

# INFORMATIKAI ALAPOK

# INFORMATIKAI ALAPOK

- Számrendszerek
- Boole algebra

# SZÁMRENDSZEREK I.

- Az informatika története során a fejlesztők több számrendszer alapján is elindultak a fejlesztéssel, így például:
  - kettes számrendszer
  - hármas számrendszer
  - tízes számrendszer
- Természetesen a projektek nagy része sikertelen volt, így a kettes számrendszer lett a mai informatika alapja.

## SZÁMRENDSZEREK II.

- A kettes (bináris) számrendszer a mai informatika alapja, de emellett használnak segédszámrendszereket is:
  - Tízes (decimális) számrendszer
    - a felhasználókkal való eredményközlésre, adatbevitelre
  - Tizenhatos (hexadecimális) számrendszer
    - minden olyan kettes számrendszerbeli számot, amelyet már túl hosszán lehetne leírni, ebben a számrendszerben írnak fel

# A TÍZES SZÁMRENDSZER I.

- A tízes (**decimális**) számrendszer az a számrendszer, amelyet mi magunk is a hétköznapi életben alkalmazunk.
- Az informatikában mint segédszámrendszer jelenik meg, hiszen az embernek ezzel a számrendszerrel a legegyszerűbb dolgoznia.
- Az informatikában azok a szabályok, amelyeket a matematika felállított a tízes számrendszeren belül, ugyancsak érvényesek.
  - Műveleti prioritások, zárójelezések, nullával osztás tiltása, stb.

## A TÍZES SZÁMRENDSZER II.

- A tízes számrendszerben a 0..9-ig terjedő számok szerepelnek.
- És ily módon, minden egyes hatványérték ezeket a számokat veheti fel:

9	5	3	5	4
$10^4$	$10^3$	$10^2$	$10^1$	$10^0$
$9 \cdot 10000$	$5 \cdot 1000$	$3 \cdot 100$	$5 \cdot 10$	$4 \cdot 1$

- Így látható, hogy ha 10-et felvenné bármelyik szám, az pont a következő hatványérték lenne.

# A TIZENHATOS SZÁMRENDSZER I.

- Más néven **hexadecimális** számrendszer.
- Ez a számrendszer is segédszámrendszerként szerepel az informatikában.
- Felhasználása nagy jelentőséggel bír a következő területeken:
  - Memóriacímek jelölésében
  - Az IPv6 címekben
  - Fizikai címekben (MAC address)
  - RGB színkód megadásában

## A TIZENHATOS SZÁMRENDSZER II.

- A tizenhatos számrendszer felépítése: a számokat 0..9-ig, illetve a betűket A..F-ig tartalmazza.
  - Az A-F betűk sorrendben 10..15-ig jelölik a helyiértékeket, ezzel elkerülve, hogy egy-egy érték akár két helyiértéket is elfoglalhasson.
- A számrendszer azért is oly kedvelt segédszámrendszerként, mert a tizenhat egymást követő hatványai megegyeznek a kettő minden negyedik egymást követő hatványával.
  - Négy bináris számjegyet egyetlen hexadecimális számjeggyel helyettesíthetünk.



## A TIZENHATOS SZÁMRENDSZER III.

- A tizenhatos számrendszerben tehát egy szám ábrázolása így néz ki:

f	3	a	2	b
$16^4$	$16^3$	$16^2$	$16^1$	$16^0$
15*65536	3*4096	10*256	2*16	11*1

- Mint látható, a tízes számrendszerhez hasonlóan a tizenhatos számrendszerben az egyes tagok a 16-nak 0-tól induló kitevőjű hatványait jelölik.

# A KETTES SZÁMRENDSZER I.

- A kettes számrendszer más néven **bináris** számrendszer.
- A számrendszerben mindösszesen két szám található: 0 és 1.
- A számrendszer azért a legalkalmasabb elektronikai eszközökön való számítások végzésére és működtetésére, mivel az elektronikában könnyen és pontosan reprodukálható a számrendszer.

