FAKULTET PRIRODOSLOVNO MATEMATIČKIH ZNANOSTI I
ODGOJNIH PODRUČJA SVEUČILIŠTA U SPLITU

PRIMJENA RAČUNALA U NASTAVI

Seminarski rad:

UČITI PRIMJENOM SUSTAVA TEx-Sys?

Studenti:
Tina Grga
Marija Katić

Mentor:
docent dr. sc. Slavomir Stankov, prof

Smjer:
Matematika-Informatika

U Splitu, siječanj, 2004. god.

CILJ:

Ispitati djelotvornost inteligentne hipermedijske autorske ljuske TEx-Sys u učenju programskog jezika Pascal u odnosu na klasično predavanje i "učenje putem otkrivanja", koristeći modul "Quiz" danog sustava.

1. UVOD

Primjena računala u nastavi postaje neizostavan dio nastavnog procesa učenja i poučavanja. Jedan od načina savladavanja gradiva uz pomoć računala je i primjena inteligentne hipermedijske autorske ljuske TEx-Sys namijenjene gradnji inteligentnih tutorskih sustava u po volji odabranom područnom znanju. Jasno je da pomoću takvog sustava gradivo možemo savladavati bilo gdje i bilo kada, kao i brzinu učenja prilagođavati sposobnostima, no upitan je učinak takvog učenja u odnosu na učenje popraćeno predavanjem "živog" učitelja. Da bismo upotpunili takvu usporedbu odlučili smo se dodati još jedan oblik učenja koji se jednim svojim dijelom oslanja na sustav TEx-Sys.

Znanje u sustavu TEx-Sys je pohranjeno u "bazama znanja". Pod pojmom "baza znanja" podrazumijevajmo bazu koja sadrži neko područno znanje.

Osnovni problem za razumijevanje i rad u TEx-Sys-u je pravilno uočavanje pojmova i njihovih svojstava te odnosa među njima, u znanju kojeg želimo prikazati. Radi jednostavnijeg rada pojmove ćemo zvati "čvorovima", a odnose među njima "vezama". Za provođenje cilja koristili smo bazu znanja "Pascal.kb", područnog znanja o programskom jeziku Pascal, odnosno sadržaj kolegija "Pascal" koji se sluša na prvoj godini Fakulteta PMZiOP u Splitu. Baza se sastoji od 104 čvora, 105 veza i 95 hipermedijskih dodataka.

2. STRUKTURA

U navedenom ispitivanju sudjeluje 15 studenata IV. godine sa smjerova:
Matematika: 2 studenta,
Matematika-Informatika: 4 studenta i
Informatika-Tehnička Kultura: 9 studenata,
koji su podijeljeni u tri skupine od po pet studenata metodom slučajnog odabira.

U savladavanju gradiva iz programskog jezika Pascal jedna skupina studenata primjenjuje sustav TEx-Sys koristeći znanje koje je pohranjeno u bazi znanja "Pascal.kb", dok druga skupina ima uvid samo u popis čvorova dane baze (učenje putem otkrivanja). I konačno, u trećoj skupini sustav Tex-Sys zamjenjuje "živi" učitelj prezentirajući tekstualni opis područnog znanja.

Nakon savladavanja gradiva provodi se testiranje u sve tri skupine koristeći modul "Quiz" (Kviz) sustava Tex-Sys.

Napomena.
Kao što je u uvodu već rečeno, studenti su slušali kolegij Pascal na prvoj godini pa se predstavljanje znanja u sve tri skupine uzima kao okvirni podsjetnik.

2.1. SKUPINA 1

Koristeći modul "Learning and Teaching" (modul za učenje i poučavanje) sustava TEx-Sys za prezentaciju baze znanja "Pascal.kb", studenti uče programski jezik Pascal. Tijekom rada učitelj je na raspolaganju svim studentima ukoliko se pojave neke nejasnoće. Vrijeme učenja je 45 minuta.

2.2. SKUPINA 2

Formalizacija tekstualnog opisa područnog znanja (ispis čvorova baze znanja) slijedi princip hijerarhijske dekompozicije isključujući veze među čvorovima. Potrebno je otkriti odnose među čvorovima (veze) za uspješno savladavanje gradiva programskog jezika Pascal.

Učitelj princip dekompozicije pojmova studentima objašnjava na konkretnom primjeru (Primjer 2.1.1.).

Primjer 2.1.1.

1) Funkcije
2) Omogućavaju rastavljanje programa u jednostavnije i razumljivije logičke cjeline koje pozivamo na izvršavanje njihovim imenom
3) Struktura
4) Deklarativni dio
5) Blok
6) Obuhvaća ime procedure ili funkcije i popis parametara preko kojih je ista povezana sa okolinom
7) Naziv
8) Popis parametara
9) Obuhvaća opis podataka i opis akcija
10) Izrazi
11) Naredbe
12) Varijable

Pomoću danih pojmova se formiraju sljedeće rečenice:

- Omogućavaju rastavljanje programa u jednostavnije i razumljivije logičke cjeline koje pozivamo na izvršavanje njihovim imenom **je TEMELJNA ZADAĆA** Funkcija.
- Struktura **je IZGLED** Funkcije.
- Deklarativni dio **je DIO** Strukture.
- Blok **je DIO** Strukture.
- Obuhvaća ime procedure ili funkcije i popis parametara preko kojih je ista povezana sa okolinom **je TEMELJNA ZNAČAJKA** Deklarativnog dijela**.**
- Naziv **je DIO** Deklarativnog dijela.
- Popis parametara **je DIO** Deklarativnog dijela.
- Obuhvaća opis podataka i opis akcija **je TEMELJNA ZNAČAJKA** Bloka.
- Izrazi, Naredbe, Varijable **su DIO** Bloka.

Ispis čvorova koje studenti primjenjuju za učenje programskog jezika Pascal se nalazi u Prilogu 1.

Tijekom rada učitelj je na raspolaganju svim studentima ukoliko se pojave neke nejasnoće. Vrijeme učenja je 45 minuta.

Napomena.
Budući da ispis čvorova prati broj puta pojavljivanja istog u danoj bazi neki se čvorovi pojavljuju više puta.

2.3. SKUPINA 3

Učitelj prezentira tekstualni opis područnog znanja iz baze znanja "Pascal.kb". Kroz predavanje su objašnjeni svi pojmovi iz dane baze, a naglašene su i veze među njima. Tekst predavanja se nalazi u Prilogu 2. Vrijeme izlaganja je 45 minuta.

3. TESTIRANJE

Testiranje se provodi u sve tri skupine koristeći modul "Quiz" (Kviz) sustava TEx-Sys. Cilj ovog modula je testirati znanje učenika u poznavanju strukture područnog znanja. Nakon završetka testa u ovom modulu možemo pregledati rezultate. Kviz znanja u sustavu TEx-Sys sastoji se od odgovaranja na serije od dva pitanja. Svako pitanje ima ponuđeno više odgovora od kojih su jedno ili više njih točni. Odgovor može biti točan,

polutočan ili netočan. Odgovor na pitanje je pogrešan ako je izabran jedan ili više pogrešnih odgovora. Pitanja su razvrstana u tri težinske kategorije. Pitanja iz prve kategorije su najlakša, dok su iz treće najteža. Studentima se daje na raspolaganje 30 minuta za rješavanje kviza.

