A halmaz absztrakt adattípus

Definíció: Halmaz
Halmazon adott tulajdonságú elemek, objektumok összességét, együttesét értjük. Minden elemről, objektumról egyértelműen el kell tudni dönteni, hogy a kijelölt halmaznak eleme-e, vagy sem.

A halmazokat általában latin nagy betűkkel jelöljük. A halmaz megadása formálisan, tömören $A=\{x \mid P(x) \text{ igaz}\}$ alakú, azaz az A halmaz olyan x elemekből áll, amelyekre a P állítás igaz és ebben a P állításban valamely tulajdonság van megfogalmazva. Egyszerűbb esetekben a halmazt megadhatjuk az elemeinek kapcsos zárójelek közötti felsorolásával is, amennyiben ez kivitelezhető: $A=\{a_0, a_1\ a_2 \ldots, a_n\}$.

Nevezetes számhalmazok
N a természetes számok halmaza.
Z az egész számok halmaza.
Q a racionális számok halmaza.
R a valós számok halmaza.
C a komplex számok halmaza.

A halmaz absztrakt adattípus halmazt tekint adatnak, tehát a típusban szereplő halmaz nem más, mint halmazok halmaza és a műveleteket halmazok között végezzük, amely műveleteknek az eredménye is halmaz lesz. A típus valamilyen alapelemek halmazából (univerzális-, vagy háttérhalmaz) indul ki és a típus elemei tulajdonképpen ennek a *H* háttérhalmaznak a részhalmazai. Maga a háttérhalmaz tartalmazhat véges sok, vagy végtelen sok elemet. Ha a *H*={vörös, narancs, sárga, zöld, kék, ibolya, fehér, fekete}, akkor egy nyolcelemű háttérhalmazról van szó, amely színekből tevődik össze és a szóbanforgó halmazok ennek a *H* halmaznak a részhalmazai. Ha *H*=**Z**, akkor *H megszámlálhatóan végtelen sok elem*et tartalmaz, míg a *H*=***R*** esetén *nem megszámlálhatóan végtelen sok elemű* a *H* halmaz. A halmaz valamilyen, - általában tetszőleges – elemekből épül fel. Csak annyit kell tudnunk, hogy valamely elem, objektum eleme-e a halmaznak, hozzá tartozik-e, vagy sem. Eközben figyelemmel kell lennünk a háttérhalmazra, mert csak annak az elemeit van értelme vizsgálni. Ha például *H*=**Z**, és az *A* a páros egész számok halmaza, akkor nincs értelme megkérdezni, hogy a „zöld” eleme-e *A*-nak.

Az „eleme” jelölése: $\in$, a „nem eleme” jelölése $\notin$. Például az előző esetben $2 \in A$, de $3 \notin A$.

Definíció: Üres halmaz
Azt a halmazt, amelynek egyetlen eleme sincs, üres halmaznak nevezzük. Jele $\{\}$, vagy $\varnothing$.

Definíció: Részhalmaz
Egy *A* halmaz egy *B* halmaznak *részhalmaza*, ha *A* minden eleme egyúttal *B*-nek is eleme. Jelölésben: $A \subset B$, vagy $A \subseteq B$. (Néha fordítva jelenik meg $B \supset A$, vagy $B \supseteq A$ alakban, amikor kényelmesebb azt mondani, hogy a *B* halmaz *tartalmazza* az *A* halmazt.

Az üres halmaz tetszőleges halmaznak mindig részhalmaza. Egy halmaz saját magának is mindig részhalmaza.

Definíció: Két halmaz egyenlősége
Azt mondjuk, hogy az *A* halmaz megegyezik (egyenlő) a *B* halmazzal, -

jelölésben $A=B$, - ha $A\subset B$ és $B\subset A$. Ha a két halmaz nem egyenlő, akkor annak jelölése: $A\neq B$.

Definíció: Valódi részhalmaz
Egy A halmaz egy B halmaznak *valódi részhalmaza*, ha $A\subset B$, de $A\neq B$.

Egy halmaz saját magának nem valódi részhalmaza.

Definíció: Halmazok ekvivalenciája
Az A és B halmazokat *ekvivalens*nek nevezzük, ha létezik kölcsönösen egyértelmű megfeleltetés a két halmaz elemei között. Jelölése: $A\sim B$.

