Bevezetés

1. Definíció. *Az alsó egészrész függvény minden valós számhoz egy egész számot rendel hozzá, éppen azt, amely a tőle nem nagyobb egészek közül a legnagyobb. Az alsó egészrész függvény jele:* $\lfloor x \rfloor$*, ahol* x *valós szám. Tömören:*

$$\lfloor x \rfloor = \max_{\substack{k \in \mathbb{Z} \\ k \leq x}} k$$

Más szavakkal formálisan: $\lfloor x \rfloor = k$*, ahol* k *olyan egész szám, hogy* $k \leq x < k+1$.

2. Definíció. *A felső egészrész függvény minden valós számhoz egy egész számot rendel hozzá, éppen azt, amely a tőle nem kisebb egészek közül a legkisebb. A felső egészrész függvény jele:* $\lceil x \rceil$*, ahol* x *valós szám. Tömören:*

$$\lceil x \rceil = \min_{\substack{k \in \mathbb{Z} \\ k \geq x}} k$$

Más szavakkal formálisan: $\lceil x \rceil = k$*, ahol* k *olyan egész szám, hogy* $k - 1 < x \leq k$.

3. Definíció. *A kerekítő függvény minden valós számhoz a hozzá legközelebb eső egész számot rendeli hozzá. Ha a legközelebbi egész szám nem egyértelmű, akkor a nagyobbat választja. A kerekítő függvény jele:* $\mathrm{Round}(x)$*, ahol x valós szám.*

$$\mathrm{Round}(x) = \left\lfloor x + \frac{1}{2} \right\rfloor$$

4. Definíció. *A törtrész függvény minden valós számhoz azt a számot rendeli hozzá, amely azt mutatja meg, hogy a szám mennyivel nagyobb az alsó egészrészénél. A törtrész függvény jele:* $\{x\}$ *, ahol* x *valós szám. Tömören:*

$$\{x\} = x - \lfloor x \rfloor$$

5. Definíció. *Legyen* a *és* b *egész szám,* $b \neq 0$*. Definíció szerint az egész osztás műveletén* (div) *az* a/b *osztás eredményének alsó egész részét értjük. Tömören:*

$$a \mathrm{div} b = \left\lfloor \frac{a}{b} \right\rfloor .$$

6. Definíció. *Legyen* a *és* b *egész szám. Definíció szerint az egész maradék képzését* (mod)*, az alábbi formulával definiáljuk:*

$$a \mathrm{mod} b = \begin{cases} a, & \text{ha } b = 0 \\ a - b \cdot \lfloor a/b \rfloor = a - b \cdot (a \mathrm{div} b), & \text{ha } b \neq 0 \end{cases}$$

7. Definíció. *Az absztrakt adat valamely halmaznak az eleme. Ezen halmaz bármely elemét felhasználhatjuk a munkánkban, számításainkban, az alkalmazott valóságmodellben, objektumainak leírásában, megadásában.*

8. Definíció. *Az absztrakt adattípus egy leírás, amely absztrakt adatok halmazát és a rajtuk végezhető műveleteket adja meg (definiálja) nem törődve azok konkrét (gépi) realizálásával.*

9. Definíció. *Az algoritmus egy meghatározott számítási eljárás, a számítási probléma megoldási eszköze.*

10. Definíció. (*Az algoritmus inputjának a mérete (problémaméret)) Legyen adott egy probléma, amely megoldható egy* A *algoritmussal. Legyen* D *az* A *algoritmus lehetséges inputjainak a halmaza. Legyen* $x \in D$ *egy input. Az* x *input méretének nevezzük az* x *konkrét megadásakor használt bitek számát. Ez egy nemnegatív egész szám, mérőszám. Jelölésben az*x *input mérete* $|x|$.

11. Definíció. . Legyen A egy algoritmus, D az algoritmus összes lehetséges input adatainak a halmaza és x egy lehetséges input. Az x input esetén $t_A(x)$-szel fogjuk jelölni az A algoritmus probléma megoldási időigényét (t-time, idő) és $s_A(x)$-szel a tárigényét (s-storage, tár).

12. Definíció. (*Az algoritmus időbonyolultsága) A*

$$T_A(x) = \sup_{\substack{x \in D \\ |x| \leq n}} t_A(x)$$

számot az A algoritmus időbonyolultságának nevezzük.

Az időbonyolultság megadja, hogy az n-nél nem nagyobb méretű inputok esetén mennyi a legnagyobb időigény.

13. Definíció. (*Az algoritmus tárkapacitás bonyolultsága) Az*

$$S_A(x) = \sup_{\substack{x \in D \\ |x| \leq n}} s_A(x)$$

számot az A algoritmus tárkapacitás bonyolultságának nevezzük.

