *ábécé elemeiből alkotott* n *hosszúságú stringek halmazát.*

Sztringek egyenlősége*: Legyenek* $\alpha, \beta \in \Sigma^n$ $\alpha = a_1, \ldots, a_n$; $\beta = b_1, \ldots, b_n$.

$$\alpha = \beta \qquad :\Longleftrightarrow \qquad a_i = b_i \quad (i = 1, 2, \ldots, n).$$

Lexikografikus rendezés Σ^n*-n: Legyenek* $\alpha, \beta \in \Sigma^n$

$$\alpha \leqslant \beta \qquad :\Longleftrightarrow \qquad \alpha = \beta \quad \text{vagy} \quad \exists k \in \{1, \ldots, n\} : a_1 = b_1, \ldots, a_{k-1} = b_{k-1} \quad \text{de} \quad a_k < b_k,$$

ahol az $a_k < b_k$ *azt jelenti, hogy* $a_k \leqslant b_k$*, de* $a_k \neq b_k$*.*

Eddig megismerkedtünk a lineáris rendezés fogalmával. A lineáris rendezésnek van egy szigorú változata, amely az alábbi módon axiomatizálható:

Szigorú rendezés. *Legyen* X *egy halmaz,* $<$ *egy reláció* X*-en. Azt mondjuk, hogy* $<$ *egy szigorú rendezés az* X*-en, ha*

1. ***Irreflexív****, azaz* $x \not< x$ *minden* $x \in X$ *esetén (azaz nincs olyan* x *eleme az* X *halmaznak, amelyre* $x < x$ *teljesül). Ez azt jelenti, hogy egy elem sem nagyobb saját magánál.*
2. ***Tranzitív****, azaz ha* $x < y$ *és* $y < z$*, akkor* $x < z$ *minden* $x, y, z \in X$ *esetén.*
3. ***Tricichotóm****, azaz vagy* $x < y$*, vagy* $x = y$*, vagy* $y < x$ *minden* $x, y \in X$ *esetén. A trichotómia most is azt garantálja, hogy bármely két (nem feltétlenül különböző) elem összehasonlítható legyen.*

A következő két tétel azt mutatja, hogy rendezésből mindig elő tudunk állítani szigorú rendezést, illetve szigorú rendezésből rendezést.

1.Tétel. *Legyen* X *egy halmaz és* $\leqslant$ *egy rendezés az* X*-en. Definiáljuk a* $<$ *relációt az alábbi módon:*

$$x < y \qquad :\Longleftrightarrow \qquad x \leqslant y \quad \text{de} \quad x \neq y \qquad (x, y \in X).$$

Ekkor a $<$ *egy szigorú rendezés az* X *halmazon.*

2.Tétel. *Legyen* X *egy halmaz és* $<$ *egy szigorú rendezés az* X*-en. Definiáljuk a qleqslant relációt az alábbi módon:*

$$x \leqslant y \qquad :\Longleftrightarrow \qquad x < y \quad \text{vagy} \quad x = y \qquad (x, y \in X).$$

Ekkor a $\leqslant$ *egy rendezés az* X *halmazon.*

A fenti két tétel bizonyítása **szorgalmi házi feladat**.

Ezeknek az egyszerű tényeknek az ismeretében a lexikografikus rendezés úgy érthető meg, hogy először az 1. tételünk segítségével létrehozunk egy szigorú rendezést a Σ ábécén. Ennek a szigorú rendezésnek a segítségével szigorú rendezést értelmezünk az n hosszúságú sztringek Σ^n halmazán az alábbi módon:

$$\alpha < \beta \quad :\Longleftrightarrow \quad a_1 < b_1 \text{ vagy } \exists k \in \{1, \ldots, n\} : a_1 = b_1, \ldots, a_{k-1} = b_{k-1} \text{ de } a_k < b_k,$$

azaz $\alpha < \beta$, pontosan akkor, ha balról jobbra haladva a sztringekben először α-ban találunk nagyobb betűt. Így a n hosszúságú sztringek halmazán kapunk egy szigorú rendezést. Ebből a szigorú rendezésből a 2. tételünk segítségével állítjuk elő a lexikografikus rendezést.