3.1. REZULTATI TESTIRANJA

Rezultati testiranja su prikazani u tablicama 3.1.1., 3.1.2., 3.1.3..

Tablica 3.1.1.

SKUPINA 1:			**DATUM:** 13. 01. 2004. god.				
STUDENT	**POČETAK**	**KRAJ**	**BODOVI**	**OCJENA**	**KAT. 1**	**KAT. 2**	**KAT. 3**
Student 1	9 : 33	9 : 41	58 / 58	odličan	0	1	9
Student 2	9 : 33	9 : 42	38,5 / 58	dobar	0	1	9
Student 3	9 : 33	9 : 40	28 / 56	dovoljan	0	2	8
Student 4	9 : 33	9: 42	42 / 56	vrlo dobar	0	2	8
Student 5	9 : 33	9 : 44	13 / 34	dovoljan	4	5	1
Prosječno vrijeme rješavanja kviza: 9 minuta Prosječna ocjena: dobar (3,2)							

Tablica 3.1.2.

SKUPINA 2:			**DATUM:** 13. 01. 2004. god				
STUDENT	**POČETAK**	**KRAJ**	**BODOVI**	**OCJENA**	**KAT. 1**	**KAT. 2**	**KAT. 3**
Student 1	9 : 33	9 : 43	13 / 34	dovoljan	3	7	0
Student 2	9 : 33	9 : 46	48 / 52	odličan	1	2	7
Student 3	9 : 33	9 : 47	14 / 32	dovoljan	4	6	0
Student 4	9 : 33	9: 51	12 / 28	dovoljan	6	4	0
Student 5	9 : 33	9 : 47	13 / 34	nedovoljan	1	2	4
Prosječno vrijeme rješavanja kviza: 14 minuta Prosječna ocjena: dovoljan (2,4)							

Tablica 3.1.3.

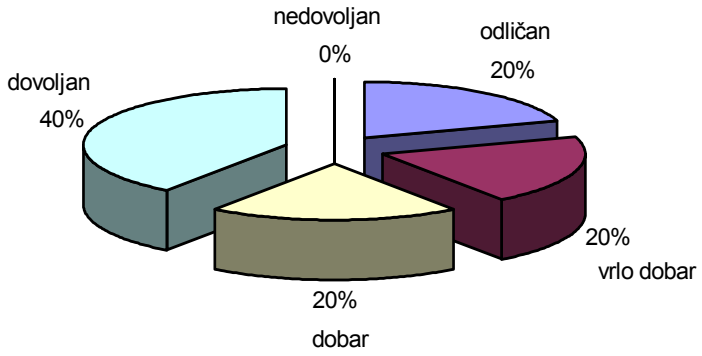
SKUPINA 3:			**DATUM:** 12. 01. 2004. god.				
STUDENT	**POČETAK**	**KRAJ**	**BODOVI**	**OCJENA**	**KAT. 1**	**KAT. 2**	**KAT. 3**
Student 1	13:11	13:18	55 / 58	odličan	0	1	9
Student 2	13:11	13:25	50,5 / 58	vrlo dobar	0	1	9
Student 3	13:11	13:23	4 / 16	nedovoljan	4	2	0
Student 4	13:11	13:30	9 / 26	nedovoljan	5	4	0
Student 5	13:11	13:22	48 / 56	vrlo dobar	0	2	8
Prosječno vrijeme rješavanja kviza: 13 minuta Prosječna ocjena: dobar (3,0)							

Radi jednostavnije usporedbe rezultata napravili smo nekoliko grafova. Rezultate smo obradili u odnosu na ocjenu i težinsku kategoriju (količina posjećenosti). Vidi slike 3.1.1., 3.1.2., 3.1.3.

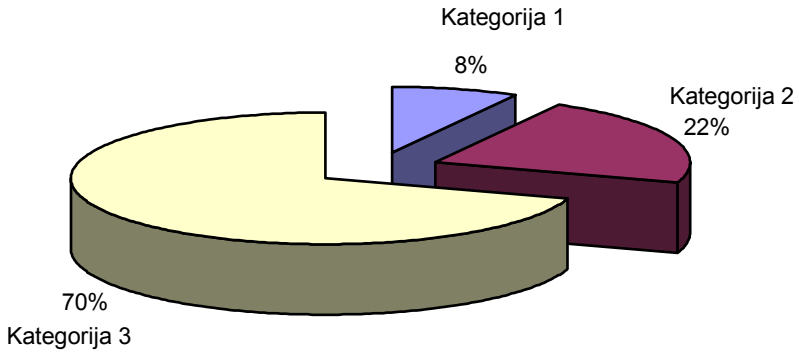
Slika 3.1.1.

SKUPINA 1:

❑ ocjene



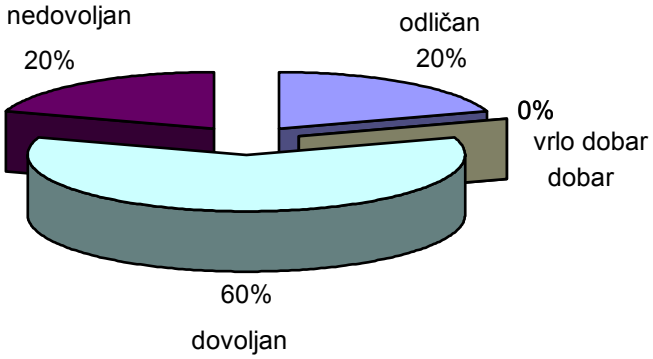
❑ kategorije



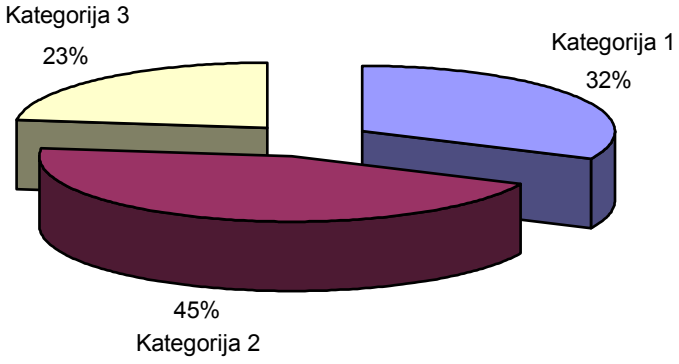
Slika 3.1.2.