14. Definíció. (*Az ordo szimbolika szimbólumai) Azt mondjuk, hogy az* $f(n)$ *függvény növekedési rendje:*

nagy ordo g(n), ha létezik olyan pozitív c konstans és pozitív n_0 probléma küszöbméret, hogy ha n a probléma mérete egyenlő a küszöbmérettel, vagy annál nagyobb, akkor az $f(n)$ függvényérték nemnegatív és a $g(n)$ függvényérték c konstansszorosától nem nagyobb. Tömören:

$f(n) = O(g(n))$, ha létezik $c > 0$ és $n_0 > 0$, hogy

minden $n \geq n_0$ esetén $0 \leq f(n) \leq cg(n)$.

nagy omega g(n), ha létezik olyan pozitív c konstans és pozitív n_0 probléma küszöbméret, hogy ha n a probléma mérete egyenlő a küszöbmérettel, vagy annál nagyobb, akkor az $f(n)$ függvényérték legalább akkora, mint a nemnegatív $g(n)$ függvényérték c konstansszorosa. Tömören:

$f(n) = \Omega(g(n))$, ha létezik $c > 0$ és $n_0 > 0$, hogy

minden $n \geq n_0$ esetén $0 \leq cg(n) \leq f(n)$.

nagy teta g(n), ha léteznek olyan pozitív c_1 és c_2 konstansok és pozitív n_0 probléma küszöbméret, hogy ha n a probléma mérete egyenlő a küszöbmérettel, vagy annál nagyobb, akkor az $f(n)$ függvényérték a nemnegatív $g(n)$ függvényérték c_1 és c_2-szerese által meghatározott zárt intervallumból nem lép ki. Tömören:

$f(n) = \Theta(g(n))$, ha létezik $c_1, c_2 > 0$ és $n_0 > 0$,hogy

minden $n \geq n_0$ esetén $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$.

kis ordo g(n), ha minden pozitív c konstanshoz létezik olyan pozitív n_0 probléma küszöbméret, hogy ha n a probléma mérete egyenlő a küszöbmérettel, vagy annál nagyobb, akkor az $f(n)$ függvényérték nemnegatív és a $g(n)$ függvényérték c konstansszorosától kisebb. Tömören:

$f(n) = o(g(n))$, ha minden $c > 0$-hoz létezik egy $n_0 > 0$, hogy

minden $n \geq n_0$ esetén $0 \leq f(n) \leq cg(n)$.

kis omega g(n), ha minden pozitív c konstanshoz létezik olyan pozitív n_0 probléma küszöbméret, hogy ha n a probléma mérete egyenlő a küszöbmérettel, vagy annál nagyobb, akkor az $f(n)$ függvényérték nagyobb, mint a nemnegatív $g(n)$ függvényérték c konstansszorosa. Tömören:

$f(n) = \omega(g(n))$, ha minden $c > 0$-hoz létezik egy $n_0 > 0$, hogy

minden $n \geq n_0$ esetén $0 \leq cg(n) \leq f(n)$.

15. Definíció: *Leggyakoribb növekedési rendek:*

Konstans	$f(n) = \Theta(1)$
Lineáris	$f(n) = \Theta(n)$
Négyzetes	$f(n) = \Theta(n^2)$
Köbös	$f(n) = \Theta(n^3)$
Polinomiális	$f(n) = \Theta(n^k) \qquad k \in \mathbb{R}$ és $k > 0$
Logaritmikus	$f(n) = \Theta(\log(n))$
Exponenciális	$f(n) = \Theta(a^n) \qquad a > 1$

16. Definíció. *A polinomiálisan gyorsabb növekedés*

Azt mondjuk, hogy az $f(n)$ *növekedési függvény polinomiálisan gyorsabban nő, mint az* n^p *polinom* $(p \geq 0)$*, ha létezik olyan* $\varepsilon > 0$ *valós szám, hogy* $f(n) = \Omega\left(n^{p+\varepsilon}\right)$*.*

17. Definíció. *A polinomiálisan lassabb növekedés*

Azt mondjuk, hogy az $f(n)$ *növekedési függvény polinomiálisan lassabban nő, mint az* n^p *polinom* $(p \geq 0)$*, ha létezik olyan* $\varepsilon > 0$ *valós szám, hogy* $f(n) = O\left(n^{p-\varepsilon}\right)$*.*

18. Definíció. *Fibonacci sorozat*

Fibonacci számsorozatnak nevezzük azt az $F_0, F_1, F_2, \ldots$ *számsorozatot, amelyet az alábbi formulapár határoz meg:*

$$\left.\begin{array}{rcl} F_0 & = & 0 \\ F_1 & = & 1 \end{array}\right\} \text{(Kezdőfeltétel)}$$
$$F_{n+2} = F_{n+1} + F_n \qquad \text{(rekurziós feltétel)}$$

19. Tétel. *(Binet formula) A Fibonacci számsorozat elemei felírhatók az index függvényében az alábbi úgynevezett Binet formula révén:*

$$F_n = \frac{1}{\sqrt{5}}\left(\Phi^n - \overline{\Phi}^n\right) \qquad n = 0, 1, 2, \ldots$$

ahol

$$\Phi = \frac{1+\sqrt{5}}{2} \approx 1.618 \qquad \text{és} \qquad \overline{\Phi} = \frac{1-\sqrt{5}}{2} \approx -0.618$$

20. Tétel. *(A Fibonacci számok előállítása kerekítéssel) Az F_n Fibonacci szám képezhető az alábbi módon a Binet formula felhasználásával:*

$$F_n = \text{Round}\left(\frac{1}{\sqrt{5}}\Phi^n\right) \qquad n = 0, 1, 2, \ldots$$