Feladat. *A Cormen-könyvben (vagy legalábbis annak magyar fordításában) szerepel az alábbi feladat: Rendezzük a lexikografikus rendezéssel az alábbi szavakat: ÓRA, LAP, TEA, GÉP, ING, SZÓ, FÜL, ORR, LÁB, SOR, KÉS, ÁGY, VÍZ, RÁK, CÉL, FIÚ.*

0.10.5. A Radix, vagy számjegyes rendezés

9. Radix rendezés *: Számjegyes rendezés.*

Azonos hosszúságú szavak rendezésére alkalmas. (Azonos hosszúságú szavak helyett gondolhatunk ugyanannyi jegyből álló pozitív egészekre.) Ha azonos hosszúságú szavakat rendezünk, akkor lexikografikus rendezést valósítunk meg. A radix rendezés felhasznál egy stabil rendezést. A szavakat a betűik alapján rendezzük jobbról balra haladva (tehát az utolsó karaktertől az első felé). Mire elérjük a legelső, tehát a bal szélső betűt, akkor a szavak rendezetté válnak.

A következő numerikus példában az

$$235, 144, 474, 508, 133, 175, 200, 741$$

számokat rendezzük a radix rendezés segítségével.

$$\begin{array}{ccc|c|ccc|c|ccc|c|ccc}
2&3&5&&2&0&0&&2&0&0&&1&3&3\\
1&4&4&&7&4&1&&5&0&8&&1&4&4\\
4&7&4&&1&3&3&&1&3&3&&1&7&5\\
5&0&8&\rightsquigarrow&1&4&4&\rightsquigarrow&2&3&5&\rightsquigarrow&2&0&0\\
1&3&3&&4&7&4&&7&4&1&&2&3&5\\
1&7&5&&2&3&5&&1&4&4&&4&7&4\\
2&0&0&&1&7&5&&4&7&4&&5&0&8\\
7&4&1&&5&0&8&&1&7&5&&7&4&1
\end{array}$$

△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O△□O

0.10.6. Az edényrendezés

Korábban beszéltünk arról, hogy egy matematikai értelemben sűrű számhalmaz (például a racionális számok halmaza) elemei leszámláló rendezéssel nem rendezhetők. Az edényrendezés egy olyan rendezés, amely például racionális számokból álló sorozatot is képes rendezni lineáris időben. Ennek is ára van, a rendezendő sorozatnak egyenletes eloszlásúnak kell lenniük valamely $[a, b[$ intervallumon.

Azt mondjuk, hogy egy $x_1, \ldots, x_n$ egy n-tagú adatsorozat egyenletes eloszlású egy $[a, b[$ intervallumon azt jelenti, hogy a részintervallumba esés valószínűsége arányos a részintervallum hosszával.

Mivel a valószínűség fogalma leginkább a relatív gyakoriság segítségével fogható meg, a fenti definíció az alábbi módon önthető matematikai formába:

$$\frac{k_n}{n} := \frac{\{i \in \{1, \ldots, n\} \mid x_i \in [c, d]\}}{n} \quad \sim \quad \frac{d-c}{b-a} \qquad (a \leqslant c < d \leqslant b).$$

Ha az egyenletes eloszlásra vonatkozó feltétel nem teljesül, akkor is működik a rendezési algoritmus, csak ebben az esetben a futási idő nem lesz lineáris.

Mivel tetszőleges $[a, b[$ intervallum egy egyszerű affin transzformációval beletranszformálható a $[0, 1[$-be, ezért most feltételezzük, hogy a sorozat tagjai eleve a $[0, 1[$ intervallumból származnak.

Az edényrendezés arról kapta a nevét, hogy a $[0, 1[$ intervallumot n darab egyenlő részintervallumra bontjuk, ezek lesznek az edények. Tehát az edények az

$$\left[0, \frac{1}{n}\right[, \quad \left[\frac{1}{n}, \frac{2}{n}\right[, \quad \ldots, \quad \left[\frac{n-1}{n}, 1\right[$$

intervallumok. Van továbbá egy n elemű tömbünk, amelyben láncolt listák fejei vannak. Használunk egy ügyes függvényt, amely egy $[0, 1[$-beli elemről kapásból megmondja, hogy melyek edénybe tartozik. Ez a függvény:

$$x_i \mapsto \lfloor n \cdot x_i \rfloor .$$

Az x_i adatelemet a kapott láncolt listához hozzáfűzzük. Ezt követően a n darab láncolt listát rendezzük, majd egyetlen listává fűzzük össze. Az algoritmus azzal a láncolt listával tér vissza, amely a eredeti tömb elemeit rendezett módon tartalmazza.