Példa: A pozitív természetes számok halmaza és a természetes számok halmaza ekvivalens egymással, mert az $f(x)=x+1$ kölcsönösen egyértelmű megfeleltetés a két halmaz elemei között, ha $x\in\mathbf{N}$.

Definíció: Véges halmaz, végtelen halmaz
Egy A halmazt *véges*nek nevezünk, ha nem ekvivalens semelyik valódi részhalmazával, egyébként *végtelen halmaz*nak nevezzük.

A természetes számok halmaza végtelen halmaz, mert ekvivalens egy valódi részhalmazával, amint az előző példában láttuk.

Definíció: Megszámlálhatóan végtelen halmaz
Egy A halmazt *megszámlálhatóan végtelen*nek nevezünk, ha ekvivalens a természetes számok halmazával – $\mathbf{N}$-nel.

Definíció: Megszámlálható halmazok
A véges halmazokat és a megszámlálhatóan végtelen halmazokat *megszámlálható halmazok*nak nevezzük.

A megszámlálható halmazok elemeit tulajdonképpen fel tudjuk sorolni, sorszámozni tudjuk, a többit nem.

Definíció: A hatványhalmaz
Egy H halmaz *hatványhalmaza* – jelölésben 2^A, vagy $P(A)$ – a halmaz összes lehetséges részhalmazának a halmaza.

Például, ha $B=\{0,1\}$, akkor $2^B=\{\varnothing,\{0\},\{1\},\{0,1\}\}$.

Definíció: Véges halmaz számossága
Egy A véges halmaz számosságának nevezzük az A elemeinek a számát. Jelölése $|A|$, vagy Card(A).

Tétel: Véges halmaz hatványhalmazának számossága
Egy A véges halmaz hatványhalmazának a számossága: $|2^A|=2^{|A|}$.

Bizonyítás: Soroljuk fel a halmaz elemeit és rögzítsük ezt a felsorolást. Ekkor minden részhalmaz felírható oly módon, hogy a felsorolásban azon elemek helyére, amelyek a részhalmazban benne vannak 1-et írunk, amelyek nincsenek, oda 0-t írunk. minden elem helyére kétféle jelet tehetünk. Az ilyen jelsorozatok száma pedig akkor $2^{|A|}$. ■.

Definíció: **Az $\aleph_0$ (alef null) számosság**
A természetes számok és minden vele ekvivalens halmaz számosságát alef null számosságnak nevezzük. Jele $\aleph_0$**.** Tehát $|\boldsymbol{N}|=\aleph_0$**.**

A halmaz absztrakt adattípus tehát áll egy *H* háttérhalmazból és annak valamely részhalmazainak a rendszeréből úgy, hogy a halmazműveletek elvégezhetők legyenek, azaz minden művelet eredménye ezen rendszer valamely tagja legyen. (Nem kötelezően kell az összes részhalmaznak ebben a rendszerben szerepelnie, de gyakran kényelmi szempontok miatt ezt alkalmazzuk. Ez a „gazdag" rendszer nem mindig jár előnyökkel a végtelen halmazoknál.) Alább megadjuk a legfontosabb műveleteket.

Unáris művelet a *komplemens* képzés. Azt mondjuk, hogy a *B* halmaz ($B \subset H$) az *A* halmaz komplemense, jelölésben $B = \overline{A}$, ha minden esetben mikor egy elem *A*-nak nem eleme, akkor viszont *B*-nek eleme.

Bináris műveletek az alábbiak.
Unió (egyesítés). Az *A* és *B* halmaz uniója, jelölésben $A \cup B$, az a halmaz, amelynek elemei mindazok az elemek, amelyek a két halmaz közül legalább az egyikhez hozzá tartoznak.

Metszet (közös rész). Az *A* és *B* halmaz metszete, jelölésben $A \cap B$, az a halmaz, amelynek elemei mindazok az elemek, amelyek a két halmaz közül mindkettőhöz hozzá tartoznak.

Különbség. Az *A* és *B* halmaz különbsége, jelölésben $A \setminus B$, az a halmaz, amelynek elemei mindazok az elemek, amelyek *A*-hoz hozzá tartoznak, de *B*-hez nem.

