PRESEK
List za mlade matematike, fizike, astronome in računalnikarje

ISSN 0351-6652
Letnik **13** (1985/1986)
Številka 4
Strani 227–239

Bojan Mohar:

HISOFT PASCAL

Ključne besede: računalništvo, programski jeziki, pascal.

Elektronska verzija: http://www.presek.si/13/790-Mohar.pdf

HISOFT PASCAL

V prejšnji številki Preseka smo objavili kratka navodila za uporabo pascala na računalniku COMMODORE 64, za to številko pa smo vam pripravili navodila za uporabo prevajalnika na računalnikih ZX SPECTRUM, SHARP in AMSTRAD.

Prevajalnik za pascal, ki so ga napisali v softverski hiši Hisoft in nosi oznako HP4TM, je trenutno najboljši tovrstni izdelek za računalnike tipa ZX Spectrum (48 K). Razen na Sinclairjevih računalnikih deluje isti prevajalnik vsaj še na dveh tipih, ki sta precej razširjena tudi pri nas. To so računalniki Sharp serije MZ in računalniki znamke Amstrad (Schneider). Izvedbe na posameznih računalnikih se razlikujejo le v nekaterih nebistvenih malenkostih, zato bo na tem mestu opisana verzija za ZX Spectrum ravno tako dobro služila lastnikom Sharpov in Amstradov.

1. Uporaba sistema

K prevajalniku seveda sodi še celoten operacijski sistem. Z njegovo pomočjo pišemo in spreminjamo programe, pišemo na kaseto in beremo z nje, program prevedemo, izvedemo itd. Prevajalnik z dodatki zasede okrog 22 K pomnilniškega prostora, ostalo pa imamo na razpolago za izvorni in prevedeni program.

Prevajalnik dobimo na kaseti preveden v strojni jezik (CODE). Zato je na začetku dodan poseben basiški program za nalaganje, ki ga naložimo v pomnilnik z ukazom

```
LOAD " "
```

Ta program v bistvu sestavlja vrstica

```
LOAD " " CODE : RANDOMIZE USR 24598
```

Ko je prevajalnik naložen, nam najprej zastavi tri vprašanja:

```
Top of RAM?
Top of RAM for 'T'?
Table size?
```

Najbolje je, da vsakič pritisnemo ENTER. V tem primeru sistem sam določi vrednosti, po katerih nas je spraševal. Te vrednosti zadostujejo normalnim potrebam.

Po naših odgovorih sistem zapiše znak '>', kar pomeni, da čaka na naše ukaze. Le–te vpisujemo znak po znak. Pri vnašanju si lahko pomagamo z naslednjimi posebnimi tipkami:

ENTER	zaključi vrsto
^C	(= tipka CAPS SHIFT in 1) nas vrne v urejevalnik
^H	(= tipka DELETE) zbriše zadnji znak
^I	(= tipka CAPS SHIFT in 8) pomik na naslednji tabulator
^P	(= tipka CAPS SHIFT in 3) omogoči izpis na tiskalniku
^X	(= tipka CAPS SHIFT in 5) zbriše vneseni tekst v tekoči vrsti
^S	(= tipka CAPS SHIFT in presledek) ustavlja izpis

Za vsako od posebnih tipk je najprej navedena oznaka, ki jo bomo uporabljali v nadaljevanju, v oklepaju pa smo zapisali, katero kombinacijo znakov moramo pritisniti na tipkovnici ZX Spectruma. V začetku se nahajamo v načinu pisanja z velikimi črkami. Če želimo preiti na malo abecedo, pritisnemo

CAPS SHIFT in 2

Male črke navadno uporabljamo le v nizih, ker prevajalnik razlikuje male in velike črke v imenih. Tako na primer imeni A in a predstavljata različni imeni, beseda **while** pa ni isto kot rezervirana beseda **WHILE**. Kako program vnašamo in popravljamo, si bomo ogledali v naslednjem razdelku. Zaenkrat si mislimo, da imamo program že vnesen. Omeniti moramo, da so vrstice programa oštevilčene, tako kot pri basicu. Takoj na začetku povejmo, da **vse** besede pišemo znak po znak (npr. FOR). Isto velja tudi za kombinacije znakov, ki so v basicu dosegljive z eno tipko (>=, <=, <>).