21. Definíció. *(Rekurzív egyenlet) Rekurzív egyenletnek nevezzük azokat a függvényegyenleteket, amelyekben a T függvény az ismeretlen, a meghatározandó, és a $T(n)$ függvényérték a T függvény n-től kisebb értékű argumentumának helyein felvett értékeinek függvényeként adott.*

22. Tétel. *(A mester tétel:) Legyenek $a \geq 1, b > 1$ konstansok, $p = \log_b a$, $f : \mathbb{N} \to \mathbb{Z}$ függvény. Definiálunk egy $g(n) = n^p$ úgynevezett tesztpolinomot. Legyen a rekurziós összefüggésünk: $T(n) = aT(n/b) + f(n)$. (Az n/b helyén $\lceil n/b \rceil$ vagy $\lfloor n/b \rfloor$ is állhat.) Ezen feltételek esetén igazak az alábbi állítások:*

[1.] Ha $f(n)$ polinomiálisan lassabb növekedésű, mint a $g(n)$ tesztpolinom, akkor

$$T(n) = \Theta(g(n)).$$

[2.] Ha $f(n) = \Theta(g(n))$, akkor

$$T(n) = \Theta(g(n) \cdot \log n).$$

[3.] Ha $f(n)$ polinomiálisan gyorsabb növekedésű, mint a $g(n)$ tesztpolinom, és teljesül az f függvényre az úgynevezett regularitási feltétel, azaz $\exists c < 1$ konstans és $n_0 > 0$ küszöbméret, hogy $n > n_0$ esetén $a \cdot f(n/b) \leq c \cdot f(n)$, akkor

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

Számelméleti algoritmusok

1. Definíció. *(Az oszthatóság) Azt mondjuk, hogy a* d *egész szám osztja az* a *egész számot, ha az osztásnak zérus a maradéka, azaz, ha létezik olyan* k *egész szám, hogy* $a = k \cdot d$. *Jelölésben:* $d|a$. *A* d *számot az* a *osztójának nevezzük. Az* a *szám a* d *többszöröse.*

2. Definíció. *(Prímszám) Prímszámnak nevezzük azt az 1-nél nagyobb egész számot, amelynek csak az 1 és saját maga a pozitív osztója.*

3. Tétel. *(A maradékos osztás tétele) Ha* a *egész szám,* n *pedig pozitív egész szám, akkor egyértelműen létezik olyan* q *és* r *egész szám, hogy*

$$a = q \cdot n + r, \qquad \text{ahol } 0 \leq r < n.$$

A q *szám neve hányados,* r *neve maradék. A hányados és a maradék felírható:*

$$q = \lfloor a/n \rfloor, \qquad r = a - q \cdot n = a \bmod n.$$

4. Definíció. *(Közös osztó) Azt mondjuk, hogy a* d *egész szám az* a *és* b *egészek közös osztója, ha* d *mindkét számot osztja (azaz* $d|a$ *és* $d|b$*).*

5. Definíció. *(Lineáris kombináció) Az* s *egész számot az* a *és* b *egészek (egész) lineáris kombinációjának nevezzük, ha létezik olyan* x *és* y *egész szám, hogy* $s = x \cdot a + y \cdot b$. *Az* x *és* y *számokat a lineáris kombináció együtthatóinak nevezzük. Az* a *és* b *számok összes lineáris kombinációjának halmazát* $L(a, b)$*-vel jelöljük.*

6. Tétel. *(A közös osztó tulajdonságai) Legyen a* d *egész az* a *és* b *egészek közös osztója. Akkor fennállnak az alábbi állítások:*

[1.] $|d| \leq |a|$, *vagy* $a = 0$.

[2.] Ha $d|a$ *és* $a|d$, *akkor* $d = \pm a$.

[3.] A közös osztó osztója az a *és* b *szám minden lineáris kombinációjának is, azaz* $\forall s \in L(a, b)$*-re* $d|s$.

7. Definíció. *(Legnagyobb közös osztó) Az alább megadott* $d*$ *egész számot az* a *és* b *egész számok legnagyobb közös osztójának nevezzük. Jele:* $\text{lnko}(a, b)$.

$$d^* = \text{lnko}(a, b) = \begin{cases} 0 & \text{ha } a = 0 \text{ és } b = 0 \\ \max\limits_{\substack{d|a \\ d|b}} d & \text{egyébként.} \end{cases}$$

8. Definíció. *(Relatív prímek) Az a és b egész számokat relatív prímeknek nevezzük, ha*

$$\text{lnko}(a,b) = 1$$

9. Tétel. *(A legnagyobb közös osztó elemi tulajdonságai) Legyen a $d*$ egész az a és b egészek legnagyobb közös osztója. Akkor fennállnak az alábbi állítások:*

[1.] $1 \leq d^* \leq \min\{|a|,|b|\}$ *kivéve ha* $a = 0$ *és* $b = 0$.

[2.] $\text{lnko}(a,b) = \text{lnko}(b,a) = \text{lnko}(-a,b) = \text{lnko}(|a|,|b|)$.

[3.] $\text{lnko}(a,0) = |a|$.

[4.] $\text{lnko}(a, k \cdot a) = |a|, k \in \mathbb{Z}$.

[5.] Ha d közös osztó és $d^* \neq 0$, akkor $d \leq d^*$.