## A KETTES SZÁMRENDSZER II.

- Hiszen a számrendszer elektronikai eszközökön a vezetékekben futó alacsony / magas feszültség szint váltakozásának segítségével felépíthető.
- A magas feszültség szint az 1-es (igaz), az alacsony pedig a 0-s (hamis) értéket jelöli.
- A programozásban is nagy szerepet játszik a kettes számrendszer, hiszen a matematikai logika, amely erre a számrendszerre alapoz, a programozás alapja.

## A KETTES SZÁMRENDSZER III.

- A kettes számrendszer felépítése tehát az alábbiak szerinti:

1	0	1	1	0
$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
$1*16$	$0*8$	$1*4$	$1*2$	$0*1$

- A kettes számrendszer is ugyanazon elvek alapján épül fel, mint az előzőekben látott két számrendszer.

# ÁTVÁLTÁS 10-ES SZÁMRENDSZERBŐL 2-ESBE

$$131_{10} = 1000?011_2$$

## Átváltás menete:

1. Készítsünk egy 2-oszlopos táblázatot
2. Írjuk fel a számot a bal felső sarokba
3. Osszuk el a számot 2-vel
  - a) Az osztás eredményét írjuk a szám alá
  - b) Az osztás maradékát írjuk a szám mellé
4. Az osztást ismételgessük, amíg a bal oldalon 0-t nem kapunk
5. A jobb oldali oszlop számjegyeit olvassuk össze letről felfelé

$\div 2$	131		1
	65		1
	32		0
	16		0
	8		0
	4		0
	2		0
	1		1
	0		

Átváltás 10-es számrendszerből 2-esbe


# ÁTVÁLTÁS 10-ES SZÁMRENDSZERBŐL 8-ASBA

$$131_{10} = 203_8$$

## Átváltás menete:

1. Készítsünk egy 2-oszlopos táblázatot
2. Írjuk fel a számot a bal felső sarokba
3. Osszuk el a számot 8-cal
  - a) Az osztás eredményét írjuk a szám alá
  - b) Az osztás maradékát írjuk a szám mellé
4. Az osztást ismételgessük, amíg a bal oldalon 0-t nem kapunk
5. A jobb oldali oszlop számjegyeit olvassuk össze letről felfelé

<b>:8</b> ↻	<b>131</b>	<b>3</b>
	<b>16</b>	<b>0</b>
	<b>2</b>	<b>2</b>
	<b>0</b>	



Átváltás 10-es számrendszerből 8-asba


# ÁTVÁLTÁS 10-ES SZÁMRENDSZERBŐL 16-OSBA

$$131_{10} = 83_{16}$$

## Átváltás menete:

1. Készítsünk egy 2-oszlopos táblázatot
2. Írjuk fel a számot a bal felső sarokba
3. Osszuk el a számot 16-tal
  - a) Az osztás eredményét írjuk a szám alá
  - b) Az osztás maradékát írjuk a szám mellé
4. Az osztást ismételgessük, amíg a bal oldalon 0-t nem kapunk
5. A jobb oldali oszlop számjegyeit olvassuk össze letről felfelé

