
12 ZÁVĚR

V této poslední kapitole se budeme zabývat příkazem, jehož užívání v programech není nutné a může vést k nepřehledným programům. Proto často bývá začátečníkům zamlčován. Rozumným užíváním však může tento příkaz program zefektivnit bez ztráty přehlednosti.

12.1 Příkaz skoku

V kap. 4 jsme hovořili o řídicích strukturách. V Pascalu jsou všechny řídicí struktury navrženy tak, že mají tvar jednoho příkazu. Program je zapsán přehledně, ale může se stát, že není dostatečně efektivní. V některých případech si totiž může užívání popsaných struktur vynutit zavedení pomocných logických proměnných, popř. vícenásobných testů též podmínky.

Kromě uvedených řídicích struktur existuje v Pascalu ještě jedna možnost řízení běhu programu. Je nazývána *skok*.

Příkazy programu zapsané za sebou v textu programu byly dosud v našich programech prováděny pořadí, v němž byly v textu napsány. Skok umožňuje toto pořadí prováděných příkazů změnit.

V každém bloku je možné libovolný příkaz označit celým číslem nazvaným *návěští*:

návěští : příkaz

Lze tedy napsat

- 12 : I := I + 1;

35 : while A < 1 do A := A - 2.3;

Na takto označený příkaz lze přesunout řízení (pokračování výpočtu) při běhu programu *příkazem skoku* (go to = jdi na).

goto návěští

např.

goto 12

Příkaz skoku může program urychlit. V mnoha případech ovšem také může program znepřehlednit, což je daleko větší nedostatek (zejména pro začá-



tečníky). Proto se užívání příkazu skoku, pokud je to možné, vyhýbáme a klademe na ně jistá omezení.

Všechna návěští deklarovaná u příkazů v daném bloku musí být definována v části definice návěští v témže bloku a každé definované návěští musí být použito:

label

návěští₁, ..., návěští_n;

Část definice návěští se nachází na začátku bloku před všemi deklaracemi.

label

návěští₁, ..., návěští_n;

const

:

type

:

var

Dále příkaz skoku nesmí směrovat zvnějšku do řídicí struktury. Není povolen např.

```
while A < 0 do
begin
:
28 : A := A - 2.3;
:
end;
:
goto 28;
```

Naopak je povolen skok z řídicí struktury, a dokonce i skok z procedury nebo funkce do bloku nadřazené funkce nebo procedury, např.

```
procedure A;
label 1;
procedure B;
begin { tělo procedury B }
:
goto 1;
:
end; { B }
begin
:
1: I := I + 1;
:
end; { A }
```

V některých algoritmech je využití příkazu skoku opodstatněné, i když není nutné. Jsou to zejména skoky ukončující okamžitě provedení řídicí struktury. Například skok na konec těla bloku

```
begin { tělo A }
:
goto 999;
:
999: end; { A }
```

popř. na konec některého z nadřazených bloků (nebo i na konec programu). Časté je také užití příkazu skoku pro předčasné ukončení cyklu.

Chceme-li např. prohledat pole celočíselných složek

```

var
  A : array [1..100] of integer;
nalézt v něm první nenulovou hodnotu a její index uložit do proměnné PRV-
NI, je možné napsat

label 100;
var
  I      : integer;
  A      : array [1..100] of integer;
  PRVNI : integer;
begin
  :
  PRVNI := 0; { hodnota 0 znamená nenašení }
  for I := 1 to 100 do
    if A[I] < > 0 then
      begin { nenulová složka nalezena }
        PRVNI := I;
        goto 100;
      end; { if }
  100: { pokračování programu }
  :
end;

```

12.2 Cvičení

1. Program 11.2 (editor) modifikujte tak, že po zadání příkazu E (konec programu) se nenastaví proměnná CYKLUJ na hodnotu *false*, nýbrž se provede skok na konec programu. Podobně lze ukončovat i další konverzní programy.

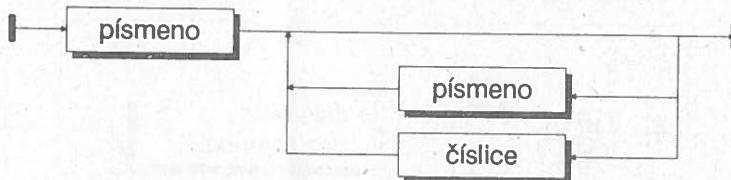
A SYNTAKTICKÁ DEFINICE PASCALU

Všechny zápisy v programu mají přesně definovaný tvar. Říkáme, že je definována jejich skladba, neboli *syntaxe*. Slovní popis syntaxe prvků programu není často dostatečně přesný, proto zavedeme jednoduché grafické prostředky pro vyjádření skladby jednotlivých konstrukcí jazyka. Říkáme jim *syntaktické diagramy*.

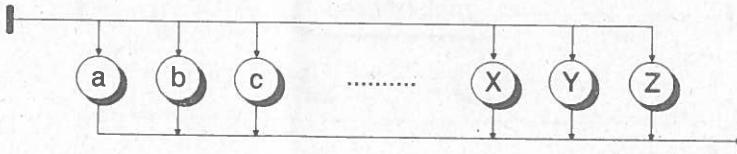
Každý syntaktický diagram popisuje jednu konstrukci Pascalu. Její jméno je vždy uvedeno vlevo nad příslušným diagramem. Syntaktický diagram určuje všechny povolené tvary konstrukce. Každý povolený tvar získáme tak, že projdeme uzly od začátku do konce diagramu. Začátek diagramu je vždy vlevo nahore.

Například syntaktický diagram definující identifikátor má tvar:

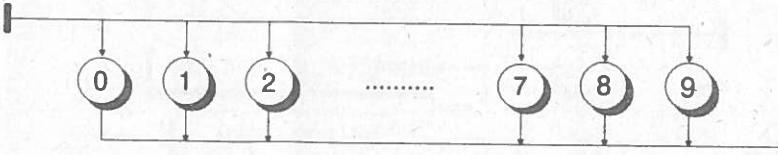
- **identifikátor**



- **písmeno**

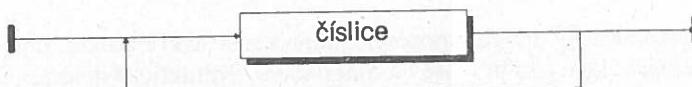


- **číslice**

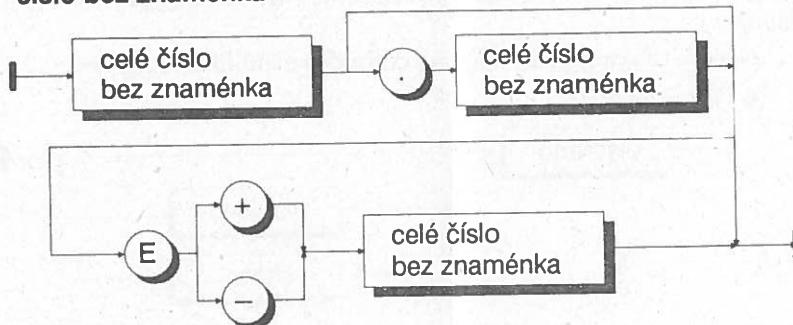


Posloupnost symbolů tvořících správnou konstrukci Pascalu vytváříme zleva doprava :