10. Edény rendezés *:*

EDENYREND(A, L)

INPUT: A egy tömb, amely a rendezendő elemeket tartalmazza.

OUTPUT: L egy láncolt lista, amely az A elemeit tartalmazza növekvő sorrendben.

$B[0, \ldots, n-1]$ *a listafejeket tartalmazó tömb.*

	EDENYREND(A, L)
1.	*B-be beszúr.*
2.	*B listáit rendezi.*
3.	*B listáit összefűzi L be.*
24.	*RETURN(L)*

Feladat. *Edényrendezés segítségével rendezzük a következő tömböt:*

$$A = [0.741, \quad 0.522, \quad 0.579, \quad 0.251, \quad 0.180, \quad 0.446, \quad 0.619, \quad 0.107, \quad 0.827]$$

Legyen $n = 4$*, azaz* $n = 4$ *db edény van.*

$$\begin{array}{cccc} 0 & 1 & 2 & 3 \\ \left[0, \frac{1}{4}\right[, & \left[\frac{1}{4}, \frac{2}{4}\right[, & \left[\frac{2}{4}, \frac{3}{4}\right[, & \left[\frac{3}{4}, \frac{4}{4}\right[\end{array}$$

$A[j]$ *elemet láncoljuk hozzá a* $B\left[\lfloor nA[j] \rfloor\right]$ *fejű listához.*

$$\begin{array}{lcl} 0,741 & \mapsto & \lfloor 4 \cdot 0.471 \rfloor = 2 \\ 0,522 & \mapsto & \lfloor 4 \cdot 0.522 \rfloor = 2 \\ 0,579 & \mapsto & \lfloor 4 \cdot 0.579 \rfloor = 2 \\ 0,251 & \mapsto & \lfloor 4 \cdot 0.251 \rfloor = 1 \\ 0,180 & \mapsto & \lfloor 4 \cdot 0.180 \rfloor = 0 \\ 0,446 & \mapsto & \lfloor 4 \cdot 0.446 \rfloor = 1 \\ 0,619 & \mapsto & \lfloor 4 \cdot 0.619 \rfloor = 2 \\ 0,107 & \mapsto & \lfloor 4 \cdot 0.107 \rfloor = 0 \\ 0,827 & \mapsto & \lfloor 4 \cdot 0.827 \rfloor = 3 \end{array}$$

Így az alábbi láncolt listákat kapjuk:

$B[0]$	$\left(x \in \left[0, \frac{1}{4}\right[\right)$	*NIL*	$\rightarrow$	$0,180$	$\rightarrow$	$0,107$		
$B[1]$	$\left(x \in \left[\frac{1}{4}, \frac{2}{4}\right[\right)$	~~*NIL*~~	$\rightarrow$	$0,251$	$\rightarrow$	$0,446$		
$B[2]$	$\left(x \in \left[\frac{2}{4}, \frac{3}{4}\right[\right)$	~~*NIL*~~	$\rightarrow$	$0,741$	$\rightarrow$	$0,522$	$\rightarrow$ $0,579$	$\rightarrow$ $0,619$
$B[3]$	$\left(x \in \left[\frac{3}{4}, \frac{4}{4}\right[\right)$	*NIL*	$\rightarrow$	$0,827$				

Rendezzük a kapott láncolt listákat.

$B[0]$	$\left(x \in \left[0, \frac{1}{4}\right[\right)$	*NIL*	$\rightarrow$	$0,107$	$\rightarrow$	$0,180$		
$B[1]$	$\left(x \in \left[\frac{1}{4}, \frac{2}{4}\right[\right)$	*NIL*	$\rightarrow$	$0,251$	$\rightarrow$	$0,446$		
$B[2]$	$\left(x \in \left[\frac{2}{4}, \frac{3}{4}\right[\right)$	*NIL*	$\rightarrow$	$0,522$	$\rightarrow$	$0,579$	$\rightarrow$ $0,619$	$\rightarrow$ $0,741$
$B[3]$	$\left(x \in \left[\frac{3}{4}, \frac{4}{4}\right[\right)$	*NIL*	$\rightarrow$	$0,827$				

Majd a rendezett láncolt listákat egyetlen listává fűzzük össze.