:16 ↺	131	3
	8	8
	0	



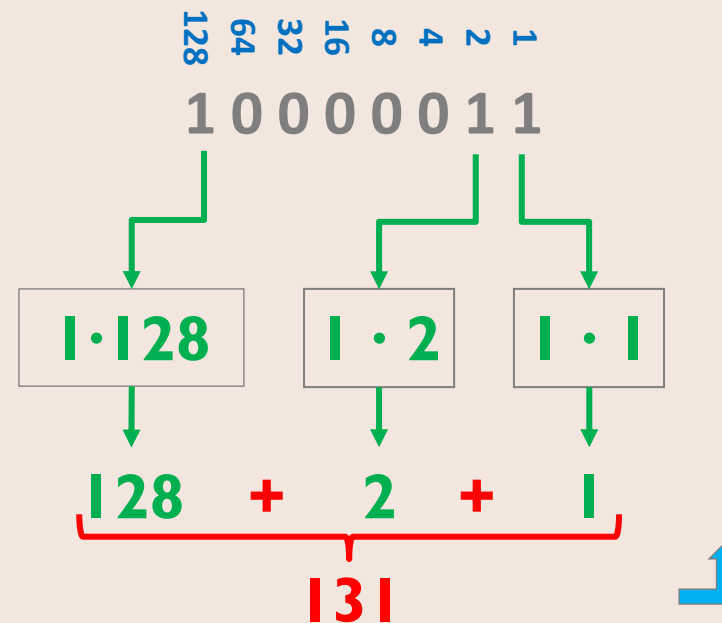
Átváltás 10-es számrendszerből 16-osba

# ÁTVÁLTÁS 2-ES SZÁMRENDSZERBŐL 10-ESBE

$$10000011_2 = 131$$

## Átváltás menete:

1. Írjuk fel az átváltandó számot
2. Írjuk a számjegyek fölé 2 hatványait
3. Szorozzuk össze a számjegyeket a fölöttük lévő hatványokkal
4. Adjuk össze a szorzatokat
5. Az összeg lesz a végeredmény



Átváltás 2-es számrendszerből 10-esbe

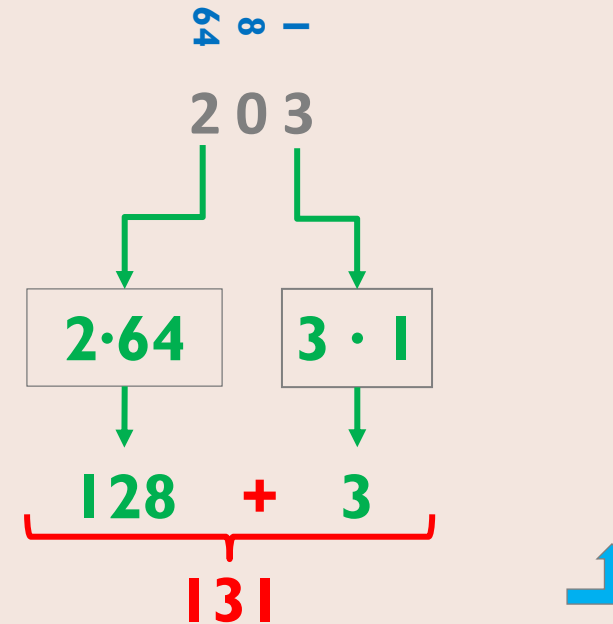


# ÁTVÁLTÁS 8-AS SZÁMRENDSZERBŐL 10-ESBE

$$203_8 = \underline{131}_{10}$$

## Átváltás menete:

1. Írjuk fel az átváltandó számot
2. Írjuk a számjegyek fölé 8 hatványait
3. Szorozzuk össze a számjegyeket a fölöttük lévő hatványokkal
4. Adjuk össze a szorzatokat
5. Az összeg lesz a végeredmény