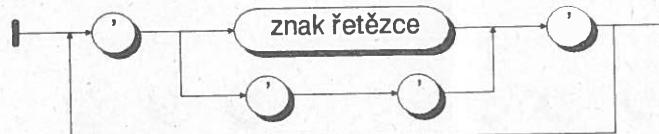
- Procházíme-li uzlem diagramu, který má tvar kruhu nebo oválu, pak symbol v uzlu přidáme do posloupnosti.
- Procházíme-li uzlem ve tvaru obdélníka (znamená odkaz na jiný vnořený syntaktický diagram), pak k vytvářené posloupnosti symbolů přidáme libovolnou posloupnost symbolů definovanou vnořeným diagramem.
- **celé číslo bez znaménka, návěští**



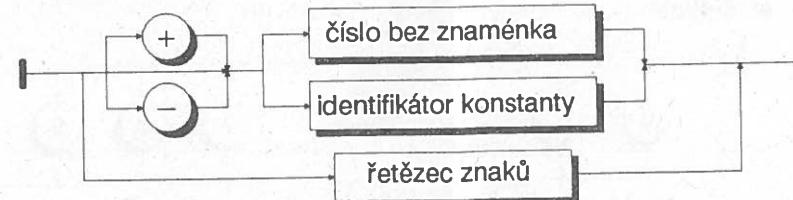
● **číslo bez znaménka**



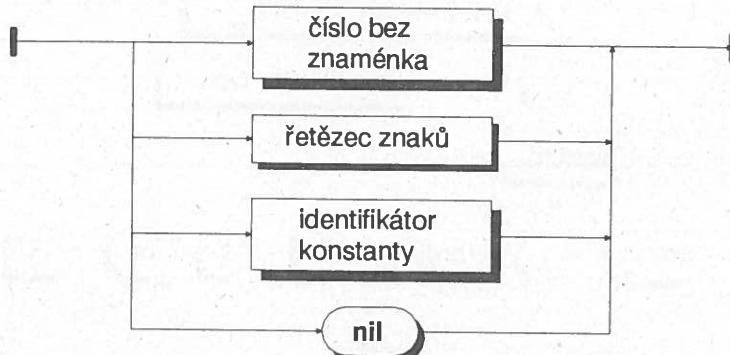
● **řetězec znaků**



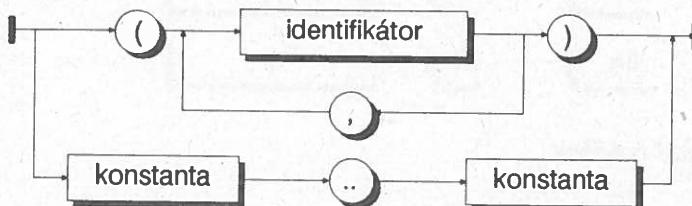
● **konstanta**



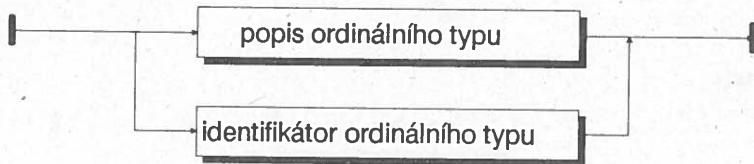
- konstanta bez znaménka



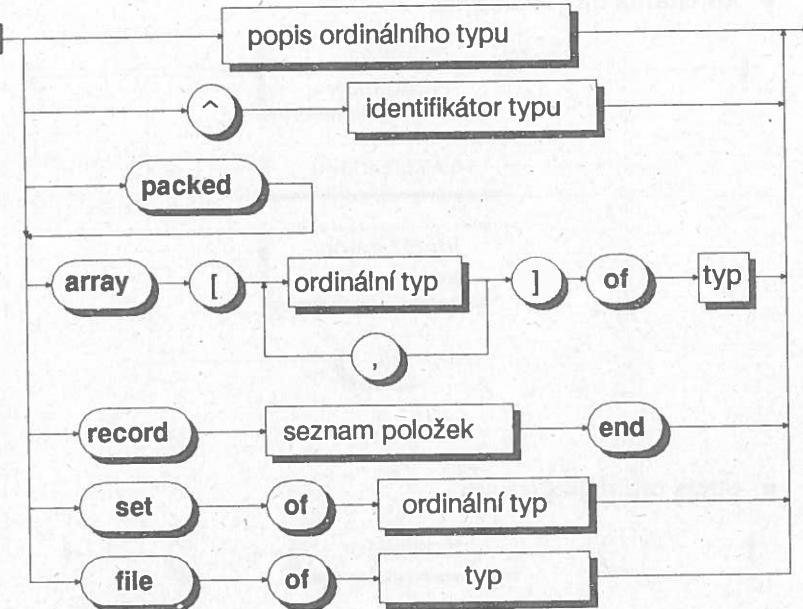
- popis ordinálního typu



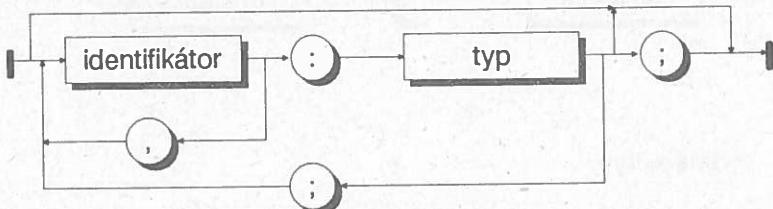
- ordinální typ



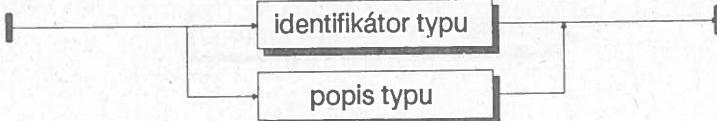
- **popis typu**



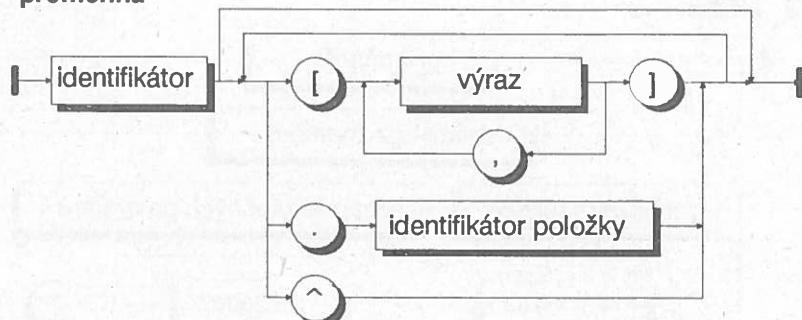
- **seznam položek**



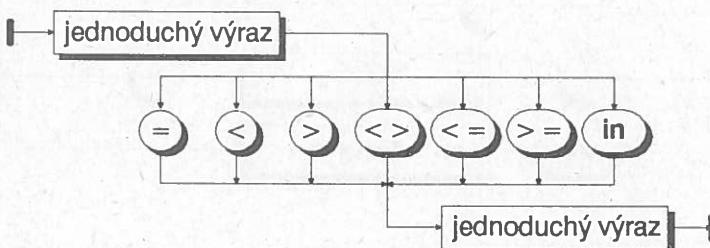
- **typ**



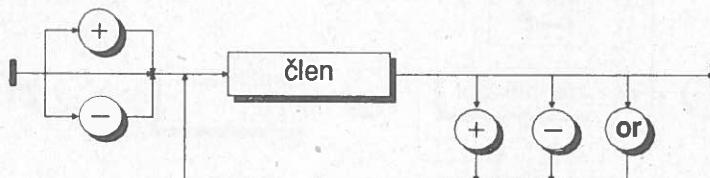
- proměnná



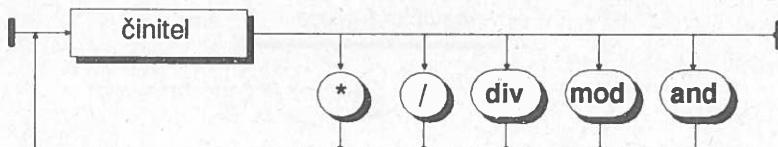
- výraz



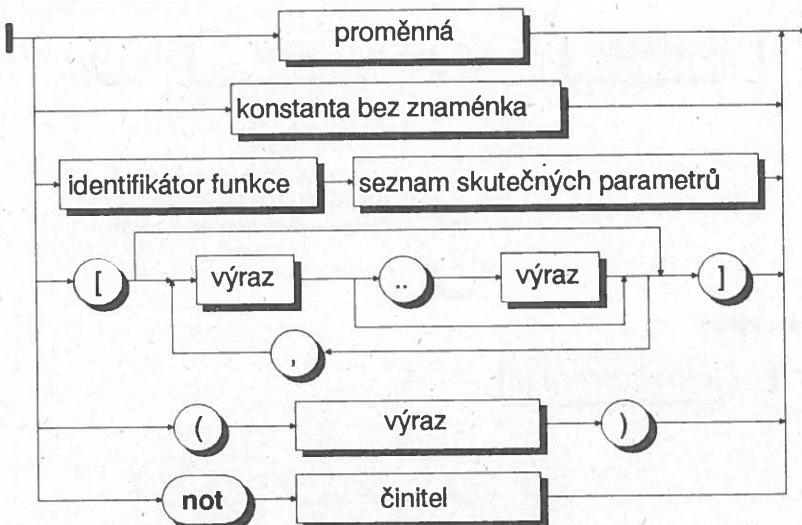
- jednoduchý výraz



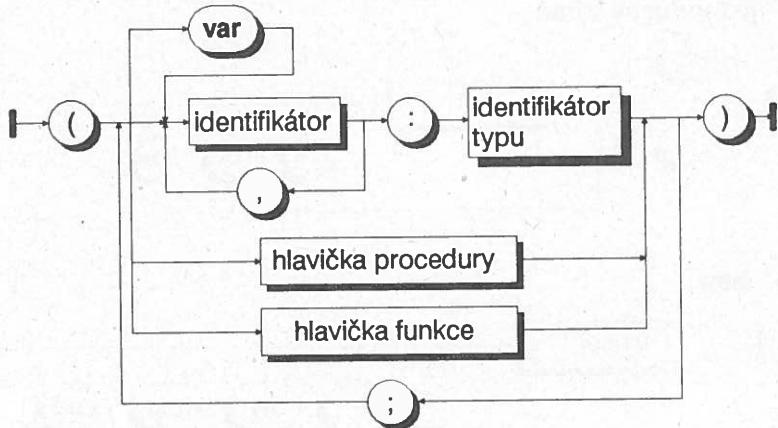
- člen