Uporabljamo lahko naslednje ukaze:

Ukaz L (list): služi za izpis programa na zaslon. Če je program daljši od enega zaslona, se po izpisanem polnem ekranu izpisovanje ustavi. Če sedaj pritisnemo ^C, se vrnemo v sistem, ob pritisku na katerokoli drugo tipko pa se izpis nadaljuje. Ukaz L je oblike:

L n, m

kjer sta n in m celi števili. Pomeni, da se izpišejo vse vrstice s številkami med n

in m. Če prvi parameter n izpustimo, pomeni isto kot n = 1, izpuščeni drugi parameter m pa pomeni izpis do konca programa. Če napišemo le 'L', se izpiše cel program.

Ukaz P (put): z njim shranimo program na kaseto. Tam bo zapisan v posebnem HP4T formatu in ga bomo lahko prebrali le z ukazom G (glej dalje). Določimo lahko še začetno (n) in končno (m) številko vrstice, ki naj se shrani. Ukaz je oblike:

P n, m, ime

kjer je zadnji parameter zaporedje znakov, ki določijo ime programa na traku. Navednice v imenu niso potrebne. Primer:

P 1, 100, PROG2

Ukaz G (get): z njim beremo programe, zapisane v obliki HP4T. Ukaz je oblike:

G ,, ime ali le G

Pri prvi obliki sta vejici obvezni. Ime napišemo brez navednic. V drugem primeru se prebere prvi najdeni program v ustreznem formatu.

Tako med shranjevanjem (ukaz P) kot pri branju se med tem, ko računalnik piše oziroma bere, pojavi napis 'Busy..'. Ukaza P in G sta v pascalovem sistemu enakovredna ukazoma SAVE in LOAD basicovega operacijskega sistema. Če že imamo napisan del programa, se bo program, prebran z ukazom G, dodal na koncu in prejšnjega programa ne bomo izgubili.

Med tem, ko ukaz G išče začetek programa, lahko nalaganje prekinemo, če odtipkamo ^C (opomba: spomnimo se, da je to oznaka za CAPS SHIFT in 1).

Pri ukazih P in G lahko uporabljamo tudi imena oblike m:ime, kjer je m številka (mikro)tračnika. Tako na primer ukaz

P ,, 2: TEST

shrani na tračnik številka 2 program pod imenom TEST.

Ukaz C (compile): S tem ukazom prevedemo program. Parametri niso potrebni. Prevajalnik ob prevajanju izpisuje program na zaslon. Izpis lahko zaustavimo, če pritisnemo ^S. Zatem se lahko s ^C vrnemo v urejevalnik, če pa pritisnemo katerokoli drugo tipko, se prevajanje in izpisovanje nadaljujeta.

Če prevajalnik pri prevajanju odkrije napako v programu, se zaustavi in pod vrstico izpiše znak '↑' na mestu, kjer je odkril napako. Zatem izpiše še številko napake. (Na koncu poglavja je seznam napak z opisom.) Če sedaj pritisnemo tipko 'E', se prevajanje konča, sistem pa skoči v urejevalnik in že lahko popravljamo vrstico z napako. S pritiskom na 'P' preidemo v popravljanje vrstice, ki se nahaja neposredno pred vrstico, kjer je bila odkrita napaka, vse ostale tipke pa pomenijo nadaljevanje prevajanja.

Če je bilo prevajanje uspešno, se pojavi vprašanje

```
RUN ?
```

Če odgovorimo z 'Y', se prevedeni program takoj izvede. Ostali odgovori, tudi 'y', pa nas vrnejo v sistem.

Ukaz R (run): prevedeni program izvedemo.