Szimmetrikus különbség. Az *A* és *B* halmaz szimmetrikus különbsége, jelölésben $A \Delta B$, az a halmaz, amelynek elemei mindazok az elemek, amelyek *A*-hoz, vagy *B*-hez hozzá tartoznak, de mindkettőhöz nem.

A különbség és a szimmetrikus különbség az első három művelettel felírható a következő módon: $A \setminus B = A \cap \overline{B}$ és $A \Delta B = \left(\overline{A} \cap B\right) \cup \left(A \cap \overline{B}\right)$. (Ellenőrizzük le!) Vegyük észre, hogy $(A \cap B) \subset A$ és $(A \cap B) \subset B$!

A komplemens, metszet és unió műveletek tulajdonságai:

1.	Komplemens komplemense	$\overline{\overline{A}} = A$	
2.	Kommutativitás	$A \cup B = B \cup A$	$A \cap B = B \cap A$
3.	Asszociativitás	$(A \cup B) \cup C = A \cup (B \cup C)$	$(A \cap B) \cap C = A \cap (B \cap C)$
4.	Disztributivitás	$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
5.	Idempotencia	$A \cup A = A$	$A \cap A = A$
6.	A háttérhalmaz és az üres halmaz hatása	$A \cup H = H$ $A \cup \varnothing = A$	$A \cap H = A$ $A \cap \varnothing = \varnothing$
7.	Elnyelés	$A \cup (A \cap B) = A$	$A \cap (A \cup B) = A$
8.	Komplemens a metszetben		$A \cap \overline{A} = \varnothing$
9.	Komplemens az unióban	$A \cup \overline{A} = H$	
10.	De Morgan szabály	$\overline{A \cup B} = \overline{A} \cap \overline{B}$	$\overline{A \cap B} = \overline{A} \cup \overline{B}$

Láthatjuk, hogy ezek a tulajdonságok erősen emlékeztetnek a logikai absztrakt adattípus műveleteinek a tulajdonságaira. Nem véletlen, mert a logikai absztrakt adattípus és a halmaz adattípus a *Boole adattípus* speciális esetei.

Definíció: Diszjunkt halmazok
Az A és B halmazokat diszjunktaknak nevezzük, ha nincs közös elemük, azaz, ha $A \cap B = \varnothing$.

Nyilvánvalóan az A és $\overline{A}$ halmazok diszjunktak. Vegyük észre, hogy $A=(A\backslash B)\cup B$, ahol az $A\backslash B$ és B halmazok diszjunktak.

Tétel: Véges halmaz számosságának tulajdonságai

1. Ha A véges halmaz, akkor $|A| \geq 0$, egyenlőséggel akkor és csak akkor, ha $A = \varnothing$.
2. Ha A és B véges halmazok diszjunktak, akkor $|A \cup B| = |A| + |B|$.
3. Ha A és B tetszőleges véges halmazok, akkor $|A \cup B| = |A| + |B| - |A \cap B|$.

Bizonyítás: Az olvasóra bízzuk. ■

Következmény: Véges halmazokra $|A| = |A \setminus B| + |B|$, és ha $A \subset B$, akkor $|A| \leq |B|$. (Lássuk be!)

A halmaz adatstruktúra

Többféle módon lehet megpróbálni a halmaz absztrakt adattípust megjeleníteni, reprezentálni. Egy lehetőség a Venn-diagramok használata, amikor a halmazt valamilyen geometriai alakzattal (általában kör) szimbolizáljuk és ezek külseje, egyesítése, metszete, különbsége stb. szemlélteti az egyes műveleteket. Véges halmazra, nem túl nagy számosság esetén akár apróbb tárgyakkal, mint elemeikkel is szemléltethetjük az egyes halmazokat.

A halmaz absztrakt adattípus implementálása

A végtelen halmazok implementálása általánosan nem megoldott, egyes speciális esetekben van mód rá, amikor szimbolikus számításokat végzünk a számítógéppel. Ez azonban messze nem meríti ki az összes lehetőséget. Véges, nem túl nagy elemszám esetén az alábbiakban leírt módon járhatunk el, ahogyan egyes programozási nyelvek meg is teszik.