[6.] A legnagyobb közös osztó minden lineáris kombinációnak osztója, azaz $\forall s \in L(a,b)$*-re* $d^*|s$.

10. Tétel. *(A legnagyobb közös osztó reprezentációs tétele) Ha az a és b egész számok nem mindegyike zérus, akkor a legnagyobb közös osztó megegyezik a két szám pozitív lineáris kombinációinak minimumával.*

$$d^* = \text{lnko}(a,b) = \min_{\substack{s \in L(a,b) \\ s>0}} s = a \cdot x^* + b \cdot y^* = s^*.$$

11. Tétel. (*A lineáris kombinációk halmazának jellemzése) Legyen M a*

$d^* = \text{lnko}(a,b)$ *egész többszöröseinek a halmaza. Állítás:*

$$L(a,b) = M.$$

(*Bővebben: az* $L(a,b)$ *minden eleme* d^* *egész többszöröse és ha egy s szám egész többszöröse, akkor az az s szám az a és b lineáris kombinációja is.)*

12. Tétel. *(A legnagyobb közös osztó redukciós tétele) Tetszőleges a és b két egész szám esetén fennáll, hogy*

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

13. Tétel. *(A legnagyobb közös osztó rekurziós tétele) Tetszőleges a és b két egész szám esetén fennáll, hogy* $\text{lnko}\,(a,b) = \text{lnko}\,(b, a \bmod b)$.

14. Tétel. *(Lamé tétele) Ha az euklideszi algoritmusban* $a > b \geq 0$ *és* $b < F_{k+1}$

valamely $k > 0$*-ra, akkor a rekurziós hívások száma kevesebb, mint* k*.*

15. Definíció. *(Kongruencia) Az* a *és* b *egész számokat kongruensnek mondjuk az* n *modulus szerint, ha az* n *szerinti osztás utáni maradékaik megegyeznek, vagy ami ugyanaz: ha* $n \mid (a - b)$*. Jelölésben:* $a \equiv b \bmod n$*.*

16. Tétel. *(A kongruenciákon végezhető műveletek tétele) Legyen* $a \equiv b \bmod n$ *és* $c \equiv d \bmod n$*. Akkor igazak az alábbi állítások:*

1. $a \pm c \equiv b \pm d \bmod n$
2. $a \cdot c \equiv b \cdot d \bmod n$
3. $\frac{a}{k} \equiv \frac{b}{k} \bmod n \;\; ha \; k \mid a \;,\;\; k \mid b \;\; és \;\; \text{lnko}\,(k, n) = 1$
4. $a \equiv b \bmod m \;\; ha \; m \mid n$

17. Definíció. *(Lineáris kongruencia egyenlet) Az*

$$a \cdot x \equiv b \mod n \qquad a, b \in \mathbb{Z}, n \in \mathbb{N}$$

egyenletet, melyben $x \in Z$ *az ismeretlen, lineáris kongruencia egyenletnek nevezzük.*

18. Tétel. *(A lineáris kongruencia egyenlet megoldhatósági tétele) Legyen az*

$$a \cdot x \equiv b \mod n \qquad a, b \in \mathbb{Z}, n \in \mathbb{N}$$

egyenletre $d^* = \text{lnko}\,(a, n) = a \cdot x^* + n \cdot y^*$*.*

A lineáris kongruencia egyenletnek akkor és csak akkor van megoldása, ha $d^* | b$*. Ha van megoldás, akkor végtelen sok van, de ezeket egy* d^* *számú megoldást tartalmazó úgynevezett megoldás alaprendszerből megkaphatjuk az* n *egész számú többszöröseinek a hozzáadásával. Az alaprendszer elemeit a* $0 \leq x < n$ *intervallumból választjuk ki. Az alaprendszer megoldásai az alábbi módon írhatók fel:*

$$\begin{aligned} x_0 &= x^* \cdot (b/d^*) \bmod n \\ x_i &= x_0 + i \cdot (n/d^*) \bmod n \qquad (i = 1, 2, \ldots, d^* - 1) \end{aligned}$$

19. Definíció. *(A multiplikatív inverz) Legyen a lineáris kongruencia egyenlet*

$$a \cdot x \equiv 1 \mod n \qquad a \in \mathbb{Z}, n \in \mathbb{N}, \qquad lnko(a,n) = 1$$

alakú (azaz a és n legyenek relatív prímek). Az egyenlet egyetlen alapmegoldását az a szám n szerinti multiplikatív inverzének nevezzük. Jelölése:

$$x = a^{-1} \mod n$$

20. Tétel. *(Fermat tétel) Ha p prím, akkor*

$$a^{p-1} \equiv 1 \mathrm{mod} p, \qquad a = 1, 2, \ldots, p-1.$$

Elemi dinamikus halmazok

1. **Definíció.** *Dinamikus halmaz Az olyan halmazt, amely az őt felhasználó algoritmus során változik (bővül, szűkül, módosul) dinamikus halmaznak nevezzük.*