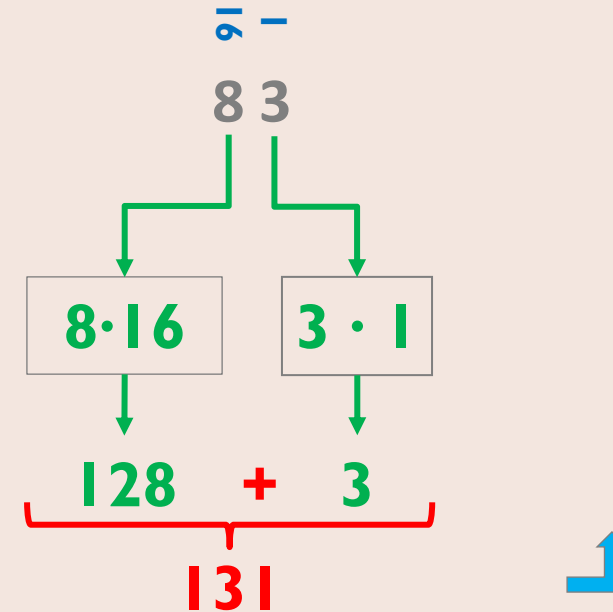
Átváltás 8-as számrendszerből 10-esbe

# ÁTVÁLTÁS 16-OS SZÁMRENDSZERBŐL 10-ESBE

$$83_{16} = \mathbf{131}_{10}$$

## Átváltás menete:

1. Írjuk fel az átváltandó számot
2. Írjuk a számjegyek fölé 16 hatványait
3. Szorozzuk össze a számjegyeket a fölöttük lévő hatványokkal
4. Adjuk össze a szorzatokat
5. Az összeg lesz a végeredmény



Átváltás 16-os számrendszerből 10-esbe

# KÜLÖNBSÉG AZ ÁTVÁLTÁSOKNÁL

## 10-esből X-esbe

**Átváltás menete:**

1. Készítsünk egy 2-oszlopos táblázatot
2. Írjuk fel a számot a bal felső sarokba
3. Osszuk el a számot X-szel
  - a) Az osztás eredményét írjuk a szám alá
  - b) Az osztás maradékát írjuk a szám mellé
4. Az osztást ismételgessük, amíg a bal oldalon 0-t nem kapunk
5. A jobb oldali oszlop számjegyeit olvassuk össze letről felfelé

## X-esből 10-esbe

**Átváltás menete:**

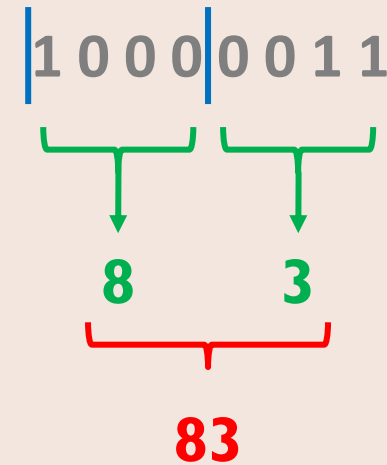
1. Írjuk fel az átváltandó számot
2. Írjuk a számjegyek fölé X hatványait
3. Szorozzuk össze a számjegyeket a fölöttük lévő hatványokkal
4. Adjuk össze a szorzatokat
5. Az összeg lesz a végeredmény

# ÁTVÁLTÁS 2-ES SZÁMRENDSZERBŐL 16-OSBA

$$10000011_2 = 83_{16}$$

## Átváltás menete:

1. Írjuk fel az átváltandó számot
2. **Hátulról indulva osszuk fel a számot**  
4 bites csoportokra (digitekre), ha kell, írjunk 0-kat a szám elé
3. **A 4 bites csoportokat egyenként alakítsuk át (segédtábla segítségével)**
4. **Az átváltások eredményét balról jobbra kell összeolvasni**
5. **A lesz a végeredmény**



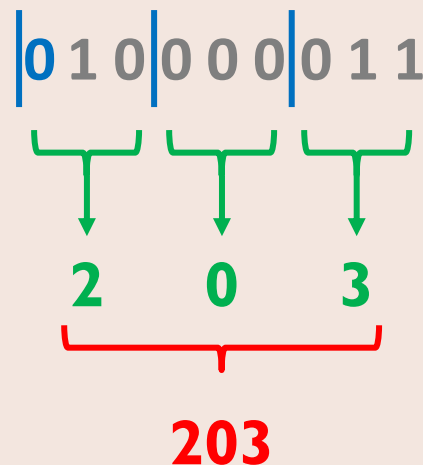
## Átváltás 2-es számrendszerből 16-osba