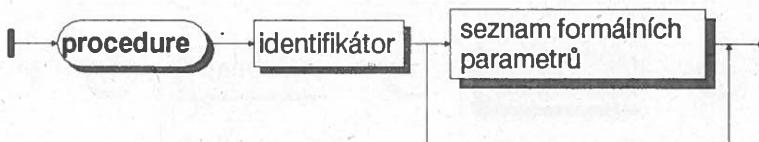
- činitel



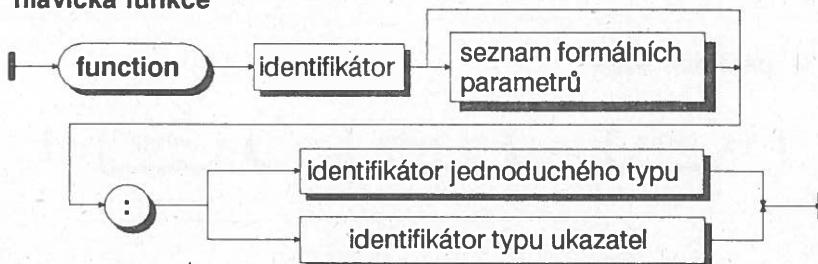
- seznam formálních parametrů



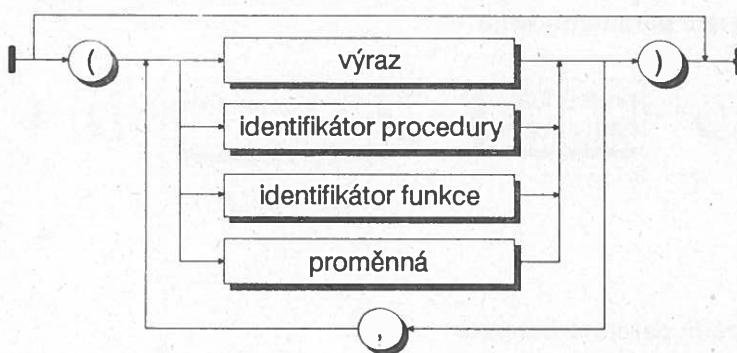
- hlavička procedury



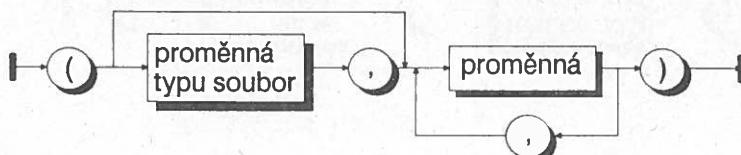
- hlavička funkce



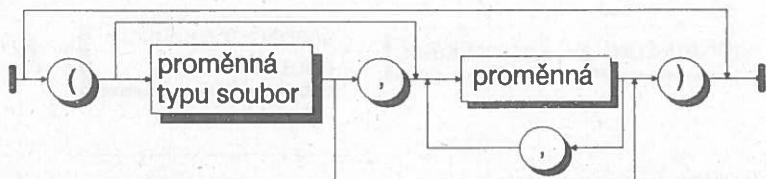
- seznam skutečných parametrů



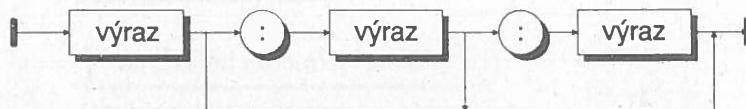
- seznam parametrů read



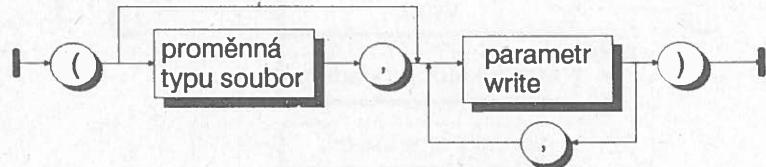
- seznam parametrů readin



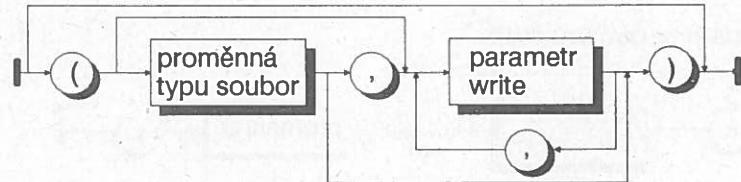
- parametr write



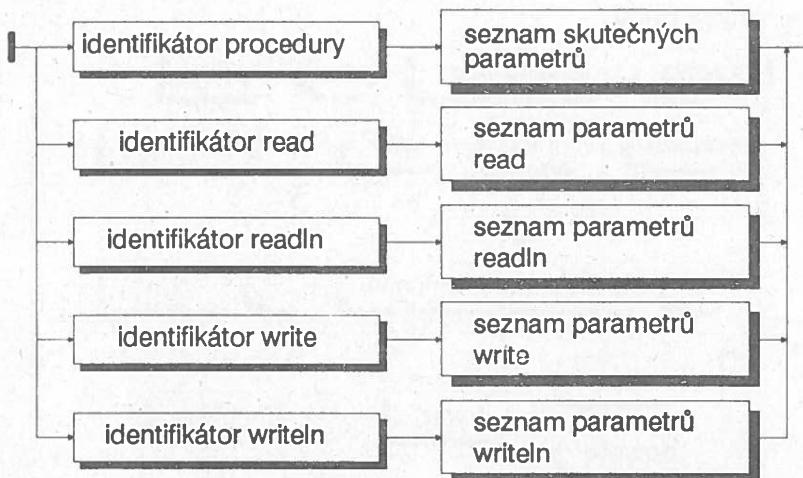
- seznam parametrů write



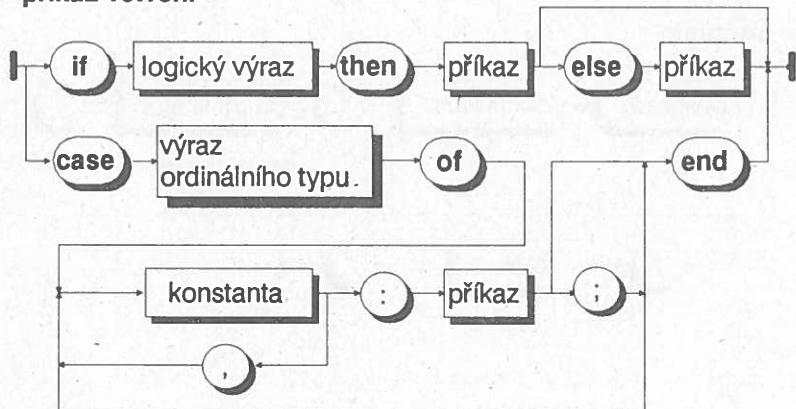
- seznam parametrů writeln



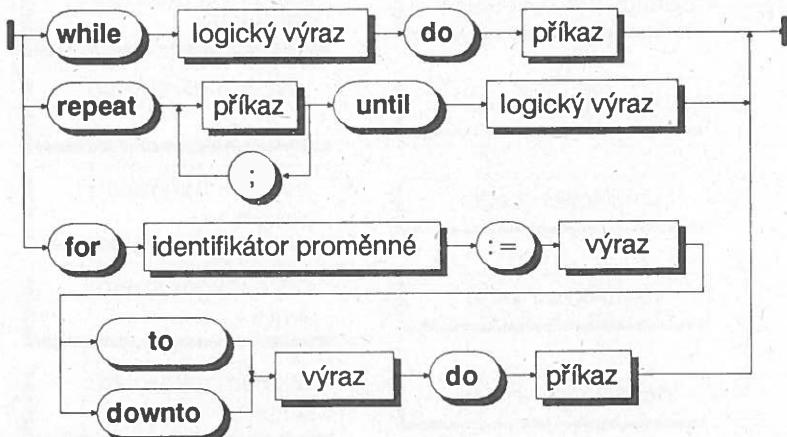
- příkaz procedury



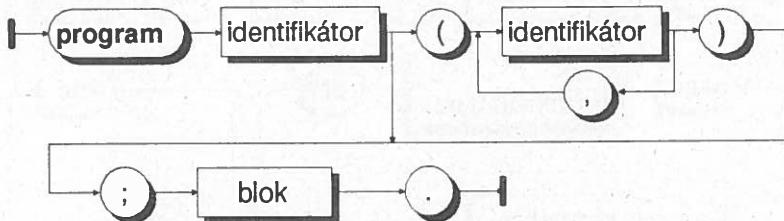
- příkaz větvení



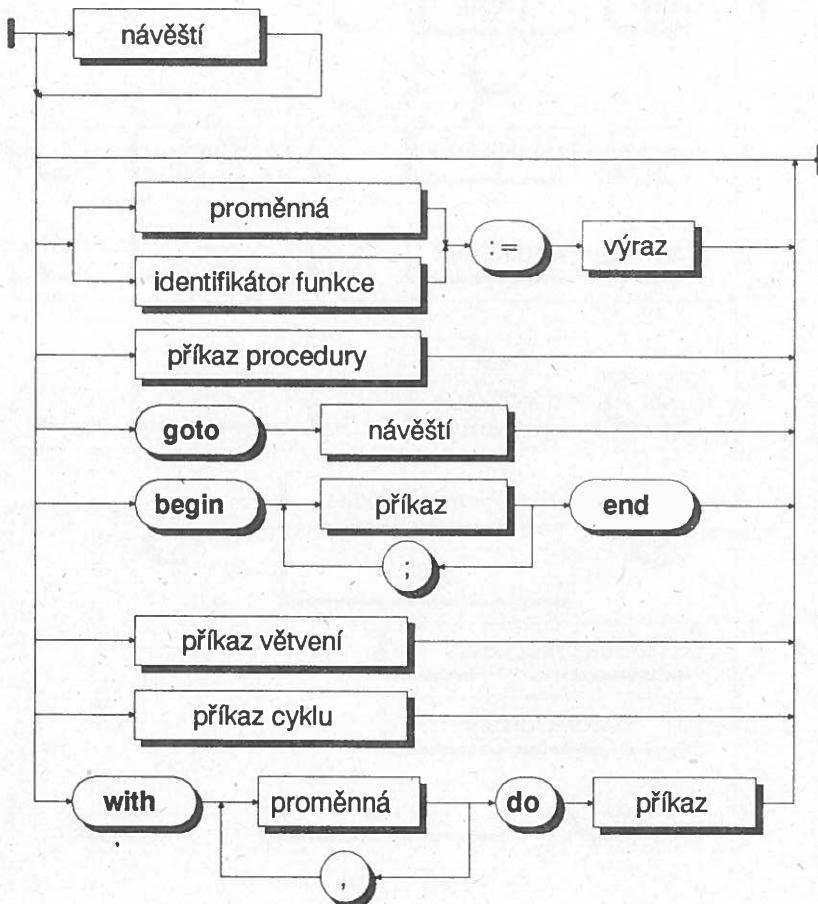
- příkaz cyklu



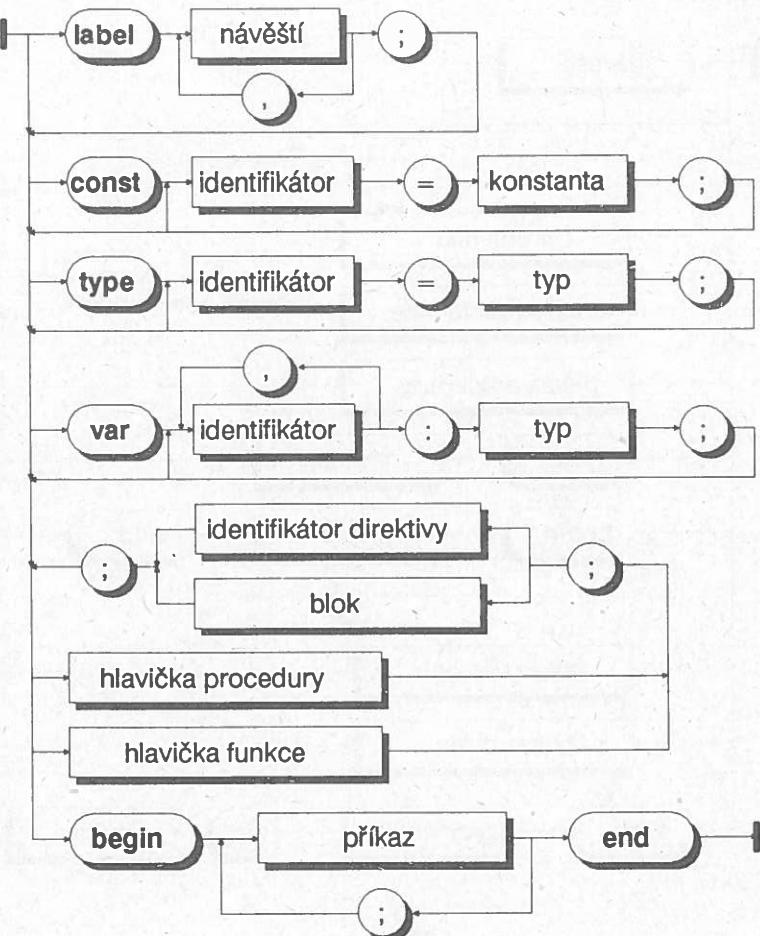
- program



● příkaz



- **blok**



B PŘÍRUČKA

• literál

7653	0	958	-128	0256
85.6	-59.638	5E16	+18 .5E-1	+65.9E +3
'a'	'X'	'0'	'%'	'''
'TOTO JE RETEZEC' '' 'vyjadreni apostrofu ''.				

• identifikátor

I	j	EDITOR	chr	text	write
ToToJeIdEnTiFiKaToR	POM1	POM2	integer		Telefon
read	a24X2545				

• komentář

