

DATOTEKE

- Predstavitev

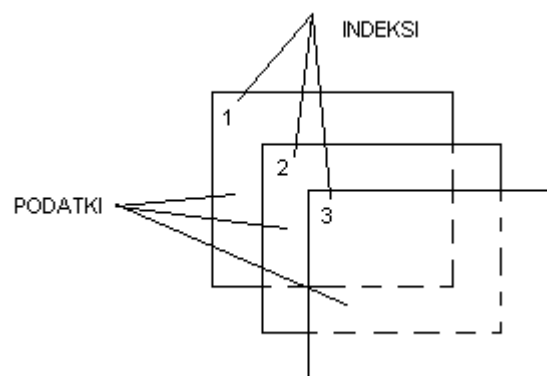
Datoteka

Record – zapis v datoteki!

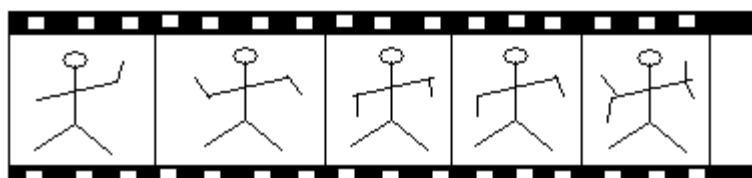
Podatki se hranijo na zunanjem pomnilniškem mediju (HD, FD,...)

Vrsta datotek:

- poznamo datoteke z direktnim dostopom, to so tekstovne datoteke, preberejo se od začetka do konca.
- Indeksne datoteke – teh pascal ne podpira (te so domene podatkovnih baz)



- sekvenčne datoteke - predstavljamo si jih kot fotografski film



1. sekvenca

2. sekvenca

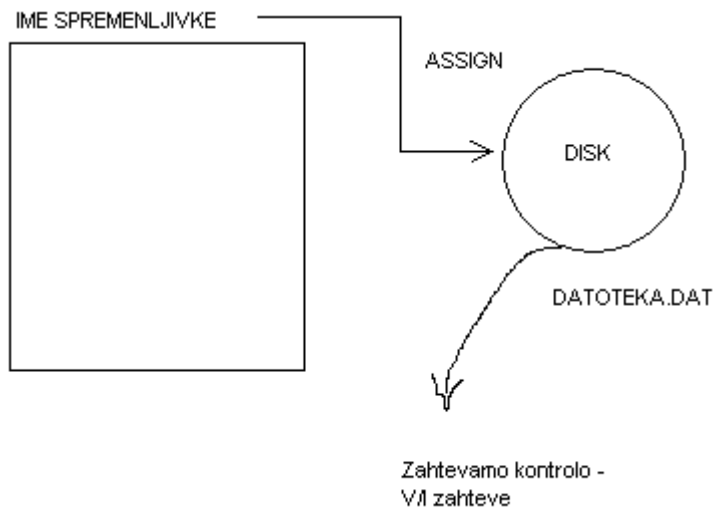
3. sekvenca

4. sekvenca

5. sekvenca

Vsako datoteko v pascalu, je potrebno najprej povezati z fizičnim imenim na disku. Logični spremenljivki v pascalu priredimo ime na disku.

Logični spremenljivki priredimo fizično ime na disku s pomočjo procedure ASSIGN!



Type

```

ime = record
  ime: string[20];
  naslov:string[100];
end;

```

ZBIRKA

Ime
Ime
Ime
Ime
Ime
Ime

Var

```

zbirka: file of ime;
pod: ime;
begin
  assign(zbirka,'C:\BAZA.DAT');
  {$I-}
  reset(zbirka);
  IF ioresult <> 0 then
    rewrite(zbirka);
  {$I+}
  pod.ime := 'Andrej';
  pod.naslov := 'cankarjeva 10';
  seek(zbirka,filesize(zbirka));
  write(zbirka,pod);
  close(zbirka);
end.

```

UKAZI ZA DELO Z DATOTEKAMI

ASSIGN – poveže nam logično spremenljivko v pascalu z fizično datoteko na disku

REWRITE – datoteko, če obstaja zbriše in na novo kreira

RESET – obstoječo datoteko odpre za branje in pisanje (sekvenčne datoteke) oz. Samo za branje (tekstovne datoteke)

APPEND – obstoječo datoteko odpre za dodajanje na koncu (samo tekstovne datoteke)

IOResult – vrača celoštevilčno vrednost, ali je ukaz za delo z V/I napravami uspel ali ne (0 pomeni uspešno opravljeno delo)

CLOSE – zapremo datoteko

FILESIZE – vrne nam velikost datoteke. Pri sekvenčnih datotekah nam vrne koliko sekvenc je dolga datoteka, pri tekstovnih pa koliko znakov zavzema datoteka.