A háttérhalmaz elemeinek rögzítjük egy felsorolását. Ebben a lineáris sorrendben minden elemhez egy bitet rendelünk hozzá, ami által egy bit *n*-est kapunk, ha *n* a háttérhalmaz elemeinek a száma. A háttérhalmaz egy részhalmazát (ezt használjuk halmazként) egy olyan bit *n*-nessel írjuk le, amelyben egyesek vannak annál az elemnél, amely a halmazban szerepel, és zérusok ott, amelyik nem szerepel. Tehát minden halmazt bit *n*-essel implementálunk. Speciálisan az üres halmaz egy tiszta zérusokból álló bit *n*-es. A halmazműveletek implementálása ezek után egyszerűen történhet a biteken végzett logikai műveletek révén. A komplemensképzés bitenkénti negáció, az unió bitenkénti diszjunkció, a metszet bitenkénti konjunkció. (Rájövünk, hogy a különbségképzés az implikáció negáltja, a szimmetrikus különbség pedig a kizáró vagy révén valósítható meg.)

Például legyen a háttérhalmaz a fentebb említett színek halmaza *H*={vörös, narancs, sárga, zöld, kék, ibolya, fehér, fekete}, egy nyolcelemű halmaz (az egyszerűség kedvéért, hogy egy byte-on elférjen egy halmaz.) Ezt a *H* háttérhalmazt egy

	vörös	narancs	sárga	zöld	kék	ibolya	fehér	fekete
H	1	1	1	1	1	1	1	1

byte formájában tárolhatjuk, ahol az egyes bitek balról jobbra jelentik, hogy a halmaz felépítésében mely elemek vesznek részt. Egyetlen halmazelem szintén ebben a formában adható meg, mint egy egyelemű részhalmaz. Legyen most *A*={narancs, sárga, zöld, ibolya, fehér} és *B*={vörös, narancs, sárga, kék, fekete}. Képezzük a következő halmazokat: $\overline{A}$, $A \cup B$, $A \cap B$, $A \backslash B$, $A \Delta B$! Legyen az *x* elem a „zöld” szín. Állapítsuk meg, hogy $x \in (A \Delta B)$ fennáll-e!

	vörös	narancs	sárga	zöld	kék	ibolya	fehér	fekete
A	0	1	1	1	0	1	1	0
$\overline{A}$	1	0	0	0	1	0	0	1

	vörös	narancs	sárga	zöld	kék	ibolya	fehér	fekete
A	0	1	1	1	0	1	1	0
B	1	1	1	0	1	0	0	1
$A \cup B$	1	1	1	1	1	1	1	1
$A \cap B$	0	1	1	0	0	0	0	0
$A \backslash B$	0	0	0	1	0	1	1	0
$A \Delta B$	1	0	0	1	1	1	1	1

Az *x* elem, mint egyelemű halmaz adható meg. Az, hogy eleme-e egy adott halmaznak, egy ÉS művelettel meghatározható. Amennyiben a művelet eredménye tiszta zérus, akkor nem eleme, ha nem zérus, akkor eleme.

	vörös	narancs	sárga	zöld	kék	ibolya	fehér	fekete
x	0	0	0	1	0	0	0	0
$A \Delta B$	1	0	0	1	1	1	1	1
$x \cap (A \Delta B)$	0	0	0	1	0	0	0	0

Az eredmény nemzérus (nem üres halmaz), tehát a zöld szín benne van a szimmetrikus különbségben.