SKUPINA 2:

❑ ocjene



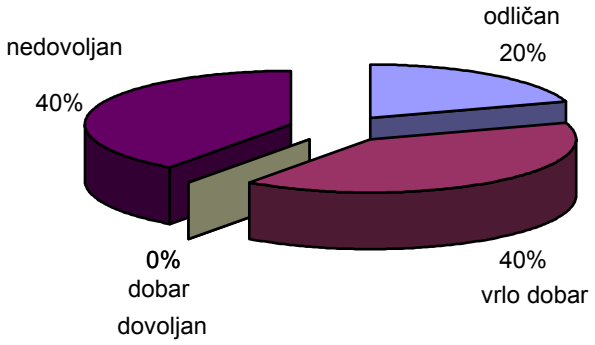
❑ kategorije



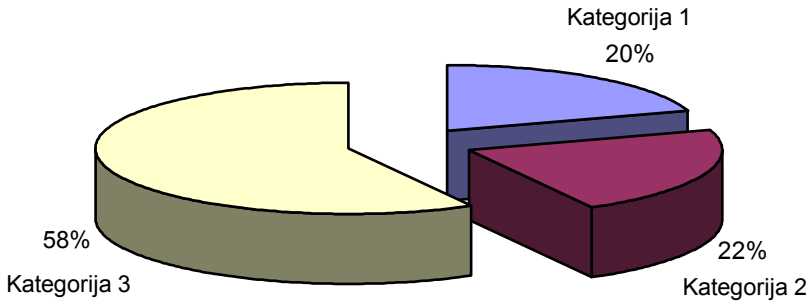
Slika 3.1.3.

SKUPINA 3:

❑ ocjene



❑ kategorije



4. ZAKLJUČAK

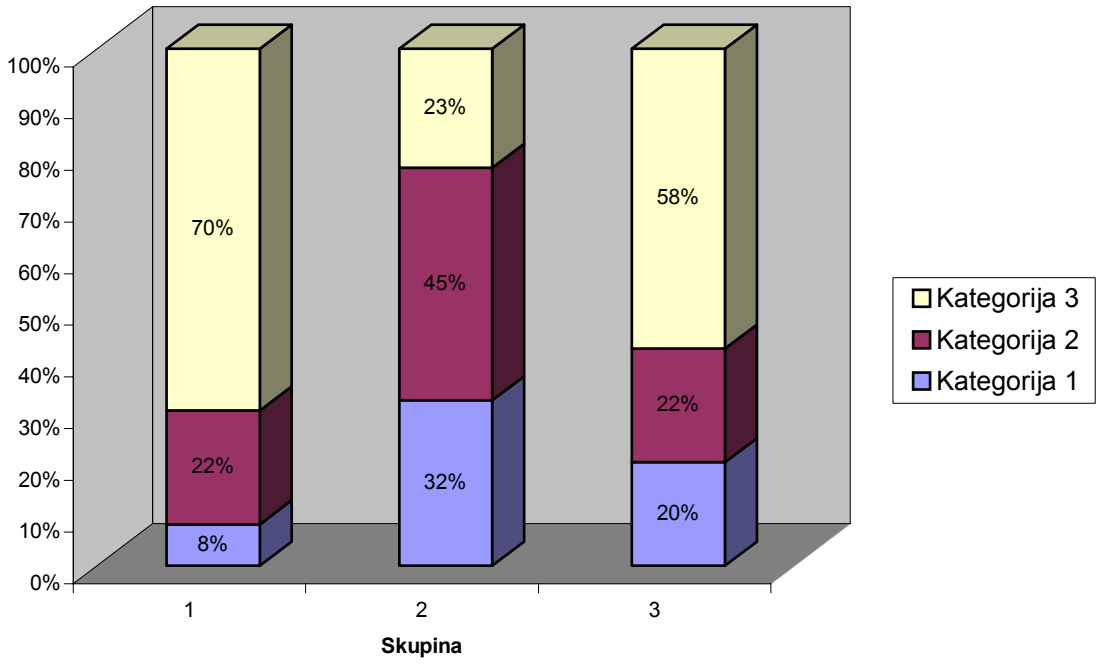
Uspoređujući rezultate svih triju skupina zaključijemo da je najbolje rezultate postigla Skupina 1, odnosno skupina koja je učila primjenom sustava TEx- Sys. Iako je prosječna ocjena bila "dobar" i u Skupini 1 i u Skupini 3, treba uzeti u obzir i kategorije u kojima su studenti odgovarali na pitanja. Naime, pitanja u trećoj kategoriji su teža od onih u prvoj, odnosno drugoj kategoriji. Dakle, najbolji rezultat su postigli oni studenti koji su bili u kategoriji 0- 1- 9 (0- 2- 8), a takvih je bilo najviše u Skupini 3 (čak 80%).

Skupina 2, tj. skupina koja je za učenje koristila samo popis čvorova iz baze znanja "Pascal.kb" postigla je najlošije rezultate.

Pokazalo se da je učiti primjenom sustava TEx-Sys korisno, pa možemo potvrdno odgovoriti na postavljeno pitanje.

Usporedba rezultata svih triju skupina prikazana je na sljedećoj slici 4.1.:

Slika4.1.



5. LITERATURA

- **Z. Vlašić, Pascal (priručnik s rješenim primjerima);**
- **I. Bratko, V. Rajković, Računarstvo s programskim jezikom Pascal;**
- **A. Lovrić, Pascal**
- **http://mapmf.pmfst.hr/~stankov/texsys/index.html**

PRILOG 1

ISPIS ČVOROVA BAZE ZNANJA "Pascal.kb"

1) Pascal
2) Programski jezik namijenjen komuniciranju između ljudi i računala
3) Viši programski jezik kreiran na postavkama strukturalnog programiranja namijenjen komuniciranju između ljudi i računala
4) Omogućava opis problema, te opis postupaka za njegovo rješavanje kao i opis polaznih podataka, rezultata i opis relacija medu njima
5) Program
6) Struktura programa u Pascalu
7) Zaglavlje programa (HEADING)
8) Blok programa (BLOCK)
9) Obuhvaća ime programa i popis parametara preko kojih je program povezan sa okolinom
10) Obuhvaća opis podataka i opis akcija
11) Definicija konstanti
12) Deklaracija oznaka
13) Deklaracija varijabli
14) Naredbe
15) Funkcije
16) Procedure
17) Tipovi podataka
18) Omogućava definiciju konstanti proizvoljnih imena koje će se koristiti u programu
19) Najavljivanje oznaka koje će se koristiti za označavanje naredbi kod upotrebe bezuvjetnog skoka (GOTO)
20) Najavljuje varijable koje će se koristiti u programu
21) Opisuju akcije koje će se izvršiti za vrijeme rada programa
22) Jednostavne naredbe
23) Strukturalne naredbe
24) Omogućavaju rastavljanje programa u jednostavnije i razumljivije logičke cjeline koje pozivamo na izvršavanje njihovim imenom
25) Struktura
26) Omogućavaju rastavljanje programa u jednostavnije i razumljivije logičke cjeline koje pozivamo na izvršavanje njihovim imenom
27) Struktura
28) Procedure s okolinom povezane preko formalnih parametara
29) Procedure s okolinom povezane preko globalnih varijabli
30) Procedure s okolinom povezane preko globalnih varijabli i formalnih parametara
31) Konačni skupovi vrijednosti za koje vrijede neka zajednička svojstva
32) Jednostavni tipovi podataka
33) Složeni tipovi podataka
34) Pokazivači (POINTERS)
35) Bezuvjetni skok