2. Definíció

Lekérdező műveletek	
KERES (S, k, x)	adott k kulcsú elem x mutatóját adja vissza, vagy NIL, ha nincs.
MINIMUM (S, x)	A legkisebb kulcsú elem mutatóját adja vissza
MAXIMUM (S, x)	A legnagyobb kulcsú elem mutatóját adja vissza
KÖVETKEZŐ (S, x, y)	az x elem kulcsa utáni kulcsú elem mutatóját adja vissza,NIL, ha x után nincs elem
ELŐZŐ (S, x, y)	az x elem kulcsa előtti kulcsú elem mutatóját adja vissza,NIL, ha x előtt nincs elem

Módosító műveletek	
BESZÚR (S, x)	az S bővítése az x mutatójú elemmel
TÖRÖL (S, x)	az x mutatójú elemet eltávolítja S-ből

3. Definíció. *(A sorozat adatstruktúra) Sorozatnak nevezzük az objektumok (elemek) olyan tárolási módját (adatstruktúráját), amikor az elemek a műveletek által kijelölt lineáris sorrendben követik egymást. Tipikus műveletek: keresés, beszúrás, törlés.*

4. Definíció. *(Tömb) A* tömb *azonos felépítésű (típusú) egymást fizikailag követő memóriarekeszeket jelent. Egy rekeszben egy elemet,* adatrekordot *helyezünk el. Az egyes tömbelemek helyét az indexük határozza meg. A tömb* attributumai*:*

Attributum	Leírás
fej[A]	A tömb első elemének indexe. NIL, ha a tömbnek nincs eleme.
vége[A]	A tömb utolsó elemének indexe. NIL, ha a tömbnek nincs eleme.
hossz[A]	A tömbelemek száma. Zérus, ha a tömbnek nincs eleme.
tömbméret[A]	annak a memóriaterületnek a nagysága tömbelem egységben mérve, ahová a tömböt elhelyezhetjük. A tömb ezen terület elején kezdődik.

5. Definíció. *(Láncolt lista adatstruktúra) A láncolt lista (linked list) olyan dinamikus halmaz, melyben az objektumok, elemek lineáris sorrendben követik egymást. A lista minden eleme mutatót tartalmaz a következő elemre. Műveletei: keresés, beszúrás, törlés.*

6. Definíció. *(A verem adatstruktúra) A verem (stack) olyan dinamikus halmaz, amelyben előre meghatározott az az elem, melyet a TÖRÖL eljárással eltávolítunk. Ez az elem mindig az időben a legutoljára a struktúrába elhelyezett elem lesz. Műveletei: beszúrás (push), törlés (pop). Az ilyen törlési eljárást Utolsóként érkezett – Elsőként távozik (Last In – First Out, LIFO) eljárásnak nevezzük.*

7. Definíció. *(A sor adatstruktúra) A sor (queue) olyan dinamikus halmaz, amelyben előre meghatározott az az elem, melyet a TÖRÖL eljárással eltávolítunk és az az elem is amelyet a BESZÚR eljárással a halmazba beteszünk. Törlésre mindig az elemek közül a legrégebben beszúrt kerül. A beszúrt elem lesz a legfrissebb elem. Műveletek: beszúrás, törlés. Az ilyen törlést Elsőként érkezik – Elsőként távozik (First In – First Out, FIFO) eljárásnak nevezzük.*

8. Definíció. *(A kiválasztási probléma) Legyen adott egy A halmaz (n különböző szám), és egy i index $1 \leq i \leq n$. Meghatározandó az A halmaz azon x eleme, melyre nézve pontosan $i-1$ darab tőle kisebb elem van az A halmazban.*

9. Definíció. *(Medián) Mediánnak nevezzük az adatsor azon elemét, amely a rendezett sorban a középső helyet foglalja el. Ha páratlan számú elem van az adatsorban, akkor $n = 2k-1$ és így a medián indexe a rendezés után k. Ha páros számú elem van az adatsorban, akkor $n = 2k$, és ekkor két középső elem van ak és a $k+1$ indexű a rendezés után. (Alsó medián, felső medián.)*

10. Definíció. ***(****A rendezési algoritmusokkal kapcsolatban felmerülő igények)*

a.	*Helyben rendezés*, azaz a rendezés eredménye az eredeti helyén jelenjen meg, legfeljebb konstans méretű többletmemória felhasználása révén.
b.	*Gyorsaság.* A rendezési idő legyen minél rövidebb.
c.	*Adaptivitás.* Az algoritmus használja ki a kulcsok között már meglévő rendezettséget.
d.	*Stabilitás.* A rendezés őrizze meg az azonos kulcsú rekordok esetén a rekordok egymáshoz képesti eredeti sorrendjét. (Például telefonszámlák készítésekor az azonos kulcsú előfizetői hívások időrendi sorrendje maradjon meg.)
e.	Az algoritmus csak a kulcsokat rendezze a rekordokra mutató pointerekkel, vagy az összes rekordot mozgassa.
f.	*Belső rendezés* legyen (csak a belső memóriát vegye igénybe a rendezéshez), vagy *külső rendezés* legyen (háttértárakat is igénybe vehet).
g.	*Összehasonlítás*on alapuljon a rendezés, vagy azt ne vegye igénybe az algoritmus. (Ez utóbbi esetben a kulcsokra további megszorításokat kell tenni.)
h.	Optimális legyen a rendezési algoritmus, vagy sem. (Nem biztos, hogy az adatok az optimális algoritmus által megkívánt módon vannak megadva.)
i.	Az összes rendezendő adatnak rendelkezésre kell-e állnia a rendezés teljes folyamata alatt, vagy sem.
j.	A rendezésnek csak a befejeztével van eredmény, vagy menet közben is a már rendezett rész tovább nem változik.