# ÁTVÁLTÁS 2-ES SZÁMRENDSZERBŐL 8-ASBA

$$10000011_2 = 203$$

## Átváltás menete:

1. Írjuk fel az átváltandó számot
2. **Hátulról indulva osszuk fel a számot**  
3 bites csoportokra, ha kell, írjunk 0-kat a szám elé
3. **A 3 bites csoportokat egyenként alakítsuk át (segédtábla segítségével)**
4. **Az átváltások eredményét balról jobbra kell összeolvasni**
5. **A kapott szám lesz a végeredmény**



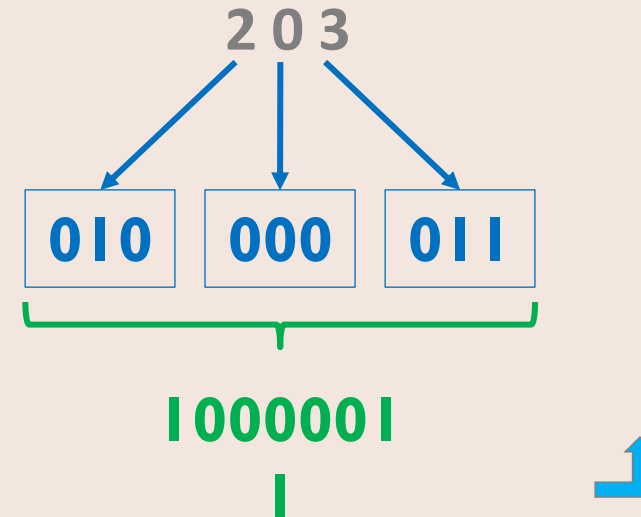
Átváltás 2-es számrendszerből 8-asba

# ÁTVÁLTÁS 8-AS SZÁMRENDSZERBŐL 2-ESBE

$$203_8 = 1000\text{?}0011$$

## Átváltás menete:

1. Írjuk fel az átváltandó számot
2. **Minden számjegyet írjunk át 3 bites bináris számra (segédtáblával)**
3. **A 3 bites csoportokat balról jobbra olvassuk össze (elején lévő 0-kat nem)**
4. **A kapott szám lesz a végeredmény**



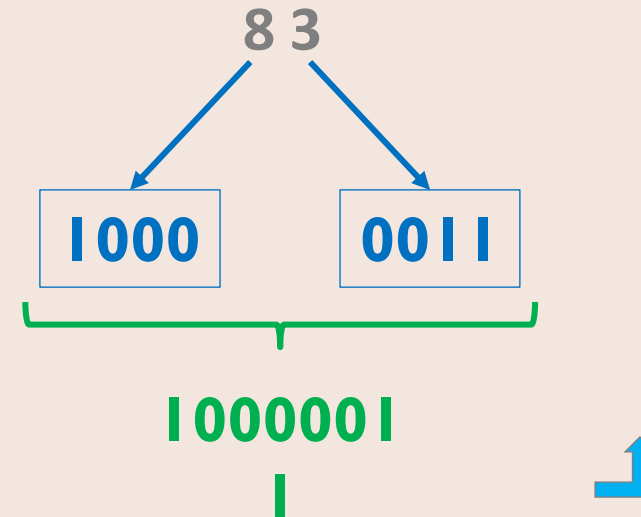
# Átváltás 8-as számrendszerből 2-esbe

# ÁTVÁLTÁS 16-OS SZÁMRENDSZERBŐL 2-ESBE

$$83_{16} = 1000?0011$$

## Átváltás menete:

1. Írjuk fel az átváltandó számot
2. **Minden számjegyet írjunk át 4 bites bináris számra (segédtáblával)**
3. **A 4 bites csoportokat balról jobbra olvassuk össze (elején lévő 0-kat nem)**
4. **A kapott szám lesz a végeredmény**



# Átváltás 16-os számrendszerből 2-esbe

# KÜLÖNBSÉG AZ ÁTVÁLTÁSOKNÁL

## 2-esből 8-asba vagy 16-osba

### Átváltás menete:

1. Írjuk fel az átváltandó számot
2. Hátról indulva osszuk fel a számot 3 vagy 4 bites csoportokra, ha kell, írjunk 0-kat a szám elé
3. A 3-4 bites csoportokat egyenként alakítsuk át (segédtábla segítségével)
4. Az átváltások eredményét balról jobbra kell összeolvasni
5. A kapott szám lesz a végeredmény

