Plik

- Nazwa
- Ciągły obszar w pamięci nieulotnej

Postać

- Binarna
- Tekstowa

Postępowanie

- Deklaracja zmiennej typu plikowego
- Powiązanie nazwy pliku ze zmienną plikową (assign)
- Otworzenie pliku do zapisu lub odczytu
- Zapis / odczyt
- Zamknięcie pliku

- Możliwość ponownego wykorzystania zmiennej plikowej

Pliki elementowe

- Przechowują dane w postaci binarnej
- Zazwyczaj nieczytelne dla człowieka
- Oszczędność miejsca (w porównaniu z reprezentacją tekstową)
- Wszystkie elementy przechowywane w plikach elementowych muszą być tego samego typu
- Możliwy jest dostęp swobodny

- **type**
  - typPlikowy = **file of** typElementow;

```
type
    Osoba = record
            Imie: string[15];
            Nazwisko: string[25];
            Adres  : string[35];
            RokUr     : integer;
    end;

zmPlikowaOsoba= file of Osoba;
zmPlikowaInteger = file of Integer;
```

Powiązanie

- **assign**(zmiennaPlikowa, nazwaPliku)

- powiązuje zmienną plikową z fizycznym plikiem znajdującym się na dysku

```
var
    zmiennaPlikowaInteger: file of Integer;
begin
   assign(zmiennaPlikowaInteger, 'plikliczb.dat');
   ...
end.
```

Tryby pracy z plikiem(otwieranie pliku):

- **reset(zmienna_plikowa)**
  - otwiera plik do odczytu (plik musi istnieć)
  - dla plików elementowych zezwala na zapis
- **rewrite(zmienna_plikowa)**
  - otwiera plik do pisania,
  - usuwa całą zawartość i zaczyna zapisywanie od początku pliku,
  - gdy plik nie istnieje - tworzy
  - dla plików elementowych zezwala na odczyt
- Procedury ustawiają tzw. wskaźnik plikowy na początku pliku

```
var
    numberFile : file of Integer;
begin
   assign(numberFile,'plikliczb.dat');
   rewrite(numberFile);
   ...
   reset(numberFile);
   ...
end.
```

Zapis i odczyt danych z pliku

- **read**(zmienna_plikowa, lista_elementów)
  - odczytuje elementy z pliku
- **write**(zmienna_plikowa, lista_elementów)
  - zapisuje elementy do pliku

- lista_elementów – oddzielone przecinkiem zmienne określonego typu

```
var
    numberFile : file of Integer;
begin
    assign(numberFile,'plikliczb.dat');
    rewrite(numberFile);
    write(numberFile,10);        //BŁĄD - elementy przekazywane są przez nazwę
    ...
    reset(numberFile);
    ...
end.
```

```
var
    numberFile : file of integer;
    i:integer;
begin
    assign(numberFile,’plikliczb.dat’);
    rewrite(numberFile);
    i:=10;
    write(numberFile,i);
    ...
    reset(numberFile);
    ...
end.
```

Uwagi

- Nie zamknięcie pliku może spowodować utratę danych
- Buforowanie operacji dyskowych:
  - Redukcja liczby fizycznych odczytów i zapisów na dysku
  - Zawartość bufora wysyłana jest na dysk po jego zapełnieniu (lub w chwili zamknięcia pliku)
- Poprawne zakończenie programu powoduje
  - automatyczne zamknięcie otwartych plików,
  - nie opróżnia buforów

Zamykanie plików

- close(zmienna_plikowa)

```
var
    numberFile : file of integer;
    i,x:integer;
begin
    assign(numberFile,'plikliczb.dat);
    rewrite(numberFile);
    i:=11;
    write(numberFile,i);
    reset(numberFile);
    read(numberFile,i);
    writeln(i);
    close(numberFile);
end.
```