FELADATOK

1. a. Bizonyítsa be, hogy véges halmaz esetében $|A \cup B| = |A| + |B| - |A \cap B|$!
   b. Ha $|H|=70$, A⊂H, B⊂H, $|A|=50$, $|B|=30$, akkor milyen legszorosabb határok között lehet $|A \cup B|$ és $|A \cap B|$?
2. a. Véges halmazok esetére fejezze ki $|A \cup B \cup C|$ értékét $|A|$, $|B|$, $|C|$, $|A \cap B|$, $|A \cap C|$, $|B \cap C|$, $|A \cap B \cap C|$-vel!
   b. Egy százötvenfős évfolyamon 100-an beszélnek angolul, 70-en németül és 49-en franciául. Nincs senki, aki ne beszélné legalább az egyik nyelvet. Mindhárom nyelvet 10-en beszélik. Hányan beszélik csak a nyelvek egyikét, és hányan pontosan kettőt?
3. a. Bizonyítsa be, hogy a természetes számok halmaza $\boldsymbol{N}$ és az egész számok halmaza $\boldsymbol{Z}$ ekvivalens halmazok!
   b. Legyen $A=\boldsymbol{N}$, a természetes számok halmaza! Legyen $B \subset A$, úgy hogy $B=\{0, 0+1, 0+1+2, 0+1+2+3, 0+1+2+3+4, \ldots\}$! Bizonyítsa be, hogy $A \sim B$!
4. Bizonyítsa be, az alábbi azonosságokat!
   a. $\overline{A \setminus B} \cap \left(\overline{A} \cup \overline{B}\right) = \overline{A}$
   b. $A \cap \left(\overline{A \setminus B}\right) = A \cap B$
5. a. Bizonyítsa be, hogy két megszámlálhatóan végtelen halmaz uniója is megszámlálhatóan végtelen halmaz!
   b. Bizonyítsa be, hogy véges sok megszámlálhatóan végtelen halmaz uniója is megszámlálhatóan végtelen halmaz!
   c. Bizonyítsa be, hogy megszámlálhatóan végtelen sok megszámlálhatóan végtelen halmaz uniója is megszámlálhatóan végtelen halmaz!
6. Legyen $H=\boldsymbol{R}$, A=(-1; 2], B=[1,4) félig zárt, félig nyitott intervallumok! Határozza meg az $A \cup B$, $A \cap B$, $A \setminus B$, $B \setminus A$, $A \Delta B$ halmazokat és komplemenseiket!

A karakter absztrakt adattípus

A karakter absztrakt adattípus szöveges információ kezelését, megjelenítését teszi lehetővé. A szövegeinket elemi egységekből építjük fel.

Definíció: Karakter
A karakter a tágabb értelemben szövegesen lejegyzett adat legkisebb, elemi egysége, egy tovább már nem bontható szimbólum.

A karakterek halmazát *X*-szel fogjuk jelölni. A karakter absztrakt adattípus T=(*X*,M) formában adható meg, ahol az M az *X*-en végezhető műveletek halmaza. Azt, hogy *X* pontosan milyen szimbólumokat is tartalmaz, egyelőre nyitva hagyjuk.

A karakterek *X* halmazát valahogyan le kell írni, össze kell az elemeket gyűjteni, rendezni kell őket, valamilyen sorrendben fel kell sorolni. A karaktereket osztályozhatjuk jellegük szerint. Az informatikában *X* véges halmaz, és elemei lehetnek számjegyek, betűk, írásjelek (pont, vessző, felkiáltó jel, stb.), vezérlőjelek (csengő, soremelés, lapdobás, kocsi vissza, file vége, stb.). A rendezettség, felsorolás megállapodáson, konvención alapul. A sorrendet rögzítjük és nevezhetjük ábécé sorrendnek. A sorrendnek megfelelően az egyes karaktereket sorszámmal látjuk el, amit zérustól indítunk és egyesével növelünk. A rendezettségi sorban az egymáshoz képesti elhelyezkedés vizsgálatához bevezethetünk két bináris műveletet, az egyik a „Hátrább álló", a másik lehet az „Előbb álló" művelet. Két karakter közül értelemszerűen az ábécé sorrendnek megfelelően hátrább, illetve az előrébb állót adják eredményül. Műveleti jelük legyen rendre $\triangleright, \triangleleft$. Például $'K'\triangleright'C'='K'$ és $'K'\triangleleft'C'='C'$ a magyar ábécében. (Milyen tulajdonságot tudunk kimutatni ezekre a műveletekre? Teljesül-e például a kommutativitás, asszociativitás, disztributivitás és egyéb tulajdonság?)