Ukaz T (translate): Program se najprej prevede. Če je bilo prevajanje uspešno, se pojavi vprašanje 'OK?'. Odgovor 'Y' pomeni, naj se delo nadaljuje. Prevedeni program se najprej zapiše na tisto mesto v pomnilniku, kjer je bil doslej HP4T prevajalnik, zatem pa se skupaj s potrebnimi pascalovimi rutinami zapiše na trak. Pri tem vzame za ime na traku kar ime, ki smo ga nazadnje uporabili v ukazu 'F' (glej razdelek o urejevalniku). Ta program lahko kasneje naložimo in poženemo z

```
LOAD " " CODE : RANDOMIZE USR 24608
```

Ukaz 'T' uporabimo, ko je program popolnoma končan. Ker se pri tem prevajalnik v pomnilniku izgubi, ga moramo ponovno naložiti s kasete, če želimo nadaljevati z delom v pascalu.

Ukaz B (basic): nas vrne v basicov sistem. Če želimo od tod nazaj v pascal, odtipkamo

```
RANDOMIZE USR 24598
```

za vrnitev, pri kateri se naš program v pascalu izgubi, ali

```
RANDOMIZE USR 24603
```

če želimo ohraniti svoj program.

2. Vgrajeni urejevalnik

Program ima oštevilčene vrstice, tako kot programi v basicu. Vrstico lahko vnesemo na primer tako, da najprej napišemo njeno številko, zatem presledek, nazadnje pa vrstico pascalovega programa. Če smo imeli že prej vrstico z isto številko, obvelja nova. Če takoj za številko pritisnemo ENTER, se bo ustrezna vrstica izbrisala iz programa. Pri pisanju in spreminjanju obstoječega programa pa smemo uporabiti več ukazov, ki nam lahko precej olajšajo delo.

Ukaz I (insert): je oblike

I n, m

kjer sta n in m celi števili. Pomeni, da želimo vnašati vrstice začenši s številko n, vsaka naslednja pa ima za m večjo vrednost. Sistem poskrbi, da se na zaslon avtomatično zapišejo številke vrstic, ki so na vrsti za vnašanje. Vrste vpisujemo, dokler želimo, končamo pa z uporabo kontrolne funkcije ^C (= CAPS SHIFT in 1).

Ukaz D (delete): služi za brisanje vrstice. Z

D n, m

zbrišemo vse vrstice od n–te do vključno vrstice s številko m. Za brisanje ene same vrstice mora biti m = n.

Ukaz M (move): je oblike 'M n, m'. Pomeni, naj se vrstica n prepiše v vrstico m.

Ukaz N (renumber): ima tudi dva parametra:

N n, m

Povzroči, da se vrstice od n dalje ponovno oštevilčijo. Razlika med dvema zaporednima vrstama bo m.

Ukaz F (find): ima štiri parametre:

F n, m, f, s

Prva dva sta števili, ki določata območje vrstic, kjer bo ukaz deloval, zadnja dva

pa sta niza. Pomen: v vrsticah, določenih z n in m, iščemo niz f. Če ga najdemo, preidemo v popravljanje ustrezne vrstice in tedaj lahko niz f nadomestimo z nizom s (glej ukaza F in S za popravljanje vrstice).

Ukaz E (edit): je oblike

E n

in nas postavi v popravljanje vrstice s številko n. Vrstica se pojavi na ekranu in lahko jo spreminjamo. Pri tem imamo na voljo več podukazov. Najpomembnejši so naslednji:

presledek:		pomik kazalčka za eno mesto naprej.
DELETE:		zbriše znak levo od kazalčka.
ENTER	:	končaj s popravljanjem vrste.
Q	:	(quit) končaj s popravljanjem vrste, vendar ne upoštevaj popravkov.
K	:	zbriši znak pod kazalčkom.
Z	:	zbriši vse znake od kazalčka (vključno) do konca vrste.
F	:	poišči niz f iz zadnjega F ukaza.
S	:	nadomesti niz f z nizom s (glej ukaz F).
I	:	odslej vrivaj vtipkane znake na mestu kazalčka. Šele ko pritisneš ENTER, bodo črke spet dobile pomen podukazov.
X	:	pomakne kazalček na konec vrste in avtomatično vključi vrivanje znakov (glej podukaz I zgoraj).
C	:	odslej (do pritiska na ENTER) naj se vtipkani znaki vpisujejo v vrstico, ki jo popravljamo, vendar tako, da pišemo preko obstoječega besedila.