- **eof**(zmienna_plikowa) – wykrywa moment napotkania końca pliku.
  - Zwraca TRUE jeżeli koniec pliku.

```
var
    numberFile : file of integer;
    i:integer;
begin
    assign(numberFile,'plikliczb.dat');
    rewrite(numberFile);
    for i:=3 to 10 do
        write(numberFile,i);
    reset(numberFile);
    while not (eof(numberFile)) do begin
        read(numberFile,i);
        writeln(i);
    end;
    close(numberFile);
end.
```

Wskaźnik plikowy

- Wskazuje na bieżący element w pliku.
- Numeracja elementów zaczyna się od zera.
- Każda operacja odczytu lub zapisu powoduje przesunięcie wskaźnika o wartość równą liczbie odczytanych lub zapisanych elementów.
- Dla plików elementowych możliwe jest dowolne przestawianie wskaźnika plikowego

- **seek**(zmienna_plikowa, numer_elementu)
- ustawia jako bieżący element (element na który wskazuje wskaźnik plikowy) element o podanym numerze numer_elementu

```
var
    numberFile : file of integer; i:integer;
begin
    assign(numberFile,'plikliczb.dat');  rewrite(numberFile);
    for i:=3 to 10 do
            write(numberFile,i);
    seek(numberFile,0);
    read(numberFile,i);
    writeln(i); {3}
    close(numberFile);
end.
```

- **filePos**(zmienna_plikowa)
- zwraca numer bieżącego elementu – wartość wskaźnika plikowego

```
var
    numberFile : file of integer; i:integer;
begin
    assign(numberFile,'plikliczb.dat');  rewrite(numberFile);
    for i:=3 to 10 do
            write(numberFile,i);
    writeln(filePos(numberFile));    {8}
    seek(numberFile,0);
    writeln(filePos(numberFile));       {0}
    close(numberFile);
end.
```

- **fileSize**(zmienna_plikowa)
- zwracająca rozmiar pliku – liczbę elementów

```
var
    numberFile : file of integer; i:integer;
begin
    assign(numberFile,'plikliczb.dat');  rewrite(numberFile);
    for i:=3 to 10 do
            write(numberFile,i);
    writeln(fileSize(numberFile));    {8}
    close(numberFile);
end.
```

- Ustawienie wskaźnika plikowego za ostatni element
  - przygotowanie pliku do dopisywania danych

```
var
   numberFile : file of integer;
begin
   ...
   seek(numberFile, FileSize(numberfile));
   ...
end.
```

- Zwrócenie losowego elementu pliku

```
var
   numberFile : file of integer;
begin
    …
    seek(numberFile,random(FileSize(numberfile)));
    …
end.
```

Pliki tekstowe

- Przechowują dane w postaci wierszy tekstu zakończonych znakami końca wiersza.
- Treść pliku tekstowego daje się łatwo odczytać i zinterpretować.
- Reprezentacja tekstowa danych jest mniej zwarta i oszczędna niż binarna
- Mogą być użyte do przechowywania mieszanych typów danych
- Umożliwiają formatowanie zapisu
- Pliki o dostępie sekwencyjnym

```
var
   textFile : text;
//tego typu jest standardowe we. i wy.
begin
   assign(textFile,’plik.txt’);
   ...
   close(textFile);
end.
```

- **reset**(zmienna_plikowa)
- otwiera plik do czytania (plik musi istnieć)

- **rewrite**(zmienna_plikowa)
- otwiera plik do pisania, usuwa całą zawartość i zaczyna zapisywanie od początku pliku, jeśli plik nie istnieje - tworzy go