Civilizációnkban rendkívül sok karakterrel találkozhatunk. Az informatika kezdetben nem sokat használt fel ezek közül. Szorított a tárolóhely hiánya. Jelentős lépés volt, amikor az akkor legfontosabbnak tekintett jelkészletet egységesítették, szabványosították. Ez volt az ASCII kódtáblázat (American Standard Code for Information Interchange, az információcsere amerikai szabványos kódja). Ez a szabvány már nem csak definiálta a karakterek halmazát, hanem az implementációt is megadta. A karakterekhez tartozó sorszámot kódnak nevezzük és az implementáció ezen kódok kettes számrendszerbeli alakja hét biten elhelyezve, tehát a kód hétbites. Zérustól 127-ig terjednek a kódok, az első 32 kód (0-31) és a 127-es vezérlőjel, a többi látható, nyomtatható jel. Vezérlőjel például a csengő (7), a soremelés (10), a lapdobás (12), a kocsi vissza (13), a file vége (26), az escape (27) stb. A láthatók, a nyomtathatók között vannak az angol ábécé nagy- és kisbetűi ((A-Z), (a-z) azaz (65-90), (97-122)). A kisbetűk kódjai 32-vel nagyobbak a megfelelő nagybetű kódjánál. A betűk ábécé sorrendben követik egymást. A 0,1,2,…,9 decimális számjegyek kódjai 48,49,…,57. A helyköz (space) kódja 32. A teljes ASCII táblázat a fejezet végén található.

Az ASCII karakterek, kódok tárolása a byte-os memóriában értelemszerűen az egy byte egy karakter elvet követte. Mivel egy byte-on 256 féle bitmintázat helyezhető el, ezért kihasználatlan maradt a 127-es kód feletti 128-255 számtartomány 128 eleme. Az informatika nemzetközivé válása és az egyre újabb és újabb területekre való bevezetése által szükségessé vált újabb szakterületek jeleinek, nemzeti ábécék jeleinek a bevezetése is. Ezt kezdetben a szabad 128 hely kitöltésével igyekeztek megoldani, amiről hamar kiderült, hogy zsákutca, mivel jóval több jelre van szükség. Sajnos a bővítés első lépései rossz irányban történtek. Megtartva a byte-os szerkezetet, bevezették a kódlap fogalmát és a felső 128 kódot a szerint kezdték értelmezni, hogy melyik kódlapot tekintették érvényesnek egy adott file (szöveg)

esetében. Definiálták például a Latin-1 kódlapot, amelybe sok ékezetes karakter is belefért. Ez azonban semmibe vette az ékezetes betűk ábécé sorrendjét. (A magyar ő és ű ide sem fért bele, ezek a Latin-2 kódlapra kerültek. Az eredmény az lett, hogy ha a megjelenítő program nem tudta (és honnan tudta volna), hogy a szöveg milyen kódlap alapján készült, akkor amennyiben ő más kódlapra volt beállítva, a szöveg olvashatatlanná vált. Ugyanannak a kódnak többféle karakter is megjelent aszerint, hogy melyik kódlapot választottuk. Vannak azonban olyan nyelvek, amelyeknél az írásjelek, szimbólumok száma több ezer. Ezek elhelyezése eleve reménytelen egy kódlapon. Megkísérelték ezeket kétbyte-os ábécében elhelyezni. Egy szöveg azonban manapság már nem biztos, hogy végig egyetlen nyelven készül, ezért a probléma megoldatlan maradt a próbált úton. A megoldást a Unicode adja.

A Unicode

A Unicode Standard 1.0 verzió 1991-ben jelent meg, jelenleg (2011) a 6.0 verziónál tart. A Unicode rendszerben a karakter absztrakt fogalom és a neve egyedileg azonosítja, amely nem változtatható. A név mellett a karaktert egy rá egyedileg jellemző egész számmal (a karakter kódpontjával (code point) kapcsoljuk össze. Ezek a kódpontok egy kódtérben (codespace) helyezkednek el, amely nevezetesen a nullával kezdődő és a hexadecimális 10FFFF-fel végződő egész számok tartománya. A kódpont mindig ugyanazt a karaktert jelöli és fordítva, egy karakternek mindig ugyanaz a kódpontja. A kódtér mérete 1 114 112 kódpont. Majdnem mindet karakterkódként használják. Vannak speciális célra fenntartott kódpont tartományok.

