A programozási nyelvek elemei (Lexikális elemek)

- A programok fordítási egységei lexikális elemek sorozatai.
- A lexikális elemek karaktersorozatok, határoló jelekkel elválasztva.
- A nyelvekben általában megkülönböztetjük
  a grafikus karaktereket (a Latin-1 vagy az angol abc betűi, számjegyek és speciális karakterek - pl. " # & ' ( ) , * - + / : ; < = _ [ ] { } | .) --),
  a formátum vezérlő karaktereket (az ISO 6429 szabványban: character tabulation (HT), line tabulation (VT), carriage return (CR), line feed (LF), and form feed (FF)), és
  az egyéb (implementáció függő) vezérlő karaktereket.

Lexikális elem:

- azonosító,
- numerikus literál,
- karakter literál,
- string literál,
- határoló,
- megjegyzés

A jelkészlet

A **karakterkészlet** különböző karakterek egy halmaza.

- Nem teszünk fel semmit a számítógép belső karakterábrázolásáról. Még rendezést sem tételezünk fel a karakterek között,
- Megadjuk a karakterek neveit és a karakter megjelenítését látható formában.
- Tartalmazhat olyan karaktereket, amelyek bizonyos megjelenítéseknél ugyanúgy néznek ki, mégis logikailag különbözőnek tekintjük.

  Például: a latin nagy „A”, a cirill nagy „A” és a görög nagybetűs Alfa „A”

Például:

EXCLAMATION !

QUESTION_MARK ?

SEMICOLON ;

A **karakterkód** egy leképezés, amely kölcsönösen egyértelmű megfeleltetést ad a karakterkészlet karakterei és a nemnegatív egészek egy halmaza között.

Egy egyedi számkódot, egy kódpozíciót rendel a karakterkészlet minden karakteréhez. (Gyakran táblázat)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.asciitable.com

A **karakterkódolás** egy **algoritmus** karakterek digitális formában történő megadására. Ez a karakterek kódszámainak sorozatait „oktetek” sorozatára képezi le.

Például az ISO 10646 karakterkódban az

'a' az 'ä' és az ‰ (ezrelék jel)

karaktereknek megfelelő számkódok

a 97, a 228 és a 8240.

A karakterkódolás adja meg, hogy a karakterkódokat hogyan képezzük le oktetek sorozatára.

Így például az ISO 10646 egy lehetséges kódolásában minden karakter kódolásához két oktetet használva a kódolási algoritmus ezeknek

a (0, 97), a (0, 228) és a (32,48)

oktetpárokat felelteti meg.

Milyen karakterek használhatók a nyelvben?

- **Pascal (ISO 7185:1990):**
  - **csak minimum követelmények, pl.: '0'..'9' rendezett és folytonos kell legyen, 'A'..'Z' rendezett, de nem feltétlenül folytonos, 'a'..'z' nem kötelező, de ha van, akkor mint a nagybetűk.**
  - **Nincs előírva a lehetséges karakterek halmaza, különböző implementációk különböző kódolást használhatnak(!) (pl. EBCDIC vagy ASCII) => különböző implementációk másképpen viselkedhetnek (pl. '}' ∈ 'A'..'Z' EBCDIC-ben és '}' ∉ 'A'..'Z' ASCII-ben, vagy 'A' < '1' EBCDIC-ben és '1' < 'A' ASCII-ben).**
  - **=> ez nem tesz jót a programok hordozhatóságának ☹**

- **C++(**IISO/IEC 14882:2011 **):**
  - **Az alap karakterkészlet 96 elemű: a space, a horizontal tab, vertical tab, new-line, form feed vezérlő karakterei, és a következő 91 grafikus karakter: 'a'..'z' 'A'..'Z' '0'..'9'**
    " # & ' ( ) , * - + / : ; < > = _ [ ]
    **{ } | . % ? ^ ~ ! **
  - **ISO 10646 szerint kódolva**
  - **univerzális karakterek elvben megengedettek** \UNNNNNNNN a neve, annak, aminek a rövid neve az ISO/IEC 10646 NNNNNNNN, ill. \uNNNN annak, aminek 0000NNNN, ez a gyakorlatban fordítófüggő, hogy működik-e
  - **=> ez precízebb, mint az 1990-es, ahol nem volt a kód előírva.**

- **CLU, Eiffel:**
  - **ASCII kód a megengedett.**
- **ADA (ISO/IEC 8652:2012(E)):**
  - **A teljes ISO/IEC 10646:2011**
- **Java:**
  - **"Java programs are written using the Unicode character set”**
- **C#:**
  - **„A source file is an ordered sequence of Unicode characters.”**

Melyek az elhatároló jelek?