36) Naredba pridruživanja
37) Prazna naredba
38) Složena naredba
39) Uvjetne naredbe
40) Naredbe za ponavljanje
41) Deklarativni dio
42) Blok
43) Deklarativni dio
44) Blok
45) Korisnički tipovi podataka
46) Standardni tipovi podataka
47) Datoteka (FILE)
48) Niz (ARRAY)
49) Skup (SET)
50) String
51) Zapis (RECORD)
52) Omogućavaju pozivanje komponenata dinamičkih struktura koje nemaju eksplicitna imena kao što ih imaju druge vrste programskih varijabli, kao i uspostavljanje relacija između podataka
53) GOTO naredba
54) CASE naredba
55) IF naredba
56) FOR naredba
57) REPEAT naredba
58) WHILE naredba
59) Obuhvaća ime procedure ili funkcije i popis parametara preko kojih je ista povezana sa okolinom
60) Naziv
61) Popis parametara
62) Obuhvaća opis podataka i opis akcija
63) Izrazi
64) Naredbe
65) Varijable
66) Obuhvaća ime procedure ili funkcije i popis parametara preko kojih je ista povezana sa okolinom
67) Naziv
68) Popis parametara
69) Obuhvaća opis podataka i opis akcija
70) Izrazi
71) Naredbe
72) Varijable
73) Intervalni (SUBRANGE) tipovi podataka
74) Pobrojani (ENUMERATED) tipovi podataka
75) Cjelobrojni (INTEGER) tipovi podataka
76) Logički (BOOLEAN) tipovi podataka
77) Realni (REAL) tipovi podataka

78) Znakovni (CHAR) tipovi podataka
79) Objedinjuje skup podataka koji su pohranjeni u pomoćnu memoriju
80) Direktna datoteka
81) Indeksno-sekvencijalna datoteka
82) Sekvencijalna datoteka
83) Akcija nad datotekom
84) Inicijalizacija datoteke
85) Otvaranje datoteke
86) Zatvaranje datoteke
87) Pod istim imenom objedinjuje čitav skup varijabli, koje se međusobno razlikuju samo po različitim indeksima
88) Jednodimenzionalni niz
89) Višedimenzionalni niz
90) Zamjenjuje složene logičke izraze, jer su operacije sa skupom puno brze
91) Obično se upotrebljava za podatke tekstualnog tipa, te kroz niz funkcija i procedura omogućava mnogobrojne operacije nad tim podacima
92) Concat
93) Copy
94) Delete
95) Insert
96) Length
97) Pos
98) Str
99) Objedinjuje skup podataka koji u jednom zapisu mogu biti različitih tipova
100) Parametri
101) Operandi
102) Operatori
103) Opisuju akcije koje će se izvršiti za vrijeme rada programa
104) Jednostavne naredbe
105) Strukturalne naredbe
106) Globalne varijable
107) Lokalne varijable
108) Parametri
109) Operandi
110) Operatori
111) Opisuju akcije koje će se izvršiti za vrijeme rada programa
112) Jednostavne naredbe
113) Strukturalne naredbe
114) Globalne varijable
115) Lokalne varijable
116) Read
117) Write
118) Assign
119) Append
120) Reset
121) Rewrite

122) Close
123) Formalni parametri
124) Stvarni parametri
125) Varijabilni parametri
126) Vrijednosni parametri
127) Bezuvjetni skok
128) Naredba pridruživanja
129) Prazna naredba
130) Složena naredba
131) Uvjetne naredbe
132) Naredbe za ponavljanje
133) GOTO naredba
134) CASE naredba
135) IF naredba
136) FOR naredba
137) REPEAT naredba
138) WHILE naredba

PRILOG 2

TEKST PREDAVANJA

PASCAL

(Literatura: Z. Vlašić –Pascal (priručnik s riješenim primjerima))

Pascal u užem smislu je programski jezik namijenjen komunikaciji između ljudi i računala, a u širem smislu to je viši programski jezik koji je kreiran na postavkama strukturalnog programiranja namijenjen komunikaciji između ljudi i računala. Temeljna zadaća Pascal-a je da omogućava opis problema, te opis postupaka za njegovo rješavanje kao i opis polaznih podataka, rezultata i relacija među njima.

U Pascal-u kreiramo programe. Svaki program se sastoji od zaglavlja (HEADING) i bloka (BLOCK).

Zaglavlje programa (HEADING)

Obuhvaća ime programa i popis parametara preko kojih je program povezan s okolinom.

Blok programa (BLOCK)

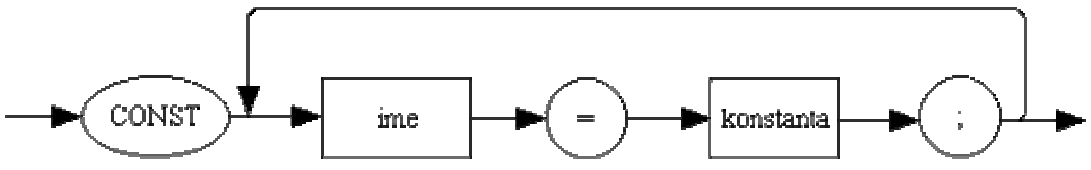
U bloku se nalazi opis podataka i opis akcija. Svi objekti koji se koriste u programu moraju prethodno biti deklarirani u pojedinim odijelcima bloka.

Dijelovi bloka su:

1. Definicija konstante
2. Deklaracija oznaka
3. Deklaracija varijabli
4. Naredbe
5. Funkcije
6. Procedure
7. Tipovi podataka.