A számunkra (Európa) szokványos karakterek többsége az első 65 536 kódponton elfér. Ezt a tartományt BMP-nek (Basic Multilingual Plane) nevezik. A maradék több mint 1 millió hely elegendő az összes ismert karakter, írásjel kódolására, beleértve a minoritások írásképeit vagy a történelemből ismert írásképeket. Lényeges tulajdonsága a Unicode-nak, hogy a karakter absztrakt fogalmát leválasztja annak a képernyőn vagy a nyomtatón megjelenő képétől. A képpel a Unicode nem foglalkozik. (Pl. a karakter mérete, színe, dőlése, kövérsége, kiemelt mivolta, alakja stb.)

A karakter Unicode kódját az U+xxxx, U+xxxxx vagy a U+xxxxxx alakban adjuk meg, ahol x hexadecimális számjegy és a számérték adja kódpontot. Az absztakt karakter reprezentálására a számítógépben (implementáció) három lehetőségünk van. Ezek a UTF-32, a UTF-16 és a UTF-8 kódolási formák. (UTF - Unicode Transformation Format.) Meg kell különböztetni a *kódolt karakter* és a *szövegelem* fogalmát. A szövegelem egy vagy több kódolt karaktert jelent. Például a hullámos vonallal felül díszített u betű (ũ) előáll az *u betű* és a *felül hullámos vonal* karakterek kódjából. Tehát a karakterkép esetleg több elemből is összetevődhet.

Tekintsük most az egyes kódolási formákat, a Unicode különböző implementációit. Bármely formát is választjuk, ezek egymásba veszteség nélkül áttranszformálhatók.

A **UTF-32** esetén a kódpont szerinti egész szám és a kettes számrendszerbeli 32 bites megfelelője között egy-egy értelmű a kapcsolat. A kód mérete fix, négy byte. Emiatt a tárigény nagy. A 00 00 00 80 … 00 00 00 FF és a 00 00 D8 00 … 00 00 DF FF kódpont tartomány érvénytelennek számít.

A **UTF-16** esetén a U+0000, ..., U+FFFF kódpontok 16 biten jelennek meg, míg a U+10000, ..., U+10FFFF kódpontok egy 16 bites páron ábrázolódnak a következő módon. Az egyforma betűk nem jelentenek szükségképpen azonos biteket.

	aaaaaaaa	bbbbbbbb	→	aaaaaaaa	bbbbbbbb		
000uuuuu	aaaaaabb	cccccccc	→	110110ww	wwaaaaaa	110111bb	cccccccc

Itt a wwww négyjegyű bináris szám az uuuuu-1 bináris számot jelenti.
Látható, hogy ez a kódolás BMP-re optimalizált. (A Unicode elődje.)

A **UTF-8** byte-orientált kódolás, ASCII alapú, az ASCII-re transzparens, azaz ugyanúgy ábrázolja. A kód mérete változó, 1-4 byte lehet. A vezető byte jelzi a karaktert leíró byte-szakaszt. Kedvező a HTML és Internetes protokollok számára. A kódpont ábrázolásának módja a következő.

	00000000	0xxxxxxx	→	0xxxxxxx			
	00000yyy	yyxxxxxx	→	110yyyyy	10xxxxxx		
	zzzzyyyy	yyxxxxxx	→	1110zzzz	10yyyyyy	10xxxxxx	
000uuuuu	zzzzyyyy	yyxxxxxx	→	11110uuu	10uuzzzz	10yyyyyy	10xxxxxx

Példák transzformációra.

	UTF-32	UTF-16	UTF-8
U+004D	00 00 00 4D	00 4D	4D
U+0430	00 00 04 30	04 30	D0 B0
U+4E8C	00 00 4E 8C	4E 8C	E4 BA 8C
U+10302	00 01 03 02	D8 00 DF 02	F0 90 8C 82

U+004D				
UTF-32	00	00	00	4D
	0000 0000	0000 0000	0000 0000	0100 1101
UTF-16			00	4D
			0000 0000	0100 1101
UTF-8				4D
				0100 1101

U+004D				
UTF-32	00	00	00	4D
	0000 0000	0000 0000	0000 0000	0100 1101
UTF-16			00	4D
			0000 0000	0100 1101
UTF-8				4D
				0100 1101

U+004D				
UTF-32	00	00	00	4D
	0000 0000	0000 0000	0000 0000	0100 1101
UTF-16			00	4D
			0000 0000	0100 1101
UTF-8				4D
				0100 1101

A további információkért utalunk a Unicode Standard leírására:

The Unicode Standard / the Unicode Consortium; edited by Julie D. Allen ... [et al.] Version 6.0. ISBN 978-1-936213-01-6 (http://www.unicode.org/versions/Unicode6.0.0/)

FELADATOK

1. Ellenőrizze, hogy teljesül-e a kommutativitási, asszociativitási, disztributivitási tulajdonság a fentebb bevezetett $\triangleright$ és $\triangleleft$ műveletekre!