## 8-asból vagy 16-osból 2-esbe

### Átváltás menete:

1. Írjuk fel az átváltandó számot
2. Minden számjegyet írjunk át 3 vagy 4 bites bináris számra (segédtáblával)
3. A 3-4 bites csoportokat balról jobbra olvassuk össze (elején lévő 0-kat nem)
4. A kapott szám lesz a végeredmény



# FELADATOK

Végezze el az alábbi átalakításokat!

1.  $2010_{10} = ?_8$
2.  $2010_{10} = ?_2$
3.  $2010_{10} = ?_{16}$
4.  $111010011_2 = ?_{10}$
5.  $2010_8 = ?_{10}$
6.  $2010_{16} = ?_{10}$
7.  $111010011_2 = ?_8$
8.  $2010_8 = ?_2$
9.  $2010_{16} = ?_8$

Végezze el az alábbi  
átalakításokat!

1.  $1011010011_2 = ?_{10}$
2.  $1011010011_2 = ?_8$
3.  $1011010011_2 = ?_{16}$
4.  $E3A_{16} = ?_{10}$
5.  $E3A_{16} = ?_8$
6.  $E3A_{16} = ?_2$
7.  $732_8 = ?_{10}$
8.  $732_8 = ?_{16}$
9.  $732_8 = ?_2$

# LOGIKA ALAPJAI

George Boole megállapítása, Augustus De Morgan azonossága

# LOGIKA ALAPJAI

- A programozás alapvetően a matematikai logikára épül, így az ott alkalmazott szabályok érvényesülnek a programozásban is.
- Az informatika és így a programozás egyik legfontosabb logikai alaptétele, amely megalapozza a mai elektronikai eszközök működési elvét, George Boole megállapítása volt.

# GEORGE BOOLE MEGÁLLAPÍTÁSA

- Az 1800-as években leírt két munkája alapján minden komplex művelet levezethető igenek és nemek sorozatából.
- Ez alapján bármilyen matematikai művelet, komplexitásától függetlenül, levezethető kettes számrendszerben is.
- Az elmélet természetesen önmagában nem alkalmas a mai informatika problémáinak megoldására.
  - George Boole maga publikálta a **Boole-algebrát**, mely a műveleteket is definiálja az elméletéhez.
  - Ezek a műveletek már végrehajthatók a processzoron, ami a meglévő adatokból új információkat állít elő.

# BOOLE ALGEBRA

- A kettes számrendszer, ahogy azt már láthattuk, a matematikai logika alapjának is tekinthető, hiszen az **1** megfelel az **igaz**, a **0** pedig a **hamis** értéknek.
- Ezáltal a matematikai logika (kijelentéslogika és következtetéslogika, illetve Boole-algebra) alapvető megállapításokat, axiómákat értelmez rajta.
- Ezen megállapítások kihatnak a programozásra is, illetve különböző helyzetekben alkalmazhatók.

# BOOLE ALGEBRA

- A kettes számrendszerben, a logikai kapcsolatok az alábbiak szerint alakulnak:
  - ÉS kapcsolat (**AND**)
    - A végeredmény akkor igaz, ha minden érték igaz.
  - VAGY kapcsolat (**OR**)
    - A végeredmény akkor igaz, ha bármely érték igaz.
  - KIZÁRÓ VAGY kapcsolat (**XOR**)
    - A végeredmény akkor igaz, ha bármely érték igaz, azonban az összes érték egyszerre nem igaz. Az igazak száma páratlan.