- **A legtöbb programozási nyelvben a helyköz (space), a tab, a sorvége elhatároló jel.**
- **Van sok <u>egy</u> karakterből álló elhatároló jel, mint pl.:**

Pascal: & ' ( ) + - * / : . ; < = >

C++: & % ^ ' ( ) + - { } | ~ [ ] \ " * / : . ; , < = > ! ?

Java: ( ) { } [ ] ; , . + - * / : < = > ! ~ ? : & | ^ %

ADA: & ' ( ) * + , – . / : ; < = > | stb.

Melyek az elhatároló jelek?

- **Van számos több karakterből álló elhatároló jel is, mint pl. :**

Pascal: := >= <= <> << ** ..

C++: -> ++ -- .* ->* == <= >= && || << >> <<= >>= != .. += -= &= ^= |= :: *= /=

Java: == <= >= << != && || ++ -- >> >>> += -= *= /= &= |= ^= %= <<= >>= >>>=

ADA: => .. ** := /= >= <= << >> <> -- stb.

Megkülönbözteti-e a nyelv a kis- és nagybetűket?

Kutya = kutya = KUTYA = KuTya = …?

- **Pascal, Fortran, CLU, ADA...**: ekvivalensek
- **C, C++, Modula, Perl, Java, C#,…**: minden betű különböző
- **Eiffel**: a kis- és a nagybetűk ekvivalensek, de használhatjuk ugyanazt a nevet egy objektumra és a típusára
  - pl. : a: A megengedett

Azonosítók

- Milyen karakterek használhatóak az azonosítók leírására?
- Mi az azonosító megengedett szintaxisa?
- Van-e hosszúsági megkötés az azonosítókra?
- Vannak-e kötött szavak?
  Van-e különbség a kulcsszavak és az előre definiált szavak között?

Milyen karakterek használhatóak azonosítókban?

- **Pascal** (ISO 7185:1990):
  - betűk ('A'..'Z’, 'a'..'z') és számjegyek ('0'..'9')
- **C++**(ISO/IEC 14882:2003):
  - betűk ('A'..'Z’, 'a'..'z'), számjegyek ('0'..'9') és '_’
- **ADA**
  - Az 1995-ös szabványban (ISO/IEC 8652:1995) még: az ISO 10646 BMP Row 00 bármely karaktere, amelynek a neve “Latin Capital Letter” és “Latin Small Letter” (ezek az ún. **identifier_letters)**, számjegyek ('0'..'9') és '_‘
  - új szabvány (Ada2012): minden, ami betű, számjegy! (Πλάτων, Чайковский stb. is)

Milyen karakterek használhatóak azonosítókban?

- **Java**:
  - A **Unicode** betűi és számjegyei, a '$’és '_'
- **CLU, Eiffel**:
  - betűk ('A'..'Z’, 'a'..'z'), számjegyek ('0'..'9') és '_’
- **Perl**:
  - az előzőeken túl $, @ és % jellel is kezdődhetnek

    $ a skalárokat, @ számmal indexelt tömböt, % asszociatív tömböt jelöl.

Milyen karakterek használhatóak azonosítókban?

- **C#:**

Az azonosítókat a Unicode szabvány ajánlásának megfelelően kell írni.

Érdekesség, hogy a kulcsszavak az @ bevezető jellel használhatóak azonosítóként is (például: @bool). Ez a jel azonosítót nem vezethet be.

Ezt a lehetőséget a programok hordozhatósága, illetve "átjárhatósága" miatt vezették be. Az @ prefixű szimbólumok neve valójában @ nélkül kerül fordításra. Így attól függetlenül, hogy egy másik nyelv nem tartalmazza például a ‘sealed’ kulcsszót, és egy programban azonosítóként használják, C#-ban lehetőség van az azonosító közvetlen használatára.

Mi az azonosítók megengedett szintaxisa?