3. Razlike s standardnim pascalom

HP4T se v glavnem drži standardov. Pomembnejša odstopanja so naslednja:

— datoteke niso implementirane. Ravno tako ni podprogramov PUT in GET. Seveda pa imamo datoteki INPUT in OUTPUT, ki pomenita tipkovnico oziroma ekran,

— ni zapisov z inačicami,

— podprogramov ne moremo prenašati kot parametre,

— tip ALFA ni definiran,

— standardnih programov PACK, UNPACK, DISPOSE ni. Namesto zadnjega lahko uporabimo MARK in RELEASE.

4. Združljivost tipov

Recimo, da imamo v programu naslednjo deklaracijo:

```
VAR   A, B   :   ARRAY [ 1..10 ] OF REAL;
      C      :   ARRAY [ 1..10 ] OF REAL;
```

Potem sta A in B istega tipa, A in C pa ne (v strogem smislu, ki ga podpira tudi HP4T). Večina prevajalnikov za pascal gornjega primera ne bi razumela tako strogo kot Hisoftov pascal.

V enakem, strogem smislu je razumljena združljivost vseh tipov, ki v svoji definiciji vsebujejo **ARRAY** ali **RECORD.**

5. Nekateri dodatni podprogrami

Sestavljalci Hisoft pascala so se potrudili in standardnim podprogramom dodali še več novih. Nekatere, na primer funkcijo za računanje tangensa kota, lahko enostavno izrazimo s standardnimi podprogrami, mnogi pa so taki, da jih začetniki gotovo ne bodo potrebovali. Teh podprogramov tu ne bomo opisovali. Omenili bomo le najpomembnejše.

Funkcija **INCH** vrne znak, ki je bil pritisnjen na tipkovnici. Če nismo pritisnili nobene tipke, vrne CHR(0). Na primer:

```
REPEAT CH := INCH UNTIL CH <> CHR(0);
```

čaka, da pritisnemo katerokoli tipko. Po pritisku je ustrezni znak shranjen v spremenljivki CH.

Funkcija **FRAC(X)** vrne neceli del realnega števila X.

Za generiranje slučajnih števil imamo na voljo funkcijo **RANDOM,** ki je brez parametrov. Njen rezultat je naključno celo število med 0 in 255. Omeniti moramo, da ta generator slučajnih števil ni ravno najboljši za profesionalne programe, bo pa dovolj dober za učenje ali za pisanje igric.

Za delo s kazalci služita prava podprograma **MARK(P)** in **RELEASE(P).** Prvi shrani naslov vrha sklada v kazalec P. Sklad se širi pri dinamični uporabi pomnilnika s pomočjo podprograma NEW. Če kasneje napravimo RELEASE(P) z istim in medtem nespremenjenim kazalcem P, se sklad zmanjša do nivoja ob času, ko je bil napravljen MARK(P).

Omenimo še nekaj podprogramov, ki bodo uporabni le pri vtikanju v raču-

nalnikov sistem. Podprogram POKE (naslov, vrednost) in funkcija PEEK (naslov, tip) sta podobna kot v basicu. Pri PEEK moramo dodati ime tipa, odvisno od tega, kakšen rezultat želimo. Če ima tip v svoji predstavitvi več kot en zlog, se vzamejo še vrednosti v zaporednih celicah od danega naslova dalje.

Podprogram INLINE (c_1, c_2, ...) ima poljubno mnogo parametrov, ki pomenijo celoštevilske vrednosti ukazov v strojnem jeziku, ki naj se na tem mestu vstavijo v prevedeni program.

Podprogram OUT (vrata, znak) izpiše znak na izhodna vrata z dano številko.