11. Definíció. *(Az oszd meg és uralkodj elv) Az oszd meg és uralkodj elv egy algoritmus tervezési stratégia A problémát olyan kisebb méretű, azonos részproblémákra osztjuk föl, amelyek rekurzívan megoldhatók. Ezután egyesítjük a megoldásokat.*

Gráfelméleti fogalmak, jelölések

1. Definíció. *Legyen V egy véges halmaz, E pedig V-beli rendezetlen elempárok véges rendszere. Ekkor a $G = (V, E)$ párt gráfnak nevezzük.*

2. Definíció. *A $G = (V, E)$ rendezett párt irányított gráfnak (digráfnak) nevezzük. A rendezett pár elemeire tett kikötések: V véges halmaz, a G-beli csúcsok halmaza. E bináris reláció a V halmazon, az élek halmaza.*

E={ (u, v) rendezett pár | $u \in V, v \in V$} $\subset V \times V$. Hurkok megengedettek.

3. Definíció. *A $G = (V, E)$ rendezett párt irányítatlan gráfnak nevezzük. A rendezett pár elemeire tett kikötések: V véges halmaz, a G-beli csúcsok halmaza. E bináris reláció a V halmazon, az élek halmaza.*

E={ (u, v) rendezetlen pár | $u \in V, v \in V$} $\subset V \times V$. Hurok nem megengedett.

4. Definíció. *Az a gráf (irányított vagy irányítatlan), amelynek minden éléhez egy számot (súlyt) rendelünk hozzá, hálózatnak (súlyozott gráfnak) nevezzük.*

5. Definíció. *A gráf egymáshoz csatlakozó éleinek olyan sorozatát, amely egyetlen ponton sem megy át egynél többször, útnak nevezzük.*

6. Definíció. *Az u csúcsból kiinduló és a v csúcsba mutató él digráfban az (u,v) él.*

7. Definíció. *Az u csúcsból kiinduló és a v csúcsba mutató él irányítatlan gráfban az (u,v) él*

8. Definíció. (*Az u csúcs szomszédja) Legyen (u,v) él egy G = (V,E) gráfban. Ekkor a v csúcsot az u csúcs szomszédjának nevezzük. A szomszédság reláció irányítatlan gráfban szimmetrikus, digráfban nem.*

9. Definíció. *(Csúcs fokszáma irányítatlan gráfban) A csúcs fokszáma a belőle kiinduló élek száma.*

10. Definíció. *(Csúcs fokszáma digráfban)*

Kimenő fokszám (kifok): *a csúcsból kimenő élek száma*
Bemenő fokszám (befok): *a csúcsba bemenő élek száma*
Csúcs fokszáma: *kifok+befok*

11. Definíció. *(Izolált csúcs) Csúcs, melynek fokszáma zérus.*

12. Definíció. *(Az u csúcsot az u' csúccsal összekötő k hosszúságú út)*
Csúcsok véges sorozata: $v_0, v_1, \dots, v_k$, *ahol* $v_0 = u, v_k = u'$ *és* $(v_i v_{i+1}) \in E$ $(i = 0,1, \dots, k-1)$

13. Definíció. *(Egyszerű út) Olyan út, melyben a benne szereplő csúcsok páronként különbözőek.*

14. Definíció. *(Az u' csúcs elérhető az u csúcsból) Azt mondjuk, hogy az u' csúcs elérhető az u csúcsból, ha van olyan út, amely az u csúcsot az u' csúccsal összeköti.*

15. Definíció. *(Körmentes gráf) Olyan gráf, amely nem tartalmaz kört.*

16. Definíció. (*Összefüggő gráf) Ha bármely két csúcsa összeköthető úttal.*

17. Definíció. *(Gráf részgráfja) A G=(E,V) gráf részgráfja a G'=(E',V') melyre* $V' \subseteq V$ *és* $E' \subseteq E$.

18. Definíció. *(Teljes gráf) Irányítatlan gráf, melyben bármely két csúcs szomszédos. (Minden lehetséges él benne van.)*

19. Definíció. *(Erdő) Körmentes, irányítatlan gráf.*

20. Definíció. *(Nyílt fagráf) Összefüggő, körmentes, irányítatlan gráf.*

21. Tétel: *A nyílt fák tulajdonságai Legyen G=(E,V) irányítatlan gráf. Az alábbiak ekvivalensek.*

1. *G nyílt fa*
2. *G bármely két csúcsához egyértelműen létezik egy őket összekötő egyszerű út.*
3. *G összefüggő, de tetszőleges élének elhagyása után a visszamaradó gráf már nem összefüggő*
4. *G összefüggő és* $|E| = |V| - 1$
5. *G körmentes és* $|E| = |V| - 1$
6. *G körmentes, de akár egyetlen éllel is bővítve E-t a kapott gráf már tartalmaz kört.*

22. Definíció. *(Gyökeres fa) T fagráf, amely egyik csúcsának kitüntetett a szerepe a többihez képest. Ez a csúcs a gyökér vagy gyökércsúcs (r).*