2. a. Egymást követő byte-okon hexadecimálisan a következőt találjuk: 41 53 43 49 49. ASCII karaktereket feltételezve milyen szöveget tartalmaznak a byte-ok? Változtassa át a kisbetű-nagybetű módot ellentétesre! Milyen byte-okat kapott?
   b. Adja meg az ASCII byte-ok tartalmát hexadecimálisan is és decimálisan is, ha a tárolandó szöveg: ’To be or not to be, that is the question!’

3. Adottak az U+10FA78, U+93DB, U+0034, U+07FE Unicode kódpontok. Készítse el a UTF-32, UTF-16 és UTF-8 kódolásban megjelenő byte sorozatot!

ASCII kódtábla

Vezérlő jelek

Hexa	Decimális	Karakter	Jelentés	Hexa	Decimális	Karakter	Jelentés
00	0	NUL	NULL	10	16	DLE	Data link escape
01	1	SOH	Start of heading	11	17	DC1	Device control 1
02	2	STX	Start of text	12	18	DC2	Device control 2
03	3	ETX	End of text	13	19	DC3	Device control 3
04	4	EOT	End of transmission	14	20	DC4	Device control 4
05	5	ENQ	Enquiry	15	21	NAK	Negative acknowledgement
06	6	ACK	Acknowledgement	16	22	SZN	Synchronous idle
07	7	BEL	Bell	17	23	ETB	End of transmis-sion block
08	8	BS	Backspace	18	24	CAN	Cancel
09	9	HT	Horizontal tab	19	25	EM	End of medium
0A	10	LF	Line feed	1A	26	SUB	Substitute
0B	11	VT	Vertical tab	1B	27	ESC	Escape
0C	12	FF	Form feed	1C	28	FS	File separator
0D	13	CR	Carriage return	1D	29	GS	Group separator
0E	14	SO	Shift out	1E	30	RS	Record separator
0F	15	SI	Shift in	1F	31	US	Unit separator
				7F	127	DEL	Delete

Nyomtatható karakterek

Hexa	Dec	Karakter	Hexa	Dec	Karakter	Hexa	Dec	Karakter
20	32	Space	40	64	@	60	96	`
21	33	!	41	65	A	61	97	a
22	34	"	42	66	B	62	98	b
23	35	#	43	67	C	63	99	c
24	36	$	44	68	D	64	100	d
25	37	%	45	69	E	65	101	e
26	38	&	46	70	F	66	102	f
27	39	'	47	71	G	67	103	g
28	40	(	48	72	H	68	104	h
29	41	)	49	73	I	69	105	i
2A	42	*	4A	74	J	6A	106	j
2B	43	+	4B	75	K	6B	107	k
2C	44	,	4C	76	L	6C	108	l
2D	45	-	4D	77	M	6D	109	m
2E	46	.	4E	78	N	6E	110	n
2F	47	/	4F	79	O	6F	111	o
30	48	0	50	80	P	70	112	p
31	49	1	51	81	Q	71	113	q
32	50	2	52	82	R	72	114	r
33	51	3	53	83	S	73	115	s
34	52	4	54	84	T	74	116	t
35	53	5	55	85	U	75	117	u
36	54	6	56	86	V	76	118	v
37	55	7	57	87	W	77	119	w
38	56	8	58	88	X	78	120	x
39	57	9	59	89	Y	79	121	y
3A	58	:	5A	90	Z	7A	122	z
3B	59	;	5B	91	[	7B	123	{
3C	60	<	5C	92	\	7C	124	|
3D	61	=	5D	93	]	7D	125	}
3E	62	>	5E	94	^	7E	126	~
3F	63	?	5F	95	_			