$\boxed{L} \rightarrow \boxed{0.107} \rightarrow \boxed{0.180} \rightarrow \boxed{0.251} \rightarrow \boxed{0.446} \rightarrow \boxed{0.522} \rightarrow \boxed{0.579} \rightarrow \boxed{0.619} \rightarrow \boxed{0.741} \rightarrow \boxed{0.827}$

0.11. A Huffman-kód

Fix hosszúságú kódok

Szeretnénk kódolni az ABRAKADABRA szót. Az első lehetőség az, hogy minden karakterhez hozzárendelünk egy fix hosszúságú kódot. Ha k jelöli a kód hosszát, akkor teljesülnie kell a

$$2^{k-1} < 5 \leqslant 2^k \qquad \Longrightarrow \qquad k = \lceil \log(5) \rceil = 3$$

egyenlőtlenségnek. Így kapjuk, hogy 3 hosszúságú kódot kell használnunk.

$$\left.\begin{array}{ll} a & 000 \\ b & 001 \\ r & 010 \\ k & 011 \\ d & 100 \end{array}\right\} \quad \textit{kódok}$$

Kódolás: 000 001 010 000 011 000 100 000 001 010 000

Ez egy 33 bites kódszó. Természetesen nincsenek szóközök, ez csak a könnyebb áttekinthetőség miatt írtam. A dekódolás rendkívül egyszerű. A kódszót 3 bites részekre kell bontanunk, és könnyen visszakapjuk a kódolt szöveget.

$$\underbrace{000}_{a}\,|\,\underbrace{001}_{b}\,|\,\underbrace{010}_{r}\,|\,\underbrace{000}_{a}\,|\,\underbrace{011}_{k}\,|\,\underbrace{000}_{a}\,|\,\underbrace{100}_{d}\,|\,\underbrace{000}_{a}\,|\,\underbrace{001}_{b}\,|\,\underbrace{010}_{r}\,|\,\underbrace{000}_{a}$$

Ezen a gyakorlaton a kódolásra egy ennél gazdaságosabb módszert ismerünk meg. Ez a Huffman-kód.

0.11.1. A Huffman kód

A Huffman-kód elkészítéséhez először is ismernünk kell a karakterek gyakoriságait (illetve relatív gyakoriságait). Ezek az értékek függnek a kiválasztott szövegtől, a szerző szóhasználatától, ... tehát sok mindentől azonban tegyük fel, hogy a birtokunkban van egy ilyen statisztika. Ezt követően felépítünk egy bináris fát. Ennek a fának a csúcsaiban számok állnak, a levelek a karakterek, az élekhez a 0 vagy az 1 értékek valamelyike van rendelve (a bal gyerekbe mutató élhez 0-át, a jobb gyerekbe mutató élhez az 1-et rebdeljük). Adott karakter kódját a gyökértől a karakterig vezető út élehihez rendelt bitsorozat adja.

A faépítés kódolás és dekódolás lépései:

- **Elindulás:** A gyökér legyen a két legkisebb gyakoriságú karakter gyakoriságának az összege. A levelek legyenek a két legkisebb gyakoriságú karakter.
- **Továbbhaladás:**
  - Már vannak részfáink illetve karaktereink gyakoriságokkal. Ezeket nagyság szerint rendezzük a részfák gyökerében szereplő szám illetve a karakterek gyakoriságai alapján.

  - A kapott sorozat két legkisebb eleméből bináris fát építünk, a kapott gráf gyökerébe a két szám (relatív gyakoriság, vagy gyökérben álló szám) összege kerül.

- **Leállás:** A rendezések és a faépítések sorozatát addig folytatjuk, míg az összes karakter illetve részgráf elfogy és egyetlen bináris fát alkotnak.