{ komentář – vysvětlující text }
(* komentář – vysvětlující text *)

• vyhrazená slova

and	array	begin	case	const	div
do	downto	else	end	file	for
function	goto	if	in	label	mod
nil	not	of	or	packed	
procedure	program	record	repeat	set	then
to	type	until	var	while	with

• výčtový typ

type
DNY = (PONELI, UTERY, STREDA, CTVRTEK, PATEK,
SOBOTA, NEDELE);

var
DEN: DNY;

DEN := PONDELI;

ord (PONDELI) = 0

ord (NEDELE) = 6

pred (SOBOTA) = PATEK

ord (STREDA) = 2

succ (STREDA) = CTVRTEK

DEN = PONDELI

PONDELI < NEDELE

CTVRTEK < = PATEK

DEN < > UTERY

SOBOTA > PONDELI

STREDA > = UTERY

• boolean

var

LOGICKA1, LOGICKA2: boolean;

LOGICKA1 := true;

write (LOGICKA1);

writeln (LOGICKA2:7);

ord (true) = 1

succ (false) = true

LOGICKA1 = true

false < true

false < = true

ord (false) = 0

pred (true) = false

LOGICKA1 < > false

true > false

true > = false

LOGICKA1 and LOGICKA2 LOGICKA1 or LOGICKA2

not LOGICKA1

• integer

const

maxint = maximální možné celé číslo;

var

I, J, K: integer;

I := 0;

readln (K);

readln (J,K);

read (J);

write (I:5);

ord (0) = 0

ord (78) = 78

pred (J) = J-1

I < > 8

ord (-345) = -345

succ (I) = I + 1

I = 0

-345 < 78

$78 > -345$ $I \leq 123$
 $I \geq 0$