1. Definicija konstante.

Za pojedine konstante uvodimo ime koje upotrebljavamo umjesto konstante. Sintaksa definicije konstanti je prikazana na sljedećoj slici:

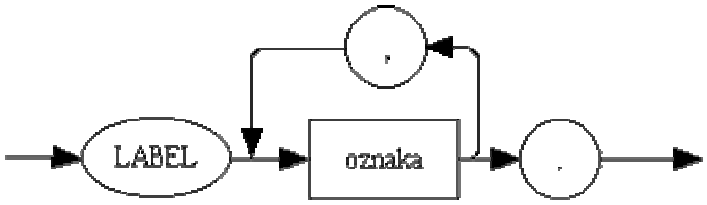


Primjer: CONST pi=3.14159;

max=80;

2. Deklaracija oznaka.

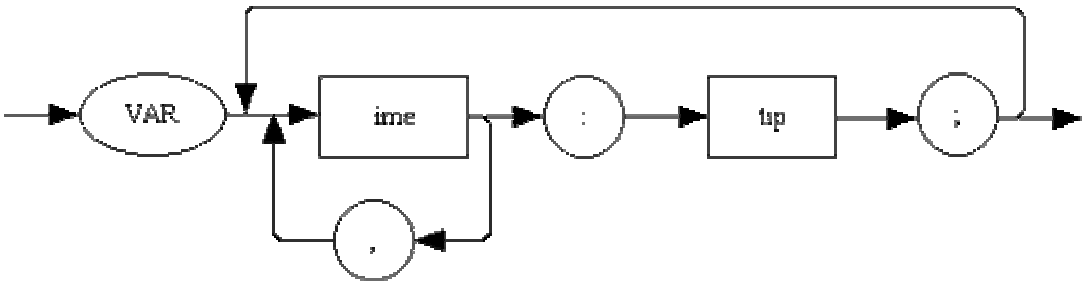
Oznaka može biti cijeli broj od 0 do 9999 i mora biti jedinstvena u jednom bloku. Sintaksa deklaracije oznaka je prikazana na sljedećoj slici:



Primjer: LABEL 1, 55,100,9999;

3. Deklaracija varijabli.

Najavljuju se varijable koje će se koristiti u programu. Varijabla je objekt čija se vrijednost mijenja tijekom rada programa. U deklaraciji varijabli navodimo sve varijable i njihove tipove. Svakoj se varijabli pridruži dio memorijskog prostora, a veličina tog prostora je određena tipom varijable.
Sintaksa deklaracije varijabli je prikazana na sljedećoj slici:



Primjer: VAR i, j: integer;
znak: char;
a, b, c: real;

4. Naredbe.

Opisuju akcije koje će se izvršiti za vrijeme rada programa.
Naredbe možemo podijeliti na:

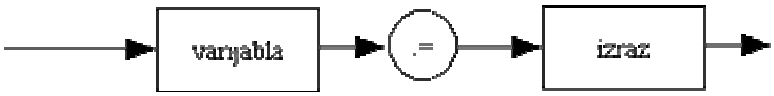
- jednostavne naredbe
- strukturalne naredbe.

Jednostavne naredbe.

1. Bezuvjetni skok – *GOTO naredba*. Prirodno se naredbe izvršavaju onim redoslijedom kojim su napisane. Međutim, naredba GOTO prekida prirodno izvršavanje naredbi te prenosi kontrolu na naredbu s navedenom oznakom (broj od 0 do 9999). Sve oznake moraju biti deklarirane na početku bloka.

Oznaka se piše ispred naredbe, a između oznake i naredbe mora biti dvotočka. Jedna oznaka u bloku može se pridružiti samo jednoj naredbi.

2. Naredba pridruživanja je jednostavna naredba koja varijabli pridružuje vrijednost izraza tj. u memorijsku lokaciju pridruženu varijabli upisuje vrijednost izraza. Znak " := " je znak pridruživanja.
   Sintaksa naredbe pridruživanja je prikazana na sljedećoj slici:

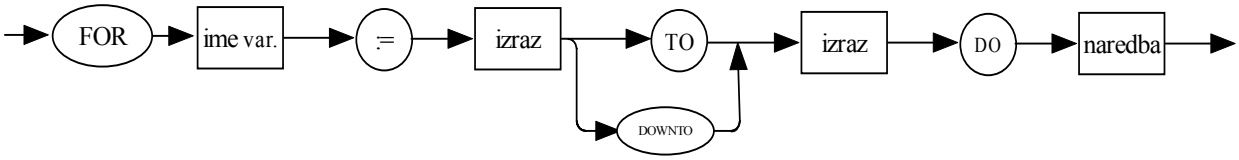


3. Prazna naredba nema nikakvu oznaku niti utječe na izvršavanje programa.

Strukturalne naredbe.

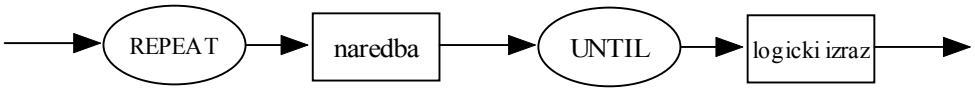
1. Naredbe za ponavljanje
   - *FOR naredba*

   Sintaktički dijagram ove naredbe je prikazan na sljedećoj slici:



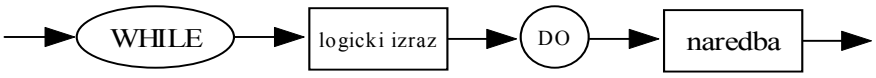
   - *REPEAT naredba*

   Blok naredbi se ponavlja dok se uvjet ne ispuni. Uvjet se provjerava na kraju. Minimalni broj ponavljanja je 1.
   Sintaktički dijagram ove naredbe je prikazan na sljedećoj slici:



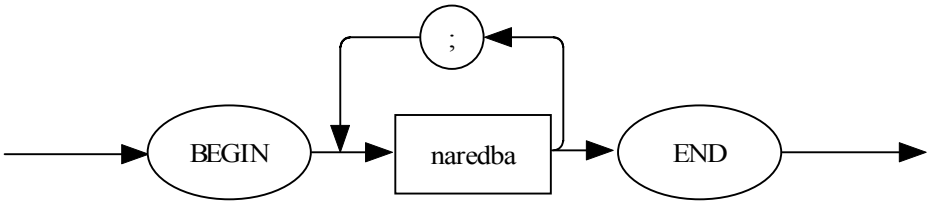
   - *WHILE naredba*

   Blok naredbi se ponavlja dok je uvjet ispunjen. Uvjet se provjerava na početku. Minimalni broj ponavljanja je 0.
   Sintaktički dijagram ove naredbe je prikazan na sljedećoj slici:



2. Složena naredba – to je niz naredbi koje su odvojene s " ; " i ograničene ključnim riječima BEGIN i END. Program u Pascal-u možemo shvatiti kao jednu složenu naredbu.

Sintaktički dijagram ove naredbe je prikazan na sljedećoj slici:



3. Uvjetna naredba

- *CASE naredba* omogućava grananje programa u više smjerova ovisno o vrijednosti selektorskog izraza . Vrijednost izraza može biti realnog tipa (char, Boolean, integer, pobrojani ili neki intervalni tip).