# BOOLE ALGEBRA

- **ÉS** kapcsolat igazságtáblája:

Első érték	Második érték	Eredmény
0	0	<b>0</b>
0	1	<b>0</b>
1	0	<b>0</b>
1	1	<b>1</b>

- **VAGY** kapcsolat igazságtáblája:

Első érték	Második érték	Eredmény
0	0	<b>0</b>
0	1	<b>1</b>
1	0	<b>1</b>
1	1	<b>1</b>

# BOOLE ALGEBRA

- KIZÁRÓ VAGY kapcsolat igazságtáblája:

Első érték	Második érték	Eredmény
0	0	<b>0</b>
0	1	<b>1</b>
1	0	<b>1</b>
1	1	<b>0</b>

- A programozásban is ezek a logikai műveletek érvényesek, így a különböző logikai feltételeknél, amelyek befolyásolják majd a program futását, ezeket igen sokszor fogjuk használni.



# BOOLE ALGEBRA

- Az előzőekben látott műveletek kivétel nélkül kétoperandusos műveletek voltak, tehát minden művelethez két operandusra van szükség.
  - $Op1 @ Op2 = \text{Eredmény}$
- Azonban léteznek egyoperandusos műveletek is.
- Ilyen művelet például a NOT, mely negálja a mögötte álló operandus vagy művelet értékét.

# BOOLE ALGEBRA

- A NOT művelet igazságtáblája:

A	Nem A
1	0
0	1

- Az eddigiekben felsorolt műveleteket a programozásban a különböző feltételek megadásában fogjuk tudni jól alkalmazni.
- Hardverközeli programozásban a matematikai műveleteket ezen logikai műveletekkel lehet elvégeztetni.

# BOOLE ALGEBRA

- A logikában további fontos törvényszerűség a **DeMorgan-azonosság**:
  - $\text{NEM } (A \text{ ÉS } B) = (\text{NEM } A) \text{ VAGY } (\text{NEM } B)$
  - $\text{NEM } (A \text{ VAGY } B) = (\text{NEM } A) \text{ ÉS } (\text{NEM } B)$
- Így tehát a két oldal megegyezik, felhasználáskor az egyik átírható a másikba.
- Fontos lehet logikai feltételek egyszerűsítése során.

# BOOLE ALGEBRA

- Oldjuk meg a következő logikai formulákat:
  - $(A \vee B) \wedge (C \vee D)$ 
    - $A = 0, B = 1, C = 0, D = 0$
  - $\neg (A \vee B) \vee C \wedge D$ 
    - $A = 1, B = 0, C = 1, D = 1$
    - Próbáljuk meg felírni a fenti formula első részét a DeMorgan-azonosság felhasználásával.
  - $A \vee B \wedge C \vee D$ 
    - $A = 1, B = 1, C = 0, D = 0$