- **Kiértékelés:** A kapott fagráf éleihez 0-kat és 1-ket rendelünk. A bal oldali gyermekre mutató élhez 0-t, a jobb oldalihoz 1-t rendelünk. A kapott fagráf levelei a karakterek. Minden karakterhez hozzárendelünk egy kódot. Ezt a kódot a gyökértől az adott levélig vezető élsorozat bitjei szolgáltatják.

- **Dekódolás:** A bináris fa ismeretében egyértelmű.

A bináris fa létrehozásánál mindig építünk, aztán rendezünk, aztán megint építünk, megint rendezünk, és így tovább. Gyakori hiba, hogy az építés és rendezésnek ezt a váltakozását nem tartják be. Mi sem írjuk ki mindig, ha a rendezettség nem borul fel.

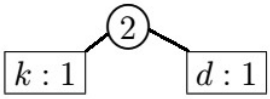
Érdemes még megjegyezni, hogy a Huffman-kód nem egyértelmű. Például, ha az azonos gyakoriságú kódokat permutáljuk, akkor különböző Huffman kódokat kapunk, azonban ezek a Huffman kódok ugyanarra a szóra azonos hosszúságú kódszavakat eredményeznek.

Bár nem életszerű, de a módszert az ABRAKADABRA szó kódolásával illusztráljuk. A gyakoriságokat nyilván nem egy szóból, hanem valamilyen hosszú szövegből kell származtatni, vagy eleve adott.

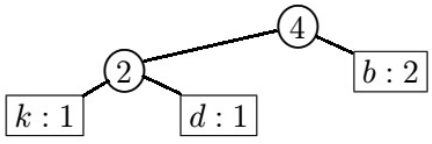
Változó hosszú kódok

$$\begin{array}{ll} a & 5\ db \\ b & 2\ db \\ r & 2\ db \\ k & 1\ db \\ d & 1\ db \end{array}$$

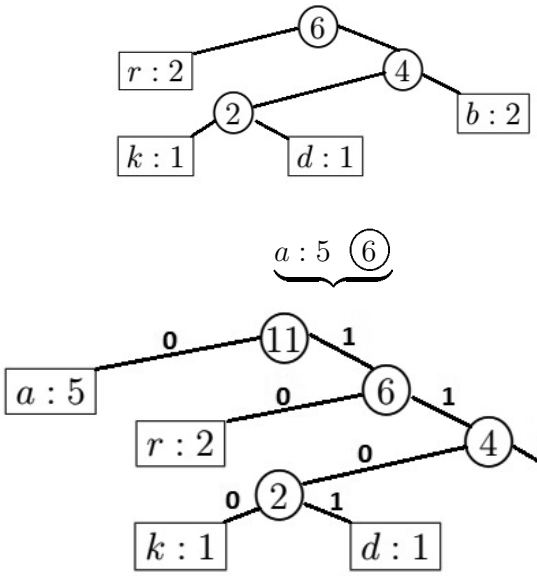
$\underbrace{k:1 \quad d:1}$ $\quad b:2 \quad r:2 \quad a:5$



$\underbrace{(2) \quad b:2}$ $\quad r:2 \quad a:5$



$r:2 \quad \underbrace{(4)}$ $\quad a:5$



A kapott fagráfot ***kódfá****nak nevezzük. A kódfából kiolvashatók a kódok:*

a	0
b	111
r	10
k	1100
d	1101

kódok

Kódolás*:* 0 111 10 0 1100 0 1101 0 111 10 0.
Dekódolás*: A kapott kód prefix kód, tehát mindig egyértelműen dekódolható.*

$$\underbrace{0}_{a}\,|\,\underbrace{111}_{b}\,|\,\underbrace{10}_{r}\,|\,\underbrace{0}_{a}\,|\,\underbrace{1100}_{k}\,|\,\underbrace{0}_{a}\,|\,\underbrace{1101}_{d}\,|\,\underbrace{0}_{a}\,|\,\underbrace{111}_{b}\,|\,\underbrace{10}_{r}\,|\,\underbrace{0}_{a}$$

Látható, hogy rövidebb (23 bit) kódot kaptunk, mint a fix hosszúságú kódolásnál (33 bit).

A fa költsége optimális.