$I + J$ $I - J$
 $I * J$ $I \text{ div } J$
 $I \bmod J$

$\text{abs}(I)$

• real

var

A, B, C: *real*;

$A := 10.05;$	<i>read</i> (B);
<i>readln</i> (A);	<i>write</i> (A:10:5);
<i>writeln</i> (B:8,C);	<i>write</i> (C);
$A = 10.05$	$A < > - 45.7E - 3$
$- 45.7E - 3 < 10.05$	$10.05 > - 45.7E - 3$
$A \leq 34.8E2$	$A \geq 78.98E - 45$

$A + B$	$A - B$
$A * B$	A / B
<i>sqrt</i> (A)	<i>sqr</i> (A)
<i>sin</i> (A)	<i>cos</i> (A)
<i>arctan</i> (A)	<i>ln</i> (A)
<i>exp</i> (A)	<i>abs</i> (A)
<i>round</i> (A)	<i>trunc</i> (A)

• char

var

C1, C2, C3: *char*;

$C1 := 'A';$	<i>read</i> (C2);
<i>readln</i> (C3);	<i>write</i> (C1:5);
<i>writeln</i> (C2:2,C3);	
$ord('A') = 65$	$ord('0') = 48$
$ord('d') = 100$	$succ('B') = 'C'$

● soubor

type

SOU1 = **file of integer**;
 SOU2 = **file of ZAZ1**;

var

S1 : SOU1;	I: <i>integer</i> ;
S2 : SOU2;	A: <i>real</i> ;
S3 : <i>text</i> ;	K: <i>char</i> ;
S1 ^ := 576;	get (S1);
<i>put</i> (S1);	I := S1 ^ ;
<i>reset</i> (S1);	<i>rewrite</i> (S1);
S2 ^ .S2 := NEDELE;	<i>eof</i> (S2)
<i>read</i> (S2,Z1);	<i>write</i> (S2,Z1A);
<i>read</i> (S3,I);	<i>readln</i> (S3,A);
<i>write</i> (S3,I:5,A:9);	<i>writeln</i> (S3,K);
<i>eoln</i> (S3)	<i>page</i> (S3);