23. Definíció. *(Az x csúcs megelőzője) A gyökérből x-be vezető úton fekvő bármely csúcs. (x is a saját megelőzője.)*

24. Definíció. *(x az y rákövetkezője) ha y x-nek megelőzője. (x is a saját rákövetkezője.)*

25. Definíció. *(x szülője y) ha az r-ből y-ba vezető úton (y,x) az utolsó él. (A gyökérnek nincs szülője T-ben.)*

26. Definíció. *(x az y gyereke) ha y az x szülője.*

27. Definíció. *(Testvérek) Azok a csúcsok, amelyeknek ugyanaz a csúcs a szülője.*

28. Definíció. *(Külső csúcs vagy levél) Az a csúcs, amelynek nincs gyereke.*

29. Definíció. *(Belső csúcs) Az a csúcs, amely nem levél.*

30. Definíció. *(x fokszáma gyökeres fában) az x gyerekeinek száma. (A szülő nem számít bele a fokszámba!)*

31. Definíció. *(x szintje) Az r-ből x-be vezető út hossza.*

32. Definíció. *(T magassága) a T-beli csúcsok szintjei közül a legnagyobb.*

33. Definíció. *(Bináris fa) Rendezett fa, melyben minden csúcs fokszáma legfeljebb kettő. Beszélhetünk bal gyerekről és jobb gyerekről.*

34. Definíció. *(Teljes bináris fa) Bináris fa, melyben a csúcsok fokszáma kettő, kivéve a leveleket, melyeké 0, valamint az összes levél azonos szinten helyezkedik el.*

35. Definíció. *(k-adrendű teljes fa) k-adrendű fa, melyben a levelek ugyanazon szintűek és az összes belső csúcs fokszáma k.*

36. Definíció. *(A bináris kereső fa)*
A bináris kereső fa egy bináris fa, amely rendelkezik az alábbi bináris kereső fa tulajdonsággal:

Legyen x a bináris kereső fa tetszőleges csúcsa.
Ha y az x baloldali részfájában van, akkor $kulcs[y] \leq kulcs[x]$.
Ha y az x jobboldali részfájában van, akkor $kulcs[x] \leq kulcs[y]$.

A jellemző műveletek bináris kereső fában:
KERES, MINIMUM, MAXIMUM, ELŐZŐ, KÖVETKEZŐ, BESZÚR, TÖRÖL.

Algoritmusok

1. *Kibővített Euklideszi algoritmus:*

1	Kibővített_Euklidesz (a, b, d^*, x^*, y^*)
2	// Input paraméterek : a,b∈Z, a,b≥0
3	// Output paraméterek: d^*, x^*, y^*∈Z, d^*≥0
4	**IF** $b = 0$
5	**THEN** $d^* \leftarrow a$
6	$x^* \leftarrow 1$
7	$y^* \leftarrow 0$
8	**ELSE** Kibővített_Euklidesz (b, a mod b, d^*, x^*, y^*)
9	$\begin{pmatrix} x^* \\ y^* \end{pmatrix} \leftarrow \begin{pmatrix} y^* \\ x^* - \lfloor a/b \rfloor \cdot y^* \end{pmatrix}$
10	**RETURN** (d^*, x^*, y^*)

2. *Lineáris kongruencia egyenlet megoldása:*

1	Lineáris_kongruencia_megoldó (a, b, n, X)
2	// Input paraméterek: a,b,n∈Z, n>0
3	// Output paraméter : X – egyindexes tömb
4	// indexelés 0-tól
5	Kibővített_Euklidesz (*a, n,* d^*, x^*, y^*)
6	Hossz[X]← *0*
7	**IF** $d^* \mid b$
8	**THEN** $x_0 \leftarrow x^* \cdot (b/d^*) \bmod n$
9	Hossz[X]← d^*
10	**FOR** $i \leftarrow 1$ **TO** $d^* - 1$ **DO**
11	$x_i \leftarrow x_0 + i \cdot (n/d^*) \bmod n$
12	**RETURN** (X)
13	// Hossz[X]=0 jelenti, hogy nincs megoldás

3. *Moduláris hatványozás:*

1	Moduláris_hatványozó (a, b, n, c)
2	// Input paraméterek: a,b,n∈Z, a,b,n>0
3	// Output paraméter: c∈Z, c≥0
4	$p \leftarrow 0$
5	$c \leftarrow 1$
6	**FOR** $i \leftarrow k$ **DOWNTO** 0 **DO**
7	$p \leftarrow 2p$
8	$c \leftarrow c^2 \bmod n$
9	**IF** $b_i = 1$
10	**THEN** $p \leftarrow p+1$
11	$c \leftarrow (c \cdot a) \bmod n$
12	**RETURN** (c)

4. *RSA algoritmus:*

1	RSA_kulcsok_meghatározása (p, q, e, P, S)
2	// Input paraméterek: p, q, e
3	// Output paraméterek: P, S
4	**IF** p vagy q nem prím vagy $e<3$ vagy e páros
5	**THEN RETURN** („Nincs kulcs”)
6	$n \leftarrow p \cdot q$
7	$f \leftarrow (p-1) \cdot (q-1)$
8	**IF** $lnko(e, f) \neq 1$
9	**THEN RETURN** („Nincs kulcs”)
10	$d \leftarrow e^{-1} \bmod f$
11	**RETURN** $(P = (e,n), S = (d,n))$