Sintaksa CASE naredbe:

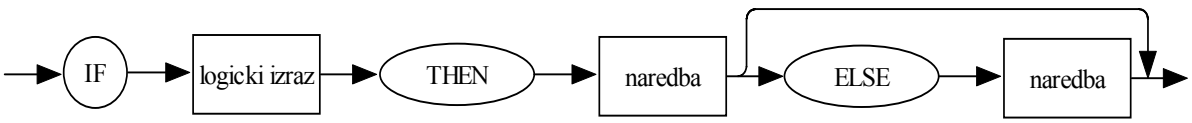
```
CASE izraz OF
        const1: blok naredbi;
        const2: blok naredbi;
        . . .
        constn: blok naredbi;
        ELSE blok naredbi
END;
```

Primjer:

```
write ('Unesi broj 1, 2 ili 3.'); readln(a);
CASE a OF
        1:writeln('Unijeli ste broj 1.');
        2: writeln('Unijeli ste broj 2.');
        3: writeln('Unijeli ste broj 3.')
                ELSE writeln('Pogrešan unos.');
END;
```

- *IF naredba.* Postoje dvije kombinacije ove naredbe:
  - IF… THEN
  - IF… THEN… ELSE

što je prikazano na slici:



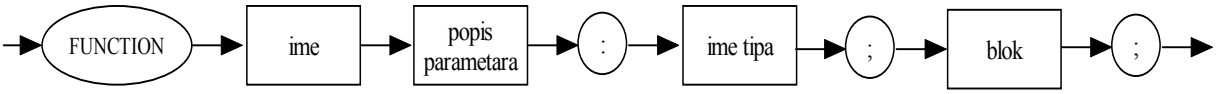
Primjer:

```
IF x mod 2 = 0 THEN writeln ('Broj je paran)
                ELSE writeln ('Broj je naparan.');
```

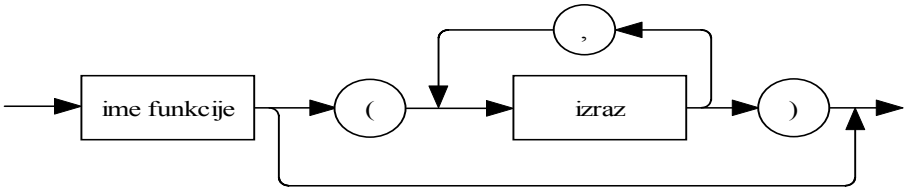
5. Funkcije.

Omogućavaju rastavljanje programa u jednostavnije i razumljivije logičke cjeline koje
pozivamo na izvršavanje njihovim imenom. Osim unaprijed definiranih standardnih funkcija korisnik može sam definirati vlastite funkcije. Funkcije su, u biti, potprogrami koji daju određenu izlaznu vrijednost koja mora biti pridružena imenu funkcije. Tip vrijednosti mora odgovarati tipu funkcije kojeg definiramo u zaglavlju tako da navedemo ime nekog prostog tipa.

Struktura funkcije prikazana je na sljedećoj slici:



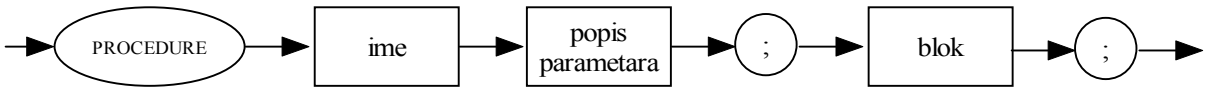
Dijagram pozivanja funkcije:



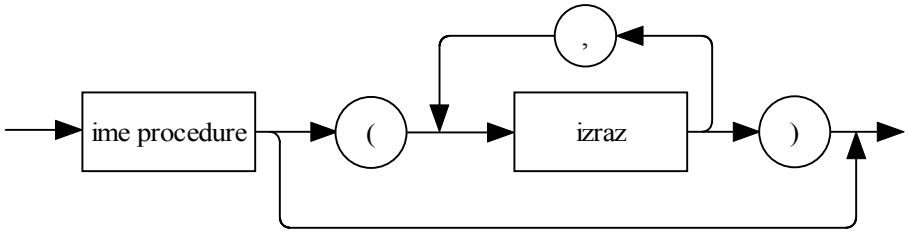
6. Procedure.

Procedure omogućavaju rastavljanje programa na jednostavnije i razumljivije logičke cjeline koje pozivamo na izvršavanje njihovim imenom. To su potprogrami koji mogu imati više ulaznih i više izlaznih vrijednosti pridruženih varijablama. Na izvršavanje ih pozivamo imenom nadopunjenim stvarnim parametrima (ako je procedura deklarirana s formalnim parametrima).

Struktura procedure je prikazana na sljedećoj slici:



Dijagram pozivanja procedure:



Procedure mogu imati različite oblike:

- Procedure s okolinom povezane preko formalnih parametara
- Procedure s okolinom povezane preko globalnih varijabli
- Procedure s okolinom povezane preko globalnih varijabli i formalnih parametara

Struktura procedure se sastoji od bloka i deklarativnog dijela.

Deklarativni dio.

Obuhvaća naziv procedure ili funkcije i popis parametara preko kojih je ista povezana s okolinom. Što se parametara tiče, razlikujemo formalne parametre i stvarne parametre. Formalne parametre dalje dijelimo na *varijabilne parametre* i *vrijednosne parametre*.

Blok.

Obuhvaća opis podataka i opis akcija.

Dijelovi bloka su:

- Izrazi. Oni određuju način izračunavanja vrijednosti. Razlikujemo *operande* i operatore. Operandi se izvršavaju prema sljedećem prioritetu:
  1. NOT
  2. AND * / DIV MOD
  3. OR + -
  4. IN = > < >= <= <>.

Operatori istog prioriteta se izvršavaju s lijeva na desno. Računanje vrijednosti funkcije ima viši prioritet od operatora. Izrazi u zagradama imaju najviši prioritet.

Važno je naglasiti da operatori moraju biti istog tipa. Tip vrijednosti izraza ovisi o tipu operanda.

- Naredbe (spomenuto prije).
- Varijable.
  - *Globalne varijable* – mogu se koristiti u svim blokovima u kojima ne postoji lokalna varijabla istog imena.
  - *Lokalne varijable* - svi objekti definirani i deklarirani u bloku procedure su lokalni za tu proceduru i ne mogu se upotrijebiti izvan tog bloka.

7.Tipovi podataka.

To su konačni skupovi vrijednosti za koje vrijede neka zajednička svojstva.

Vrste tipova podataka su:

- jednostavni tipovi podataka
- složeni tipovi podataka
- pokazivači (POINTERS).

U Pascal-u pored standardnih tipova možemo koristiti i vlastite tipove podataka.

Primjer:

```
TYPE  dan = 1..31;
      slova = 'A'..'Z';
      cijeli = 'integer';
```

Jednostavni tipovi podataka.

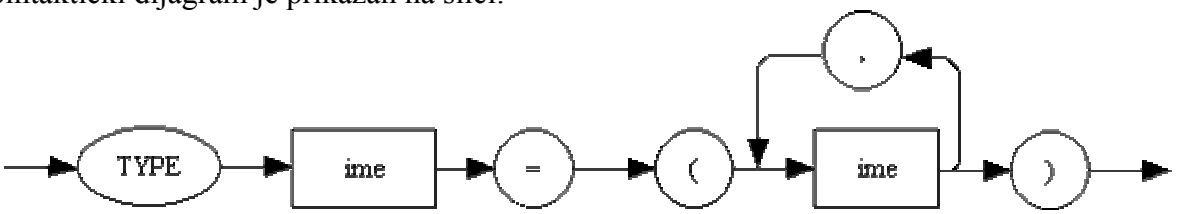
Dijele se na korisničke tipove podataka i standardne tipove podataka.