● množina

type

MNO1 = **set of DNY**;
 MNO2 = **set of char**;
 MNO3 = **set of 0..127**;

var

M1A,	
M1 : MNO1;	
M2 : MNO2;	
M3 : MNO3;	
M1 := M1A;	M1A := [];
M1 := [PONDELI, UTERY];	
M1 + M1A	M1 * [PATEK..NEDELE]
M1A - [UTERY..PATEK]	
M1 = M1A	M1 < > [PONDELI]
M1A < = M1	M1A > = M1

CTVRTEK in M1	'a' in M2
M2 := [];	M2 := ['A'..'Z','0'..'9']
M3 := [I * 3,100] I in [1..10, 90..112]	M3 := [];

• ukazatel

```

type
    UKA1 = ^UZEL;
    UZEL =
        record
            CO : integer;
            DALSI: UKA1;
        end; { UZEL }

var
    U1 : UKA1;
    U2 : UKA1;

    new (U1);
    new (U1^.DALSI);
    U1 = U2
    U1 := U2;
    U1^.CO := 345;
    U2 := U1^.DALSI^.DALSI;
    U1 <> U2
    dispose (U2);

```

• priorita operátorů

1. = < > < > < = > = in
2. + - or
3. * / div mod and
4. not

• příkazy

<i>proměnná</i> := <i>výraz</i>	přiřazení
<i>identifikátor</i> (<i>výraz</i> , ... , <i>výraz</i>)	příkaz procedury
<i>goto návěští</i>	skok na návěští

		prázdný příkaz
begin	<i>příkaz;</i>	složený příkaz
	<i>:</i>	
	<i>příkaz</i>	
end		
if <i>výraz</i>		neúplný
then	<i>příkaz</i>	podmíněný příkaz
if <i>výraz</i>		úplný
then	<i>příkaz</i>	podmíněný příkaz
else	<i>příkaz</i>	
case <i>výraz</i> of		příkaz case
<i>konstanta</i> , ... , <i>konstanta</i> : <i>příkaz</i> ;		
<i>:</i>		
<i>konstanta</i> , ... , <i>konstanta</i> : <i>příkaz</i> ;		
end		
for <i>proměnná</i> := <i>výraz</i> to <i>výraz</i> do		cyklus for
<i>příkaz</i>		
for <i>proměnná</i> := <i>výraz</i> downto <i>výraz</i> do		
<i>příkaz</i>		
while <i>výraz</i> do		cyklus while
<i>příkaz</i>		
repeat		cyklus repeat
<i>příkaz</i> ;		
<i>:</i>		
<i>příkaz</i>		
until <i>výraz</i>		
with <i>proměnná</i> , ... , <i>proměnná</i> do		příkaz with
<i>příkaz</i>		

Tabulka znakového kódu ASCII

znak	klávesa	<i>ord(znak)</i>	osmičkově	šestnáctkově
NUL	Ctrl @	0	000	00
SOH	Ctrl A	1	001	01
STX	Ctrl B	2	002	02
ETX	Ctrl C	3	003	03
EOT	Ctrl D	4	004	04
ENQ	Ctrl E	5	005	05
ACK	Ctrl F	6	006	06
BEL	Ctrl G	7	007	07
BS	Ctrl H	8	010	08
TAB	Ctrl I	9	011	09
LF	Ctrl J	10	012	0A
VT	Ctrl K	11	013	0B
FF	Ctrl L	12	014	0C
CR	Ctrl M	13	015	0D
SO	Ctrl N	14	016	0E
SI	Ctrl O	15	017	0F
DLE	Ctrl P	16	020	10
DC1	Ctrl Q	17	021	11
DC2	Ctrl R	18	022	12
DC3	Ctrl S	19	023	13
DC4	Ctrl T	20	024	14
NAK	Ctrl U	21	025	15
SYN	Ctrl V	22	026	16
ETB	Ctrl W	23	027	17
CAN	Ctrl X	24	030	18
EM	Ctrl Y	25	031	19
SUB	Ctrl Z	26	032	1A
ESC	Ctrl [27	033	1B
FS	Ctrl \	28	034	1C
GS	Ctrl]	29	035	1D
RS	Ctrl ^	30	036	1E
US	Ctrl _	31	037	1F
''	mezerník	32	040	20
'!'	!	33	041	21
'''	"	34	042	22
'#'	#	35	043	23
'\$'	\$	36	044	24
'%'	%	37	045	25
'&'	&	38	046	26
'''	,	39	047	27

znak	klávesa	<i>ord(znak)</i>	osmičkově	šestnáctkově
'('	(40	050	28
')')	41	051	29
'*'	*	42	052	2A
'+'	+	43	053	2B
','	,	44	054	2C
'_'	-	45	055	2D
'.'	.	46	056	2E
'/'	/	47	057	2F
'0'	0	48	060	30
'1'	1	49	061	31
'2'	2	50	062	32
'3'	3	51	063	33
'4'	4	52	064	34
'5'	5	53	065	35
'6'	6	54	066	36
'7'	7	55	067	37
'8'	8	56	070	38
'9'	9	57	071	39
:	:	58	072	3A
,	;	59	073	3B
'<'	<	60	074	3C
'='	=	61	075	3D
'>'	>	62	076	3E
'?'	?	63	077	3F
'@'	@	64	100	40
'A'	A	65	101	41
'B'	B	66	102	42
'C'	C	67	103	43
'D'	D	68	104	44
'E'	E	69	105	45
'F'	F	70	106	46
'G'	G	71	107	47
'H'	H	72	110	48
'I'	I	73	111	49
'J'	J	74	112	4A
'K'	K	75	113	4B
'L'	L	76	114	4C
'M'	M	77	115	4D
'N'	N	78	116	4E
'O'	O	79	117	4F

znak	klávesa	<i>ord(znak)</i>	osmičkově	šestnáctkově
'P'	P	80	120	50
'Q'	Q	81	121	51
'R'	R	82	122	52
'S'	S	83	123	53
'T'	T	84	124	54
'U'	U	85	125	55
'V'	V	86	126	56
'W'	W	87	127	57
'X'	X	88	130	58
'Y'	Y	89	131	59
'Z'	Z	90	132	5A
'[[91	133	5B
'\'	\	92	134	5C
'`'	l	93	135	5D
'^'	^	94	136	5E
'-'	-	95	137	5F
''	'	96	140	60
'a'	a	97	141	61
'b'	b	98	142	62
'c'	c	99	143	63
'd'	d	100	144	64
'e'	e	101	145	65
'f'	f	102	146	66
'g'	g	103	147	67
'h'	h	104	150	68
'i'	i	105	151	69
'j'	j	106	152	6A
'k'	k	107	153	6B
'l'	l	108	154	6C
'm'	m	109	155	6D
'n'	n	110	156	6E
'o'	o	111	157	6F
'p'	p	112	160	70
'q'	q	113	161	71
'r'	r	114	162	72
's'	s	115	163	73
't'	t	116	164	74
'u'	u	117	165	75
'v'	v	118	166	76
'w'	w	119	167	77

znak	klávesa	<i>ord(znak)</i>	osmičkově	šestnáctkově
'x'	x	120	170	78
'y'	y	121	171	79
'z'	z	122	172	7A
'{'	{	123	173	7B
' '		124	174	7C
'}'	}	125	175	7D
'~'	~ (tilda)	126	176	7E
del	del	127	177	7F

Znaky s ordinálními čísly 0 až 31 jsou řídicími znaky. Na klávesnici se zapisují současným stlačením klávesy **Ctrl** a některé další klávesy. Na obrazovce nebo jiném výstupním zařízení jsou obvykle nezobrazitelné. Řídicí znaky jsou pojmenovány zkratkami vycházejícími z jejich anglických názvů.