Datoteke nam delno nadomeščata ukaza

TOUT (ime datoteke, začetek, velikost)

za izpis na trak in

TIN (ime datoteke, začetek)

za branje zapisa, predhodno napravljenega s TOUT. Ime datoteke je tipa **ARRAY** [1..8] **OF** CHAR, začetek pa je celo število, ki pomeni naslov v pomnilniku, od koder se pričnejo jemati zlogi, ki se s TOUT izpisujejo na trak. Pri tem nam prideta zelo prav funkciji ADDR(v) in SIZE(v), ki nam vrneta začetni naslov in velikost prostora v pomnilniku, ki pripada spremenljivki v. Pri TIN pa se vrednosti s traku začnejo vpisovati v pomnilnik na naslovu začetek. Tretji parameter pri TOUT pove, koliko zaporednih zlogov naj se zapiše na trak.

6. Dodatek – želvina grafika

Skupaj s prevajalnikom lahko dobimo še paket podprogramov, ki omogoča delo v grafiki visoke ločljivosti. Sistem risanja je t.i. želvina grafika, kot jo poznamo iz jezika LOGO.

Če so podprogrami zapisani na kaseti takoj za HP prevajalnikom, napravimo naslednje. Po nalaganju prevajalnika ustavimo kaseto, zatem pa zahtevamo, da sistem naloži podprograme:

G , , TURTLE ali samo G

Naloženi del programa navadno zasede vrstice s številkami od 10 do 1350, odvisno od verzije, ki jo imamo. Če ga želimo izvajati, ga moramo najprej dopol-

niti do popolnega programa. V vrsticah 1 – 9 vpišemo glavo programa, deklaracije konstant, tipov in spremenljivk. Tudi če naš program nima svojih spremenljivk, moramo napisati vsaj simbol **VAR**, ker so v vrstici 10 paketa TURTLE deklarirane nekatere globalne spremenljivke, ki sodijo v deklaracije spremenljivk. Na koncu, v vrsticah od 1360 dalje, pa dodamo še svoje podprograme in seveda glavni program. Ob zaključku tega razdelka, ko bomo sestavili dva zgleda, bomo zapisali le dodatne vrstice, ki jih moramo sami vnesti.

Pri grafiki visoke ločljivosti je zaslon razdeljen na 256 x 176 točk. Vsaka točka ima koorditati (x, y), kjer x pomeni oddaljenost od levega, y pa razdaljo od zgornjega roba. Gornji levi vogal ima koordinati (0,0), sredina ekrana pa (127, 87). Pri želvini grafiki si lahko predstavljamo, da je na takem ekranu v neki točki majhna želva, ki se lahko premika sem ter tja. Pri vsakem pomiku ta želva za seboj pusti sled – narisano črto. S pomikanjem v različnih smereh lahko taka živalca nariše vse mogoče slike.

Podprogrami dodatka TURTLE opisujejo želvo s štirimi parametri. To so štiri globalne spremenljivke, ki so deklarirane povsem na začetku paketa:

XCOR in **YCOR:** v teh spremenljivkah sta shranjeni koordinati mesta, kjer se nahaja želva. Tako lahko z:

```
XCOR := 0; YCOR := 0
```

postavimo želvo v zgornji levi kot.

HEADING: je realna vrednost, ki določa smer (v kotnih stopinjah), kamor je obrnjena želva. Vrednost 0 pomeni vzhod (to je smer od leve proti desni), vrednost 90 pa sever (navzgor).

PENSTATUS: določa stanje 'peresa'. Če ima ta spremenljivka vrednost 0, je pero spuščeno in pri premikanju bo želva risala črto. Če pa je 1, je pero dvignjeno in želva za seboj ne pušča sledi.

Pri programiranju nam ni treba skrbeti za vrednosti zgornjih količin. Za njihovo spreminjanje imamo na voljo podprograme. Oglejmo si jih.

TURTLE: postavi želvo na sredino ekrana in ji da smer od leve proti desni. Pero je spuščeno, ekran moder, črte pa bodo rumene. Navadno ta podprogram pokličemo na začetku programa.