- **append**(zmienna_plikowa)
- umożliwia dopisywanie nowych danych do pliku (plik musi istnieć - tylko dla plików TEKSTOWYCH)

```
var
   textFile : text;
begin
     assign(textFile,’plik.txt’);
     rewrite(textFile);
     for i:=0 to 5 do
        write(textFile,i);
     read(textFile,i); {BŁĄD}
     close(textFile);
end.
```

```
var
    textFile : text;
    i:integer;
begin
      assign(textFile,'plik.txt');
      rewrite(textFile);
      for i:=0 to 5 do
          write(textFile,i);
      reset(textFile);
      read(textFile,i); {12345}
      close(textFile);
end.
```

```
var
    textFile : text; i:integer;
begin
      assign(textFile,'plik.txt');
      rewrite(textFile);
      for i:=0 to 5 do
          write(textFile,i);
      reset(textFile);
      read(textFile,i);
      append(textFile);
      write(textFile,i);
      close(textFile);
end.
```

- **read**(zmienna_plikowa, lista_elementów)
- odczytuje elementy z pliku

- **readln**(zmienna_plikowa, lista_elementów)
- odczytuje elementy + znak końca wiersza

- **write**(zmienna_plikowa, lista_elementów)
- zapisuje elementy do pliku

- **writeln**(zmienna_plikowa, lista_elementów)
- zapisuje elementy do pliku + znak końca wiersza

```
var
    textFile : text;
    i:integer;
begin
     assign(textFile,’plik.txt’);
     rewrite(textFile);
     for i:=0 to 5 do
         write(textFile,i);
     reset(textFile);
     read(textFile,i); {12345}
     close(textFile);
end.
```

```
var
    textFile : text;
    i:integer; s:string;
begin
    assign(textFile,'plik.txt');
    rewrite(textFile);
    for i:=0 to 5 do
        write(textFile,i);
    reset(textFile);
    read(textFile,s);        {012345}
    close(textFile);
end.
```

```
var
    textFile : text; i:integer; s:string;
begin
      assign(textFile,’plik.txt’);
      rewrite(textFile);
      i:=5;
      write(textFile,10);
      write(textFile, 10*5*i);
      reset(textFile);
      read(textFile,s);         {10250}
      close(textFile);
end.
```

- **seekEof**(zmienna_plikowa)
- zwracająca TRUE jeżeli w pliku nie ma więcej tekstu za wyjątkiem odstępów i pustych wierszy.

- **seekEoln**(zmienna_plikowa)
- zwracająca TRUE jeżeli w wierszu nie ma więcej tekstu za wyjątkiem odstępów

Pliki binarne

```
var
bFile : file;

begin
assign(bFile,’bplik.txt’);
…
close(bFile);
end.
```

Pliki blokowe

- **reset**(zmienna_plikowa, rozmiar_bloku)
- otwiera plik do czytania (plik musi istnieć)

- **rewrite**(zmienna_plikowa, rozmiar_bloku)
- otwiera plik do pisania, usuwa całązawartość i zaczyna zapisywanie od początku pliku, jeśli plik nie istnieje - tworzy go

- rozmiar bloku – liczba bajtów
  - odczytywana z pliku, zapisywana do pliku; liczba bajtów o którą przesuwa sięwskaźnik plikowy.

Pliki blokowe

```
var
    bFile : file;
    i:integer;
begin
     assign(bFile,’bplik.txt’);
     rewrite(bFile,1);
     ...
     reset(bFile,1);
     ...
     close(bFile);
end.
```

- blockread(zmienna_plikowa, bufor,
  liczba_blokow,liczba_przeczytana);

- blockwrite(zmienna_plikowa, bufor,
  liczba_blokow,liczba_zapisana);

  - Liczba_blokow – liczba bloków do przeczytania/zapisania
  - Liczba_przeczytana – liczba bloków, które zostały przeczytane/zapisane

```
var
    bFile, bFile2 : file;
    nr,nw:integer;
    bufor:array [1..1028] of char;
begin
    assign(bFile,'p.txt');assign(bFile2,'p2.txt');
    rewrite(bFile,1); reset(bFile2,1);
    repeat
        blockRead(bFile2,bufor,sizeOf(bufor),nr);
        blockWrite(bFile, bufor, nr, nw);
    until (nr = 0) or (nr <> nw);
    close(bFile); close(bFile2);
end.
```