Korisnički tipovi podataka.

1. *Intervalni (SUBRANGE) tipovi podataka.* Zadajemo ga kao interval prethodno definiranog rednog tipa (char, Boolean, integer) određujući donju i gornju granicu sa dvije točke između. Donja granica ne može biti veća od gornje. Uređaj se prenosi iz pridruženog rednog tipa. Sintaktički dijagram definicije tipa je prikazan na slici:

2. *Pobrojani (ENUMERATED) tipovi podataka*. Tip zadajemo tako da u okruglim zagradama navedemo sve elemente tipa. Uređaj je definiran redoslijedom nabrajanja. Definirane su standardne funkcije:
   - succ(x) – sljedbenik elementa x,
   - pred(x) – prethodnik elementa x,
   - ord(x) – redni broj elementa x (redni broj prvog elementa je 0).

   Sintaktički dijagram je prikazan na slici:



Standardni tipovi podataka.

1. *Cjelobrojni (INTEGER) tipovi podataka.* Integer je standardni tip podatka i predstavlja podskup skupa cijelih brojeva. Negativni broj se predstavlja kao dvojni komplement odgovarajućeg pozitivnog broja.
   Na tipu integer definirane su sljedeće standardne operacije:
   - * - množenje,
   - DIV - cjelobrojno dijeljenje (npr. 7 DIV 2 = 3),
   - MOD – ostatak cjelobrojnog dijeljenja (npr. 7 MOD 2 = 1),
   - + - zbrajanje,
   - - - oduzimanje.

   Sljedeće standardne funkcije daju rezultat tipa integer:
   abs(i) – apsolutna vrijednost (ako je i tipa integer),
   sqr(i) – kvadrat (ako je i tipa integer),
   trunc(x) – cijeli dio realnog broja (npr. trunc(3.7) = 3),
   round(x) – najbliža cjelobrojna vrijednost (npr. round(3.7) = 4),
   succ(i) = i+1 – sljedbenik,
   pred(i) = i-1 – prethodnik.

2. *Logički (BOOLEAN) tipovi podataka.* Logički tip je standardni tip podatka sa dva elementa false i true, gdje su false (neistina) i true (istina) standardna imena.
   Nad logičkim tipom definirani su standardni operatori:
   - NOT – negacija,
   - AND – konjunkcija,
   - OR – disjunkcija.

   Upotrebljavajući relacijske operatore na ovom tipu, te svojstvo da je false < true, možemo definirati i:
   - <> - ekskluzivna disjunkcija,
   - <= - implikacija,
   - = - ekvivalencija.

3. *Realni (REAL) tipovi podataka*. Tip podataka real je standardni tip i predstavlja konačan podskup skupa realnih brojeva. Zbog toga se operacije vrše uglavnom s približnim vrijednostima, a rezultati su najčešće aproksimacije točnih vrijednosti. Tip real i tip integer su sva disjunktna skupa.
   Na tipu real su definirani standardni operatori:
   - * - množenje,
   - / - dijeljenje (operandi mogu biti tipa integer, ali je rezultat tipa real),
   - + - zbrajanje,
   - - - oduzimanje.

   Standardne funkcije:
   abs(x) i sqr(x) daju rezultat tipa real ako su argumenti tipa real.
   Sljedeće funkcije daju rezultat tipa real bez obzira da li su argumenti tipa real ili integer:
   sin(x) – sinus kuta u radijanima,
   cos(x) – kosinus kuta u radijanima,
   arctan(x) – arkus tangens,
   exp(x) – eksponencijalna funkcija (e na x),
   ln(x) – prirodni logaritam,
   sqrt(x) – drugi korijen.

4. *Znakovni (CHAR) tipovi podataka*. Char je standardni tip podatka i predstavlja linearno uređen konačan skup znakova (slova, znamenke, posebni znakovi). Svakom znaku pridružen je redni broj (njegov interni kod). Konstanta tipa char se zapisuje u jednostrukim navodnicima kao npr. 'A', 'z', '3'.
   Standardne funkcije:
   ord(c) – redni broj znaka c (ord('A') = 65),
   chr(n) – znak čiji je redni broj n (chr(65) = 'A').
   Funkcije ord i chr su međusobno inverzne.

Složeni tipovi podataka.

1. Niz (ARRAY).

Pod istim imenom objedinjuje čitav skup varijabli koje se međusobno razlikuju samo po različitim indeksima.

Niz je sastavljen od komponenti istog tipa. Tip komponente može biti bilo koji jednostavan ili složeni tip osim FILE. Pojedinu komponentu niza možemo dobiti preko imena i rednog broja (indeksa) u nizu. Najveća duljina niza (broj komponenata) mora biti određen na početku programa i ne može se naknadno mijenjati. Ovisno o veličini i tipu niza određuje se potreban memorijski prostor. Komponente označavamo imenom niza dodajući u uglatim zagradama indeksni izraz: x[1], x[2], x[3],…

Vrste nizova:

- *jednodimenzionalni niz*

Sintaksa:

```
TYPE niz = ARRAY [1..n] of real;
```

Deklaracija varijabli:

```
VAR a: niz;
```

- *višedimenzionalni niz*

Dvodimenzionalni niz (matrica):

Sintaksa:

```
TYPE matrica = ARRAY [1..m,1..n] of real;
```

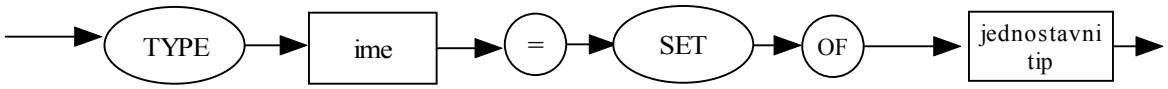
Deklaracija varijabli:

```
VAR a: matrica;
```

2. Skup (SET).

Zamjenjuje složene logičke izraze, jer su operacije sa skupom puno brže. Skup zadajemo tako da navedemo elemente odvojene zarezom u uglatim zagradama. Skup se u memoriji predstavlja nizom logičkih vrijednosti čija i-ta komponenta određuje prisutnost ili odsutnost nekog elementa u skupu.