Nejpoužívanější jsou následující řídicí znaky:

- BEL – zvukový signál,
- BS – posun o znak zpět,
- TAB – tabelátor,
- LF – posun na další řádek,
- VT – vertikální tabelátor,
- FF – posun na začátek další stránky,
- CR – návrat vozíku na začátek řádku.

LITERATURA

- [1] ATKINSON,L.V.: A Student's guide to Programming in Pascal. London, John Wiley a Sons 1982
- [2] ATKINSON,L.V.: Pascal Programming. London, John Wiley a Sons 1980
- [3] BOWEN,K.A.: Speaking Pascal. New Jersey, Hayden Book Comp.1981
- [4] BUTOMO,I.D. – SAMOČADIN,A.V. – USANOVA,D.V.: Programirovanie na algoritmičeskom jazyke Pascal dlja mikro EVM. Leningrad, Izdateľstvo leningradskovo universiteta 1985
- [5] COLLINS,W.J.: Intermediate Pascal Programming. New York, McGraw – Hill Book Company 1986
- [6] COLLINS,W.J.: An Introduction to Programming in Pascal. New York, Macmillan Publishing Company 1984
- [7] EISENBACH,S. – SADLER,C.: Pascal for Programmers. Berlin, Springer Verlag 1981
- [8] FAUSTUS,L. – POLÍVKA,F.: Botanický klíč. Praha, SPN 1975
- [9] GROGONO,P.: Programming in Pascal. Reading, Adison – Wesley 1980
- [10] HAMZA,J. a kol.: Rok s Evou. Bratislava, Obzor 1984
- [11] HORÁK,Z. – KRUPKA,F.: Fyzika. Praha, SNTL 1976
- [12] ISO/TC97/SCS Programming Languages Specification for Computer; Programming Language Pascal. Third Draft Proposal ISO/DP7185, 1981
- [13] JENSEN,K. – WIRTH,N.: Pascal – rukovodstvo dlja polzovatela i opisanie jazyka. Moskva, Finansy i statistika 1982 (překlad z angličtiny)
- [14] JINOCH,J. – MÜLLER,K. – VOGEL,J.: Programování v jazyku Pascal. Praha, SNTL 1987
- [15] KERNIGHAN,B.W.: Software Tools in Pascal. New York, Addison – Wesley Publishing Company 1981
- [16] KIEBURTZ,L.: Structured Programming and Problem Solving with Pascal. London, Prentice Hall 1977
- [17] KOHLMAN,Č.: Matematika sdělovačí techniky. Praha, Technicko – vědecké vydavatelství 1951
- [18] LEWIS,T.: Pascal Programming for the Apple. London, Prentice Hall 1981
- [19] MOLNÁR,L.: Programovanie v jazyku Pascal. Bratislava, Alfa 1987
- [20] OCHRANOVÁ,R.: Úvod do programování. [Skriptum.] Brno, UJEP 1982
- [21] PAULIN,G. – SCHIEMANGK,H.: Programmieren mit Pascal. Berlin, Akademie Verlag 1981
- [22] PERMINOV,O.N.: Jazyk programirovanijsa Pascal. Moskva, Radio i svjaz 1983

- [23] RÁBOVÁ,Z. a kol.: Počítače a programování. [Skriptum.] Brno, VUT 1980
- [24] RŮŽIČKA,J. – HRUŠKA,E.: Fyzika. [Skriptum.] Brno, VŠZ 1983
- [25] SCHNEIDER,G.M. – WEINGART,S.W. – PERELMANN,D.M.: An Introduction to Programming and Problem Solving in Pascal. New York, John Wiley & Sons 1978
- [26] Turbo Pascal 4.0, manuál firmy Borland International 1987
- [27] WIRTH,N.: Algoritmy + struktury dannyh = programy. Moskva, Mir 1985 (překlad z angličtiny)
- [28] WIRTH,N.: Systematické programovanie. Bratislava, Alfa 1981
- [29] WIRTH,N.: The Programming Language Pascal. Acta Informatica, 1, 1971, str. 35 – 63

REJSTŘÍK

- Abs 32
- algoritmus 9
- and 43
- apostrof 28, 78
- arctan 32
- array 154
- ASCII 10, 79, 83
 - tabulka kódu 357
- assign 239

- Begin 47
- bit 9
- blok 107
- boolean 42
- BS 77
- byte 9, 78

- Case 57
- close 239
- const 36
- cos 32
- CR 77, 261, 331
- Ctrl 78
- cyklus 65
 - for 65
 - repeat 71
 - while 68

- Část deklarační 24
 - příkazová 25
- čistota typová 279
- číslice 77
- číslo celé 15, 20
 - ordinální 10, 77, 78
 - reálné 30

- Definice návěstí 334
 - syntaktická 337
- deklarace funkce 122
 - konstant 36
 - procedury 94, 97, 107

- proměnné 22
- typů 86
- dělení 32
 - celočíselné 16
- diagram syntaktický 337
- direktiva 148
- dispose 282
- div 16
- do 66, 68
- downto 67

- Editor 11, 318
- editování 11
- efekt vedlejší 140
- ekvivalence 44
- else 46, 51, 61
- end 47
- eof 225
- eoln 248
- exp 32
- exponent 31
- external 149

- False 42
- file 222
- for 66
- formát 14
 - volný 54
- forward 148
- function 122
- funkce 122
 - rekurzivní 133
 - standardní 32