SETXY (x, y: REAL): postavi želvo na točko (x, y).

SETHD (a: REAL): definira smer želve (HEADING).

PENUP: dvigne pero.

PENDOWN (c: INTEGER): spusti pero in določi barvo črt (c).

INK (c: INTEGER): določi barvo peresa.

PAPER (c: INTEGER): določi barvo ozadja.

FWD (d: REAL): ta podprogram pomakne želvo za d enot v smeri, kamor želva trenutno gleda. Če je pero spuščeno, se pri tem nariše črta.

BACK (d: REAL): pomik nazaj za d enot. Smer želve se pri tem ne spremeni.

TURN (a: REAL): zasuk želve za a stopinj v levo.

LEFT (a: REAL): isto kot TURN(a).

RIGHT (a: REAL): isto kot LEFT (–a).

VECTOR (a, d: REAL): najprej postavi želvo v smeri a, zatem pa jo pomakne za d enot. Ekvivalentno z SETHD (a); FWD (d).

ARCR (r: REAL; a: INTEGER): želva se s tem ukazom pomakne po loku (in ne po ravni črti) kroga velikosti r ($r = \pi = 3.14159...$ pomeni, da je polmer kroga enak 180 enot). Dolžina loka, ki ga pri tem opiše želva, je enaka a, merjeno v stopinjah kota, ki ga določa lok s središčem kroga. Lok se začne risati v smeri, kamor trenutno gleda želva, in v smeri, nasprotni smeri urinega kazalca. Primer: če želimo narisati krog polmera R s središčem v točki (X, Y), lahko napravimo naslednje:

```
SETXY (X, Y + R);        {postavimo želvo na krožnico}
SETHD (90);              {želvo obrnemo v smeri tangente}
ARCR (R* 3.14159 / 180, 360)
```

Na koncu je želva obrnjena navzgor.

COPY: prenese sliko z ekrana na ZX tiskalnik.

Obstajajo še podprogrami PLOT, LINE, SPOUT, ki so podobni ustreznim basicovim ukazom in jih ne bomo opisovali.

Za konec si oglejmo še dva zgleda. Prvi program sestavi zanimiv lik iz kvadratov. Pri N = 4 je ta lik videti kot okno, pri večjih vrednostih pa kot nekakšna rozeta.

```
   1 PROGRAM zasukanikvadrati;
   2 {nariši N kvadratov s skupno točko}
   3 VAR    N, I :   INTEGER;
   4         ALFA:  REAL;
  10      .....
  20
    .
    .      vrstice paketa TURTLE
    .
1350      .....
1400 PROCEDURE KVADRAT (A: REAL);
```

```
1410 {želva prehodi kvadrat s stranico A }
1420 BEGIN FWD(A); TURN (90); FWD(A); TURN (90);
1430       FWD(A); TURN (90); FWD(A); TURN (90);
1440 END;
1500 BEGIN {glavni program }
1510       WRITE ('Vnesi stevilo kvadratov: ');
1520       READ(N);
1530       ALFA := 360/N;
1540       TURTLE ; {postavi želvo na sredino}
1550       FOR I:= 1 TO N DO BEGIN
1560             KVADRAT (50); TURN (ALFA)
1570       END
1580 END.
```

Drugi program pa nam bo pokazal, kako narišemo kvadratno "spiralo". Zaradi enostavnosti je zgled sestavljen tako, da program riše, dokler se ne ustavi, ker črte padejo iz zaslona.

```
   1 PROGRAM spirala;
   2 {risanje kvadratne spirale }
   3 VAR D:     INTEGER;
 ...
1400 BEGIN { glavni program }
1410       D := 0;
1420       TURTLE;
1430       WHILE TRUE DO BEGIN
1440             D := D + 5;
1450             FWD (D);
1460             TURN (90)
1470       END
1480 END.
```