Definicija tipa je prikazana na sljedećoj slici:



3. STRING.

Obično se upotrebljava za podatke tekstualnog tipa, te kroz niz funkcija i procedura omogućava mnogobrojne operacije nad tim podacima. Duljina stringa se može mijenjati za vrijeme izvođenja programa, a ne može biti veća od 255 znakova. Može se definirati string manje duljine tako da u uglatim zagradama navedemo najveću duljinu stringa npr.

```
VAR ime: string[20];
```

Sljedeće funkcije/ procedure omogućavaju rad sa stringovima:

- *concat* – FUNCTION concat (s1,.., sk: string): string;
  Concat je, dakle, funkcija koja spaja stringove s1,.., sk u jedan string.
- *copy* - FUNCTION copy (s: string; m, b: integer): string;

Copy je, dakle, funkcija koja daje dio stringa s počevši od m-tog znaka duljine b znakova.

- *delete* – PROCEDURE delete (VAR s: string; m, b: integer)
  Delete je, dakle, procedura koja unutar stringa s briše b znakova počevši od m-tog znaka.
- *insert* - PROCEDURE insert (s1: string; VAR s:string; m: integer)
  Insert je, dakle, procedura koja unutar stringa s dodaje string s1 počevši od m-tog znaka.
- *length* - FUNCTION length (s: string): integer;
  Length je, dakle, funkcija koja daje trenutnu (dinamičku) duljinu stringa s.
- *pos* - FUNCTION pos (s1, s: string): integer;
  Pos je, dakle, funkcija koja daje mjesto (indeks s prvog znaka) prve pojave podskupa s1 unutar stringa s.
- *str* - PROCEDURE str (n: integer; VAR s:string)
  Str je, dakle, procedura koja pretvara cijeli broj ili proširени cijeli broj u string s.

4. Zapis (RECORD).

Objedinjuje skup podataka koji u jednom zapisu mogu biti različitih tipova. Zapis je sastavljen od određenog broja komponenti (polja) koje mogu biti proizvoljnog tipa (osim FILE).

Primjer:
```
TYPE
    datum = RECORD
              dan: 1..31;
              mjesec: 1..12;
              godina: integer;
            END;
  var a,b: datum;
```
Zapis može biti komponenta i nekog drugog složenog tipa (ARRAY, RECORD, FILE). Pojedine komponente označavamo tako da navedemo ime varijable tipa RECORD, te ime polja (komponente) s točkom između (npr. a.dan := 25; b.mjesec := 7;).

5. Datoteka (FILE).

Objedinjuje skup podataka koji se pohranjuju u pomoćnu memoriju. Poredak komponenata u datoteci definiran je redoslijedom unošenja. Broj komponenti određuje duljinu datoteke i ona nije unaprijed određena. Dozvoljena je i prazna datoteka.

Deklaracija varijabli:

VAR d: datoteka;

Svaka datoteka ima oznaku za kraj: eof (end of file).

Vrste datoteka:

- *Direktna datoteka.* Vrijednosti u direktnoj datoteci moguće je direktno pročitati. Naime, direktan pristup je metoda kojom se može pročitati pojedinačni zapis iz datoteke (s diska) bez da se pristupa bilo kojem drugom zapisu.

- *Sekvencijalna datoteka.* Čitanje sekvencijalne datoteke moguće je samo onim redom kojim su vrijednosti bile pohranjene u pomoćnu memoriju.
- *Indeksno- sekvencijalna datoteka.* Ima brži pristup od sekvencijalne i posjeduje zapis ključeva i zapis podataka, pa pri čitanju imamo dvije akcije. Najprije čitamo zapis ključeva, pa onda traženi zapis podataka.

Operacije nad datotekama:

- *Inicijalizacija datoteke*
  - ASSIGN – funkcija pomoću koje definiramo i dajemo ime datoteci
- *Otvaranje datoteke*
  - APPEND – procedura koja omogućava otvaranje datoteke za upisivanje, tj. dodavanje podataka, ali ne i brisanje podataka.
  - RESET - procedura koja omogućava otvaranje datoteke za čitanje podataka.
  - REWRITE – procedura koja omogućava otvaranje datoteke za upisivanje sadržaja. U slučaju da datoteka nije postojala, sad se stvara, a ako je postojala briše se stari sadržaj i dodaju se novi podaci.
- *Akcija nad datotekama* – moguće je čitanje iz datoteke (READ) i pisanje u datoteku (WRITE).
- *Zatvaranje datoteke* – datoteka se zatvara sa CLOSE.

Pokazivači (POINTERS).

Omogućavaju pozivanje komponenata dinamičkih struktura koje nemaju eksplicitna imena, te uspostavljanje relacija između podataka.

Dinamičke varijable se po potrebi stvaraju i brišu za vrijeme izvođenja programa. One se ne označavaju određenim imenom i ne nalaze se u deklaraciji varijabli. Vrijednost određene dinamičke varijable možemo dobiti preko adrese memorijske lokacije koju sadrži pridružena pokazivačka varijabla.

Definicija tipa:

```
TYPE pokaz = ^T;
```

Tip pokaz je definiran kao skup vrijednosti koje pokazuju varijable tipa . Uvijek sadrži NIL koja ne pokazuje ni jednu dinamičku varijablu.

Deklaracija varijable:

```
VAR p: pokaz;
```

Deklaracijom pokazivačke varijable p tipa pokaz, određuje se memorijska lokacija samo za p. Vrijednost varijable p u početku nije definirana.

Primjeri programa u Pascal-u.

Primjer 1.

```
PROGRAM  tablica (output);
(*ispisuje tablicu faktorijela*)
CONST n=7;
```

```
VAR  i, fakt: integer;
        BEGIN
                fakt := 1;
                i := 1;
                REPEAT
                        fakt := fakt * i;
                        writeln (i, '!=', fakt);
                        i := i + 1
                UNTIL i > n
        END.
```

Primjer 2.

```
PROGRAM kvadrat (input, output);
(*čita stranicu kvadrata i ispisuje opseg i površinu*)
VAR stranica, opseg, površina: real;
BEGIN
        write ('Upišite stranicu kvadrata: ');
        read (stranica);
        opseg := 4 * stranica;
        površina := sqr (stranica);
        writeln ('Stranica kvadrata = ', stranica);
        writeln ('Opseg kvadrata = ', opseg);
        writeln ('površina kvadrata = ', površina);
END.
```

Primjer 3.

```
PROGRAM najmanji (input, output);
(*najmanji od tri učitana broja*)
VAR i, j, k, min: integer;
BEGIN
        writeln ('Upišite tri cijela broja: ');
        read (i, j, k);
        min := i;
        IF j < min THEN min := j;
        IF j < min THEN min := j;
        writeln ('Najmanji je ', min);
END.
```

Primjer 4.

```
PROGRAM najveći (input, output);
(*najvećii od tri učitana broja*)
VAR x, y, z, max: integer;
BEGIN
```

```
        writeln ('Upišite tri cijela broja: ');
        read (x, y, z);
        writeln (x,', ', y', ', z);
        IF x > y THEN
                IF x > z THEN max := x
                        ELSE max := z
        ELSE
                IF y> z THEN max := y
                        ELSE max := z
        writeln ('Najveći je ', min);
END.
```