- Get 225
- goto 333

- Hlavíčka procedury 94
- hodnota logická 42

- Char 77
- chr 79
- chyba 11
 - při překladu 11
 - — spuštění 11
- Identifikátor 20, 349
 - standardní 21
- if 46, 50
- implikace 44
- in 273
- index řádkový 170
 - sloupcový 170
- inicializace cyklu 69
- inkluze množinová 273
- input 27, 250
- instrukce 9
- integer 22
- interval 88
- Jazyk programovací 10
- Koefficient binomický 133
- komentář 13, 21, 56, 349
- kompatibilita vzhledem k přiřazení 279
- konstanta 35
 - pojmenovaná 35
 - strukturovaná 165
- konstruktor množiny 272
- konverze 79, 138, 245
- Label 334
- ladění 11
- LF 77, 261
- literál 20, 349
 - textový 28
 - znakový 78
- ln 32
- Matice 170
- maxint 36, 83
- metoda lichoběžníková 142
 - návrhu shora dolů 119, 279, 308
 - Newtonova iterační 75
- množina 271
 - prázdná 272
- mod 17
- Násobení 17
- návěstí 333
- new 282
- nil 283
- nonekvivalence 44
- norma Pascalu 11
- not 43
- Objekt globální 108
 - lokální 108
 - nelokální 108
 - soukromý 106
 - standardní 21
- odčítání 17
- operace 16
 - relační 83
- operand 16
- operátor 16
 - binární 16
 - relační 41, 161
 - unární 19
- or 43
- ord 79, 84
- otherwise 61
- output 25, 250
- Pack 163
- packed 160
- page 248
- parametr 112
 - formální 112, 113
 - funkcionální 145
 - procedurální 145
 - programu 226
 - skutečný 112, 113
 - volaný hodnotou 115, 279
 - — odkazem 115, 160, 278
 - vstupní 113
 - výstupní 113, 114
 - write 247
- Pascal 10
- písmeno 77
- počítáč 9
- pohyb aktivní složky souboru 224
- pole 152
 - konstantní 165
 - vícerozměrné 170
 - zhuštěné 160
- pred 82, 84

- priorita 18, 19, 31, 42, 43, 355
- procedura 94
 - rekurzívní 133
- procedure 94
- procesor 9
- program 9, 24
 - BLOKOVASTRUKTURA 109
 - DATUM 102, 116
 - DELKA 34, 39
 - EDITOR 324
 - ENAXTOU 74
 - EXPERT 49
 - FAKTORIAL 67
 - FORMAT 261
 - INTEGRAL 143, 146
 - JIDLO 165
 - KOEFIC 137
 - KONVERZE 81
 - KOPTEXT 248, 251
 - MERENI 157
 - NAVIGACE 130
 - OBSAHY 63
 - PRUMER 69
 - PUJCKA 24, 27, 29
 - SPORT 210, 265, 298
 - SPOTREBA 13
 - STATISTIKA 274
 - TELEFON 239
 - VEDLEJSIEFEKT 140
 - ZIVOT 178
- programování 9
 - proměnná 13, 22, 114
 - dynamická 280
 - přístupová soubor 223
 - řídicí cyklu 66, 279
 - statická 280
 - průnik množin 273
- překladač 10
- přepínač 57
- příkaz 10
 - case 57, 356
 - for 65, 356
 - přířazovací 22, 279, 355
 - podmíněný neúplný 50, 356
 - úplný 46, 356
 - prázdný 52, 356
 - procedury 94, 355
 - repeat 71, 356
 - skoku 333, 355
 - složený 47, 356
- while 68, 356
- with 207, 356
- put 224
- Read 246
 - pro celá čísla 26
 - — obecný soubor 225
 - — reálná čísla 33
 - — znaky 79
 - readln 14, 247
 - pro celá čísla 26
 - — reálná čísla 33
 - — znaky 79
- real 30
- record 188
- rekurze 124
- repeat 71
- reset 224, 253
- rewrite 224
- round 37
- rozdíl množin 273
- Řetězec znaků 28, 160
- Sčítání 16
- set 271
- seznam dvousměrně vázaný 308
 - jednosměrně vázaný 286
- sin 32
- schéma Hornerovo 80
- sjednocení množin 273
- skok 333
- slabika 9, 78
- slово klíčové 21
 - rezervované 21
 - vyhrazené 21, 349
- složka souboru aktivní 222
 - pole 155, 172
 - záznamu 189, 208
- soubor 222
 - externí 226
 - interní 226
 - pracovní 226
 - textový 245
 - vnější 226
- soustava desítková 15
- sqr 32
- sqrt 32
- stav souboru 224

- string 321
 - struktura bloková 107
 - datová 152
 - – dynamická 283
 - řídicí 45
 - výrazu 18
 - středník 52, 56
 - succ 82, 84
 - symbol speciální 21
 - znakový 21
 - Then 46, 50
 - to 66
 - true 42
 - trunc 37
 - třídění bubenové 182
 - Turbo Pascal 8, 11, 239
 - tvář desetinný 31
 - semilogaritmický 31
 - zápisu infixový 16
 - typ 22
 - abstraktní 279, 306
 - boolean 350
 - datový 22, 277
 - doménový 281, 285
 - char 351
 - indexu 153
 - integer 350
 - interval 352
 - jednoduchý 83
 - konkrétní 307
 - množina 354
 - ordinální 83
 - pole 352
 - real 351
 - řetězec znaků 160
 - soubor 354
 - standardní 83
 - strukturovaný 152
 - ukazatel 355
 - uživatelský 86
 - výčtový 86, 349
 - výsledku funkce 122
 - záznam 353
 - type 86
 - typy kompatibilní 278
 - totožné 278
- Údaje binární 9
 - dvojkové 9
 - ukazatel 281
 - umocňování 32, 65
 - unpack 163
 - until 71
 - úprava grafická 55
- Var 13, 25, 113
 - volání procedury 94
 - výraz 16, 113
 - celočíselný 16
 - logický 42, 43
 - množinový 274
 - reálný 31
 - výsledek funkce 122, 286
 - výstup
 - formátovaný 29, 33
 - neformátovaný 29, 33
 - vzorec Heronův 59
- While 68
 - with 207
 - write 247
 - pro celá čísla 20
 - – logické hodnoty 42
 - – obecný soubor 225
 - – reálná čísla 33
 - – znakové řetězce 28, 162
 - – znaky 79
 - writeln 14, 248
 - pro celá čísla 19
 - – logické hodnoty 42
 - – reálná čísla 33
 - – znakové řetězce 28, 162
 - – znaky 79
- Záhlaví procedury 94
 - záznam 188
 - zbytek po dělení 17
 - znak řídicí 77
 - způsob práce dávkový 251, 268
 - konverzační 8, 252

Ing. Tomáš Hruška, CSc.

PASCAL
PRO
ZAČÁTEČNÍKY

D T 800.92

681.3.04

Vydalo SNTL — Nakladatelství technické literatury, n. p.,
Spálená 51, 113 02 Praha 1

v roce 1989

jako svou 11 038. publikaci

Redakce teoretické literatury

Odpovědná redaktorka RNDr. Irena Bergerová

Vazbu navrhl Vladislav Jacák

Grafická úprava Ing. Tomáš Hruška, CSc.,

Technická redaktorka Viola Urbanová

Výtiskly Tiskařské závody, s. p., závod 2, Praha 1,
Karmelitská 6

368 stran, 38 obrázků, 14 ilustrací, 3 tabulky

Typové číslo L11-E1-V-85/12087 Vydání první

Náklad 35 000 výtisků. 18,55 AA, 19,19 VA

03/2

Cena vázaného výtisku Kčs 40,—

505/21, 856

Publikace je určena nejširší veřejnosti, tj. všem, kteří mají
zájem se naučit programovat v jazyku Pascal.

04-003-89

Kčs 40,—