Na naslednji strani je v razdelku

7. Seznam napak pri prevajanju

1. Preveliko število
2. Manjka podpičje
3. Nedeklarirana spremenljivka
4. Tu mora biti ime
5. Uporabi '=' namesto ':=' v deklaraciji konstant
6. Manjka '='
7. To ime ne sme stati na začetku stavka
8. Manjka ':='
9. Manjka ')'
10. Napačen tip
11. Manjka '.'
12. Moral bi biti faktor
13. Morala bi biti konstanta
14. To ime ni konstanta
15. Manjka 'THEN'
16. Manjka 'DO'
17. Manjka 'TO' ali 'DOWNTO'
18. Manjka '('
19. Izraza tega tipa se ne da izpisati
20. Manjka 'OF'
21. Manjka ','
22. Manjka ':'
23. Manjka 'PROGRAM'
24. Tu mora biti spremenljivka, ker je var parameter
25. Manjka 'BEGIN'
26. Pri READ mora biti spremenljivka
27. Izrazov tega tipa ne moremo primerjati
28. Mora biti tip INTEGER ali REAL
29. Spremenljivke tega tipa se ne da prebrati.
30. To ime ni tip
31. Pričakovan eksponent v realnem številu
32. Pričakovan skalarni in ne številski izraz
33. Prazen niz ni dovoljen (uporabi CHR(0))
34. Manjka '['
35. Manjka ']'
36. Indeksi morajo biti skalarnega tipa
37. Manjka '..'
38. Pri deklaraciji tabele manjka ']' ali ','
39. Spodnja meja je večja od zgornje
40. Prevelike množice (več kot 256 elem.)
41. Tip funkcije mora biti definiran z imenom tipa
42. V množici manjka ',' ali ']'
43. V množici manjka '..' ali ',' ali ']'
44. Tip parametra mora biti definiran z imenom tipa
45. Prazna množica ne sme biti prvi faktor v tem izrazu
46. Manjka skalarna vrednost (lahko tudi REAL)
47. Manjka ordinalna vrednost (brez REAL)
48. Teh množic ne moremo primerjati
49. Z znakoma '<' in '>' ne moremo primerjati množic
50. Manjka 'FORWARD', 'LABEL', 'CONST', 'VAR', 'TYPE' ali 'BEGIN'
51. Tu mora biti šestnajstiško število
52. Množice ne moremo vpisati v spomin z ukazom POKE
53. Tabela prevelika (večja kot 64K)
54. V deklaraciji zapisa manjka 'END' ali ';'
55. Tu bi moralo biti ime polja v zapisu
56. Za WITH mora biti spremenljivka
57. Spremenljivka v stavku WITH mora biti zapis
58. Ime polja ni v stavku WITH
59. Za simbolom 'LABEL' mora biti nepredznačeno celo število
60. Za 'GOTO' mora biti nepredznačeno celo število
61. Oznaka je na napačnem nivoju
62. Nedeklarirana oznaka
63. Parameter podprograma SIZE mora biti spremenljivka
64. Kazalca lahko primerjamo le z '=' ali '<>'
67. Format za izpis celega števila je e:n ali e:n:H
68. V nizu ne sme biti znak za konec vrste
69. Parameter pri NEW, MARK in RELEASE mora biti kazalec
70. Parameter podprograma ADDR mora biti spremenljivka

8. Seznam napak pri izvajanju

Če pri izvajanju programa pride do napake, pascal izpiše eno od dolnjih sporočil, zatem pa doda še "at PC=xxxx". V xxxx je zapisan naslov v pomnilniku, kjer je napaka. Če želimo ugotoviti, v katerem delu programa je ta napaka, poiščemo naslov xxxx v izpisu programa, ki je bil narejen pri prevajanju.

1. Stop
2. Preveliko število
3. Premajhen pomnilnik
4. Deljenje z nič (lahko tudi pri DIV ali MOD)
5. Premajhen indeks
6. Prevelik indeks
7. Napaka pri klicu matematične funkcije
8. Preveliko število
9. Moralo bi biti število
10. Predolga vrsta
11. Manjka eksponent

Bojan Mohar