- A leggyakrabban:

  letter { letter | digit }

  néha az '_' beszúrása is megengedett.
- **Pascal**: letter{letter|digit}
- **ADA:** letter{[_]letter|[_]digit},
  - itt a betűk halmaza nagyobb, minden, aminek a nevében szerepel az, hogy „letter”
- **Java:** Java_letter{[_]Java_letter|[_]digit},
  - αβγδε vagy El_Ňiňo is OK.

Van-e hosszúsági megkötés az azonosítókra?

- **Fortran66, Fortran77**: max. 6 jel
- **Fortran90**: max. 31 jel
- **ADA**: be kell férjen egy sorba…
- **Pascal, Java, …**: tetszőleges hosszúság

Vannak-e fenntartott szavak?

- **Pascal, C, C++, Java, Perl, CLU, ADA** etc.:
  - a fenntartott szavak nem használhatók azonosítóként
- **ADA:**
  - különbséget tesz a kulcsszavak és az előredefiniált szavak között, mint pl. : Integer, True, etc..
  - Átdefiniálhatóak:

  type Integer is range -999_999..+999_999;

  és így a program implementáció-független lesz

  de: True: Integer … -- nincs értelme…

Fortran, PL/1:

megengedett a kulcsszavakat azonosítóként használni! ☹

Literálok

- Milyen numerikus literálok vannak?
- Milyen más alapok megengedettek a 10-esen kívül?
- Mi az egész, ill. valós literálok szintaxisa?
- Többsoros sztring literálok megengedettek-e?
- Vezérlőkarakterek sztring literálokban megengedettek-e, és hogyan írjuk le őket?

Numerikus literálok:

- A gyakorlatban csak a '0'..'9’ és az 'A'..'F'
- néha '_'
- ' ' (space) soha :-(

Mi a számok szerkezete?

- **Egészek**: decimális: [-]digit{digit} -123, 456789
  pl. **ADA, Perl, Eiffel**: '_' is: 456_789
  ADA-ban: exponenciális alak is - pl. 1E6 -, ahol a kitevő nem negatív
  C, C++, Java, C#: *int* és *long* különbsége: 'L' a literál végén, pl. 2 vagy 2L. ‘U’ az unsigned

- **<u>Valósak</u>**:

  [-]digit{digit}.digit{digit}[exponent]

  -123.456E+3

  Majdnem minden nyelv ad ehhez valami specialitást:

  Pascal: [-]digit{digit}[exponent] 12E+3

  ADA: ‘_’ is lehet: -1_234.0

  Eiffel: [-]{digit}.{digit}[exponent] -1.

  Java: “float_suffix” is lehet a végén

Milyen alapok megengedettek?

- Általában: decimális számok

  néha más is -- pl. 2, 8 16
- **Pascal, Modula, Oberon**: csak decimális számok
- **C, C++, Java**: Oktális, decimális és hexadecimális
  oktális: 0734
  hexadecimális: 0xFF, 0xC0B0L.
- **ADA**: minden alap 2 és 16 között:
  base#mantissa#[exponent]
  pl. 16#FFF.F#E1
- **Mathematica**: minden alap 2 és 36 között!
  (minden számjegy és angol betű OK.) :
  basis ^^mantissa pl..:30^^Mathematica =
  $13\ 207\ 019\ 439\ 499\ 570_{10}$

Karakterek és karaktersorozatok

- Karakterek: 'A' vagy „escape szekvenciák”: '\n’, '\'’, '\ooo' stb.
- Sztringek "A”
- tárolásuk különböző lehet.

Megjegyzések

- Szerkezete befolyásolja a program megbízhatóságát
- Szokásos lehetőségek:
  - egy speciális oszloptól (jeltől) a sor végéig
    - pl. Fortran ‘C’ az első oszlopba
  - speciális jel(ek) elején - végén
  - speciális jel(ek) elején - vége a sor vége

Megjegyzések

- **Pascal:** (* és *) vagy { és }
- **CLU:** %-tól sor végéig
- **C**: /* és */
- **C++:** // -tól sor végéig
  vagy /* és */
- **Java:** mint **C++-**ban, vagy:
  /** documentation */
- **C#:** mint **C++-**ban, vagy:
  /// egysoros dokumentációs megjegyzés
- **ADA, Eiffel:** -- -tól sor végéig