SEEK – pomikamo se po datoteki

WRITE – vnos v datoteko

Writeln – vnos v datoteko in skok v naslednjo vrstico (samo tekstovne datoteke)

READ – branje iz datoteke

Readln – branje cele vrstice iz datoteke (samo tekstovne datoteke)

EOF – pregledamo, če smo na koncu datoteke

Paziti moramo, ker pri branju sekvenčnih datotek se z vsakim branjem pomikamo po datoteki proti koncu. Po vsakem branju je zaželeno pregledati, ali smo na koncu ali ne. Namreč branje preko konca povzroči napako!

Dobro je tudi vedeti, ko sekvenčno datoteko odpremo, je kazalec postavljen na začetek. Pri vnosu podatka, če se ne premaknemo na konec, se podatki prepisujejo (situacija: imamo 10 podatkov v datoteki. Datoteko odpremo za branje in pisanje. Kazalec datoteke je postavljen na začetek.

Izvedemo proceduro za pisanje in prepiše se nam prvi podatek – podatek izgubimo. Če se pa najprej postavimo na konec datoteke, se pa nam ustvari nov zapis z podatki.

Kriterij za izbor najustreznejše organizacije datotek:

- poraba pomnilnega prostora
- čas dostopa do poljubnega zapisa v datoteki
- čas potreben za dodajanje, brisanje zapisa v datoteko
- čas dostopa do logično naslednjega zapisa

Programski jezik pascal pozna datoteke z direktnim dostopom (tekstovne datoteke) in sekvenčne datoteke (datoteke z lastnim podatkovnim tipom).

Razlika se opazi pri deklaraciji, pa tudi pri uporabi ene in druge strukture pride do razlik.

Deklaracija datoteke z direktnim dostopom (tekstovne).

Var

a:TEXT;

s:string;

c:char;

spremenljivka a je tipa TEXT. Oba tipa datotek je potrebno najprej povezati z fizičnim imenom na disku (assign ukaz). Razlike pridejo pri odpiranju datoteke. Pri tekstovnih reset odpre samo za branje in ni mogoče pisati. Kazalec se postavi na začetek, rewrite datoteko zbriše in ni možno branje, ampak le pisanje v datoteko, medtem ko append je mogoče uporabiti samo na tekstovnih datotekah, le ta pa datoteko odpre za pisanje in postavi kazalec na konec.

Pisanje in branje poteka enak v tekstovnih, kot tudi v sekvenčnih datotekah.

Writeln(a,'tekst ki gre v datoteko');

read(a,c);

readln(a,s);

Pri tekstovnih datotekah lahko uporabimo, z razliko sekvenčnih datotek, tudi readln in writeln. Prvi nam prevere celotno vrstico v spremenljivko tipa string, drugi pa nam vnese niz v datoteko. Pri tem oba skočita v naslednjo vrstico.

Deklaracija datoteke z sekvenčnim dostopom.

```

Type
oseba=record
  ime:string[20];
  priimek:string[20];
  telefon:string[20];
end;
var
  a:oseba;
  f:file of oseba;

```

Pri uporabi procedure rewrite, se nam datoteka popolnoma zbriše in kreira na novo (tudi če ne obstaja se kreira). Po datoteki lahko pišemo in beremo. Ravno tako pri uporabi reset, le v tem primeru, mora datoteka že obstajati – lahko pišemo in beremo. Append ukaz pri delu z sekvenčnimi datotekami ne obstaja.

UKAZA {\$I-} in {\$I+}

Ukaz {\$I-} nam izključi kontrolo V/I naprav v operacijskem sistemu. Namreč operacijski sistem sam vedno kontrolira ali je mogoče pisati ali ni mogoče pisati (zaslon, tipkovnico, datoteko). Ker navadno ne vemo, ali datoteka obstaja ali ne, nam v primeru, če datoteka ne obstaja in mi želimo pisati v datoteko operacijski sistem vrne napako (napaka v primeru, da se program ustavi in izpiše napako). Take napake so nezaželjene in programer mora v tem primeru izključiti kontrolo operacijskega sistema in sam izvesti kontrolo. Torej ukaz {\$I-} prekine kontrolo operacijskega sistema in prepusti kontrolo programerju. Po izvedbi vsakega ukaza, ki dela z V/I napravami je potrebno preveriti ali je ukaz uspešen ali neuspešen. Za to opravilo nam pascal ponuja funkcijo IOResult, ki vrne 0, ko je bilo vse v redu obdelano ali bilokatero drugo število ob napaki. Vsako število pomeni neko drugo napako (mogoče napake: datoteka ne obstaja, nimamo dostopa do datoteke, datoteke ne moremo brisati, datoteke ne moremo kreirati,...).

Splošna praksa je, da datoteko najprej poizkušamo odpreti (ukaz reset), kot da datoteka obstaja.

Nato preverimo, če je vse v redu (torej datoteka obstaja in je pripravljena za branje/pisanje/dodajanje). V primeru napak pa datoteko poizkušamo kreirati (ukaz rewrite) (po vsej verjetnosti datoteka ne obstaja).

Ko datoteko ustvarimo za obdelavo, lahko kontrolo zopet prepustimo operacijskemu sistemu in z ukazom {\$I+} le-to naredimo.