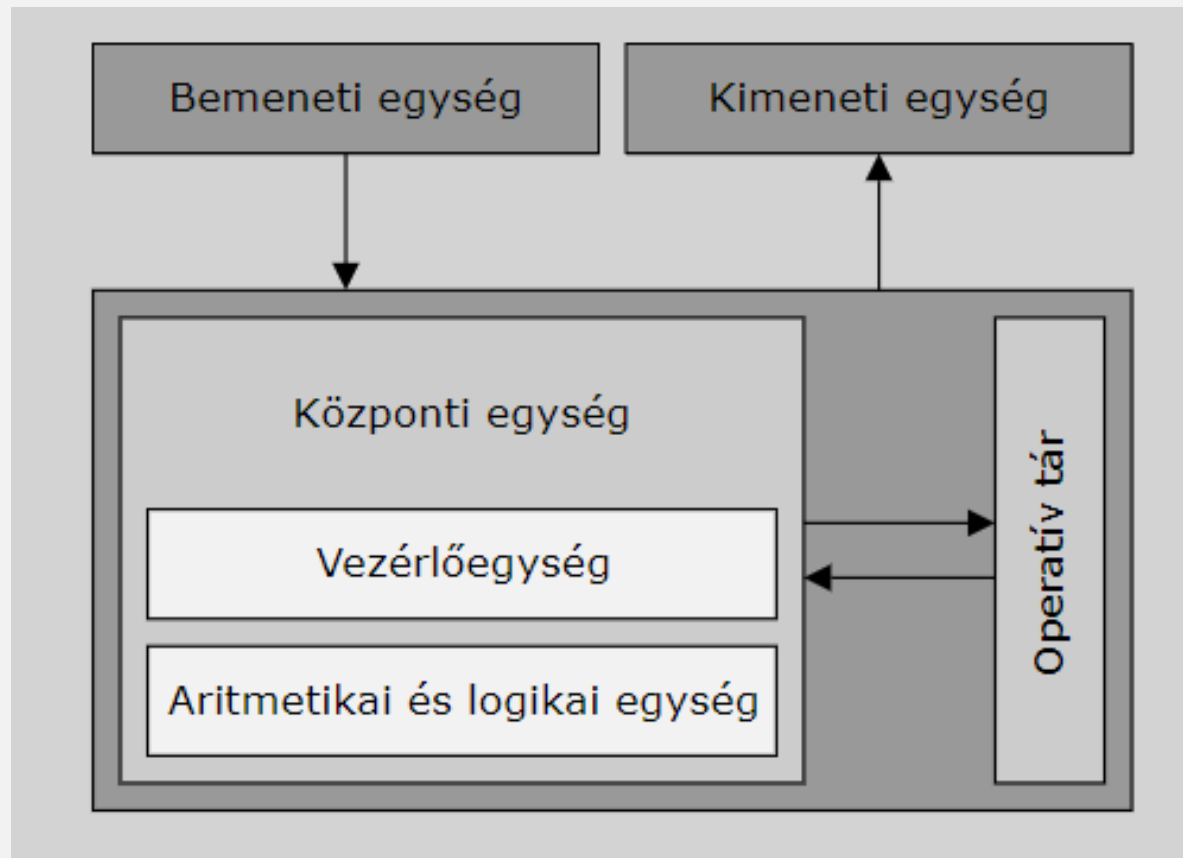
# ADATOK TÁROLÁSA

- Mivel a mai számítógépek a kettes számrendszert használják hardveres alapnak, így egyetlen kettes számrendszerbeli érték tárolását egy feltölthető cella végzi, amely vagy feltöltött vagy kiürített állapotban van.
  - Az egyetlen helyiértéket tartalmazó helyet hívjuk **bit**nek.
- A biteket oktális csoportokba szervezve kapjuk meg a **byte**-ot.
  - 8 bit = 1 byte
- Majd a byte-okat a különböző SI-prefixek alapján csoportosíthatjuk:
  - KB – Kilobyte – 1000 byte
  - MB – Megabyte – 1.000.000 byte
  - GB – Gigabyte – 1.000.000.000 byte
  - TB – Terabyte –  $10^{12}$  byte
  - PB – Petabyte –  $10^{15}$  byte
  - EB – Exabyte –  $10^{18}$  byte

# ADATOK TÁROLÁSA

- Azonban az előző prefixumok az SI-prefixek alapján  $10^x$ -t jelölik (decimális prefixumok), a számítástechnikában azonban a  $2^x$ -nel (bináris prefixumokkal) célszerűbb jelölni, emiatt egy új jelölésrendszert vezettek be:
  - KiB – Kibibyte – 1024 byte ( $2^{10}$  byte)
  - MiB – Mebibyte – 1.048.576 byte ( $2^{20}$  byte)
  - GiB – Gibibyte – 1.073.741.824 byte ( $2^{30}$  byte)
  - TiB – Tebibyte –  $2^{40}$  byte
  - PiB – Pebibyte –  $2^{50}$  byte
  - EiB – Exbibyte –  $2^{60}$  byte

# A NEUMANN-ELVŰ SZÁMÍTÓGÉP FELÉPÍTÉSE



# EGY MAI SZÁMÍTÓGÉP FELÉPÍTÉSE