5. *Keresés tömbben:*

```
1  KERESÉS_TÖMBBEN (A, k, x )
2  // Input paraméter: A - a tömb
3  //                  k – a keresett kulcs
4  // Output paraméter: x - a k kulcsú elem pointere (indexe), ha van ilyen elem
                          vagy NIL, ha nincs
5  // Lineárisan keresi a k kulcsot.
6  //
7  x ← fej[A]
8  IF  hossz[A] ≠ 0
9      THEN    WHILE    x ≤ vége[A] és kulcs[A_x] ≠ k DO
10                      INC(x)
11             IF       x> vége[A]
12                      THEN x ← NIL
13 RETURN (x)
```

6. *Láncolt listában keresés:*

```
1  LISTÁBAN_KERES ( L, k, x )
2  // Input paraméter:   L – a lista
3  //                    k – a keresett kulcs
4  // Output paraméter:  x   a kulcselem pointere.
                          NIL, ha a kulcs nincs a listában.
5  //
6  x ← fej[L]
7  WHILE   x ≠ NIL és kulcs[x] ≠ k DO
8              x ← köv[x]
9  RETURN (x)
```

7. *Láncolt listába beszúrás:*

```
1  LISTÁBA_BESZÚR ( L, x )
2  // Input paraméter:   L – a lista
   //                    x  a beszúrandó elemre mutató mutató
3  // Az új elemet a lista elejére teszi
4  //
5  köv[x] ← fej[L]
6  elő[x]  ← NIL
7  IF    fej[L] ≠ NIL
8        THEN elő[fej[L]] ← x
9  fej[L] ← x
10 RETURN
```

8. *Minimumkeresés tömbben:*

1	MINIMUMKERESÉS(A, min)
2	// Input paraméter: A – a tömb
3	// Output paraméter: min - a minimum értéke
4	//
5	min ← A_1
6	FOR i←2 TO hossz[A] DO
7	IF min > A_i
8	THEN min ← A_i
9	RETURN (min)

9. *Feloszt eljárás:*

	4.1.5.1. algoritmus **Résztömb felosztása előre adott érték körül**
	// $T(n)=\Theta(n)$
1	FELOSZT(A,p,r,x, q)
2	// Input paraméter: A – a tömb
3	// p - a felosztandó rész kezdőindexe
4	// r - a felosztandó rész végindexe
5	// x - az előre megadott érték, amely a felosztást szabályozza
6	// Output paraméter: A – a megváltozott tömb
7	// q – a felosztás határa $A_{p...q}$; $A_{q+1...r}$
8	//
9	i ← p – 1
10	j ← r + 1
11	**WHILE IGAZ DO**
12	**REPEAT** j ← j – 1
13	**UNTIL** $A_j \le x$
14	**REPEAT** i ← i + 1
15	**UNTIL** $A_i \ge x$
16	**IF** i < j
17	**THEN** Csere $A_i \leftrightarrow A_j$
18	**ELSE** q ← j
19	**RETURN** (A, q)

10.Beszúró rendezés:

```
1  BESZÚRÓ_RENDEZÉS ( A )
2  // Input paraméter: A - a rendezendő tömb
3  // Output paraméter: A - a rendezett tömb
4  //
5  FOR j ← 2 TO hossz[A] DO
6        kulcs ← A_j
7        // Beszúrás az A_{1..j-1} rendezett sorozatba
8        i ← j − 1
9        WHILE i > 0 és A_i > kulcs DO
10             A_{i+1} ← A_i
11             DEC(i)
12       A_{i+1} ← kulcs
13 RETURN (A)
```

+

11.*Bináris fába beszúrás:*

```
1  FÁBA_BESZÚR(T, z)
2  y ← NIL
3  x ← gyökér[T]
4  WHILE x ≠ NIL DO
5            y ← x
6            IF kulcs[z] < kulcs[x]
7                 THEN x ← bal[x]
8                 ELSE x ← jobb[x]
9  szülő[z] ← y
10 IF y = NIL
11      THEN gyökér[T] ← z
12      ELSE IF kulcs[z] < kulcs[y]
13                THEN bal[y] ← z
14                ELSE jobb[y] ← z
```

12. Bináris fából törlés:

FÁBÓL_TÖRÖL(T, z)	
Három esetre bontjuk a törlést:	
z-nek nincs gyereke:	egyszerűen kivágjuk
z-nek egy gyereke van:	kivágjuk úgy, hogy a szülője és a gyereke között kapcsolatot hozunk létre.
z-nek két gyereke van:	kivágjuk z azon legközelebbi rákövetkezőjét, amelynek már nincs baloldali gyereke és ezt a rákövetkezőt z helyére illesztjük.

13. Inorder fabejárás:

```
1  INORDER_FA_BEJÁRÁS(x)
2  IF x ≠ NIL
3      THEN INORDER_FA_BEJÁRÁS(bal[x])
4           Print(kulcs[x])
5           INORDER_FA_BEJÁRÁS(jobb[x])
```