OPERACJE NA PLIKACH

Podstawowe pojęcia:

- **plik fizyczny**,
  → zbiór informacji w pamięci zewnętrznej wykorzystywany do trwałego przechowywania danych lub jako przedłużenie pamięci operacyjnej w przypadku przetwarzania struktur danych o dużych rozmiarach.

- **plik dyskowy**
  → plik fizyczny przechowywany na nośniku o własności trwałego zapamiętywania zapisanych danych: dyskietka, dysk stały, taśma, płyta CD, karta magnetyczna, ...

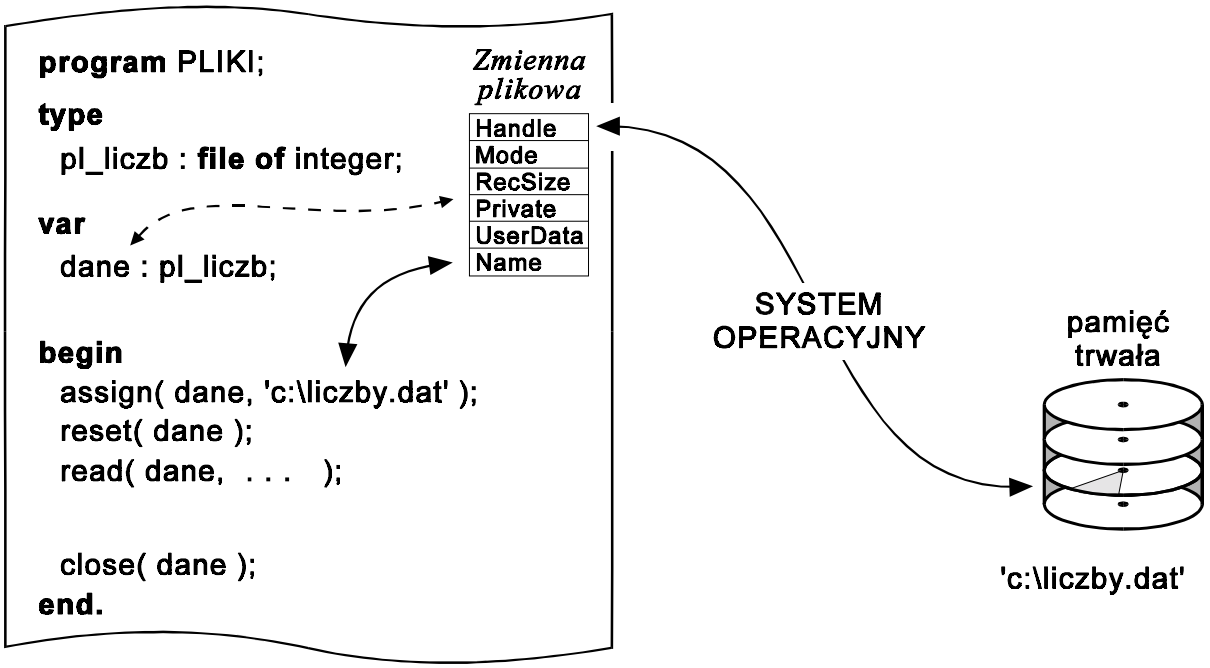
  → identyfikowany poprzez symbol napędu, ścieżkę dostępu do pliku (lista podkatalogów) oraz nazwę pliku:

  'symbol_napędu:\nazwa_katalogu\ ... \nazwa_katalogu\nazwa_pliku'

- **urządzenia wejścia/wyjścia** (też są traktowane jako pliki fizyczne)
  → klawiatura, monitor ekranowy, drukarka, porty szeregowe. Urządzenie identyfikowane poprzez łańcuch znakowy zawierający jego nazwę:

'CON'	–	konsola
'LPT1'	–	drukarka nr 1
'LPT2'	–	drukarka nr 2
'PRN'	–	domyślna drukarka
'COM1'	–	port szeregowy nr 1
'COM2'	–	port szeregowy nr 2
'AUX'	–	domyślny port szeregowy
'NUL'	–	urządzenie puste

- **zmienna plikowa**,
  → model logiczny służący do reprezentacji pliku fizycznego,
  → zmienna wykorzystywany do komunikacji pomiędzy programem a urządzeniem przechowującym plik fizyczny.



Reprezentacja zmiennych plikowych w pamięci:

```
type
  FileRec =    record
                 Handle    : Word;                 {uchwyt pliku}
                 Mode      : Word;                 {tryb otwarcia}
                 RecSize   : Word;                 {rozmiar elementu}
                 Private   : array[1..26]of Byte;  {pole zarezerwowane}
                 UserData  : array[1..16]of Byte;  {dowolne dane}
                 Name      : array[0..79]of Char;  {nazwa pliku}
               end;
```

Deklaracja typu i zmiennej plikowej dla plików elementowych

```
TYPE
  TypPlikowy  =  FILE OF  Element ;
VAR
  ZmiennaPlikowa    :  TypPlikowy ;
  ZmiennaPlikowa_2  :  FILE OF  Element_2;
```

Typ *Element* elementu pliku może być typem: liczbowym, znakowym, łańcuchowym, strukturalnym typem tablicowym lub typem rekordowym.
Nie może być typem plikowym !

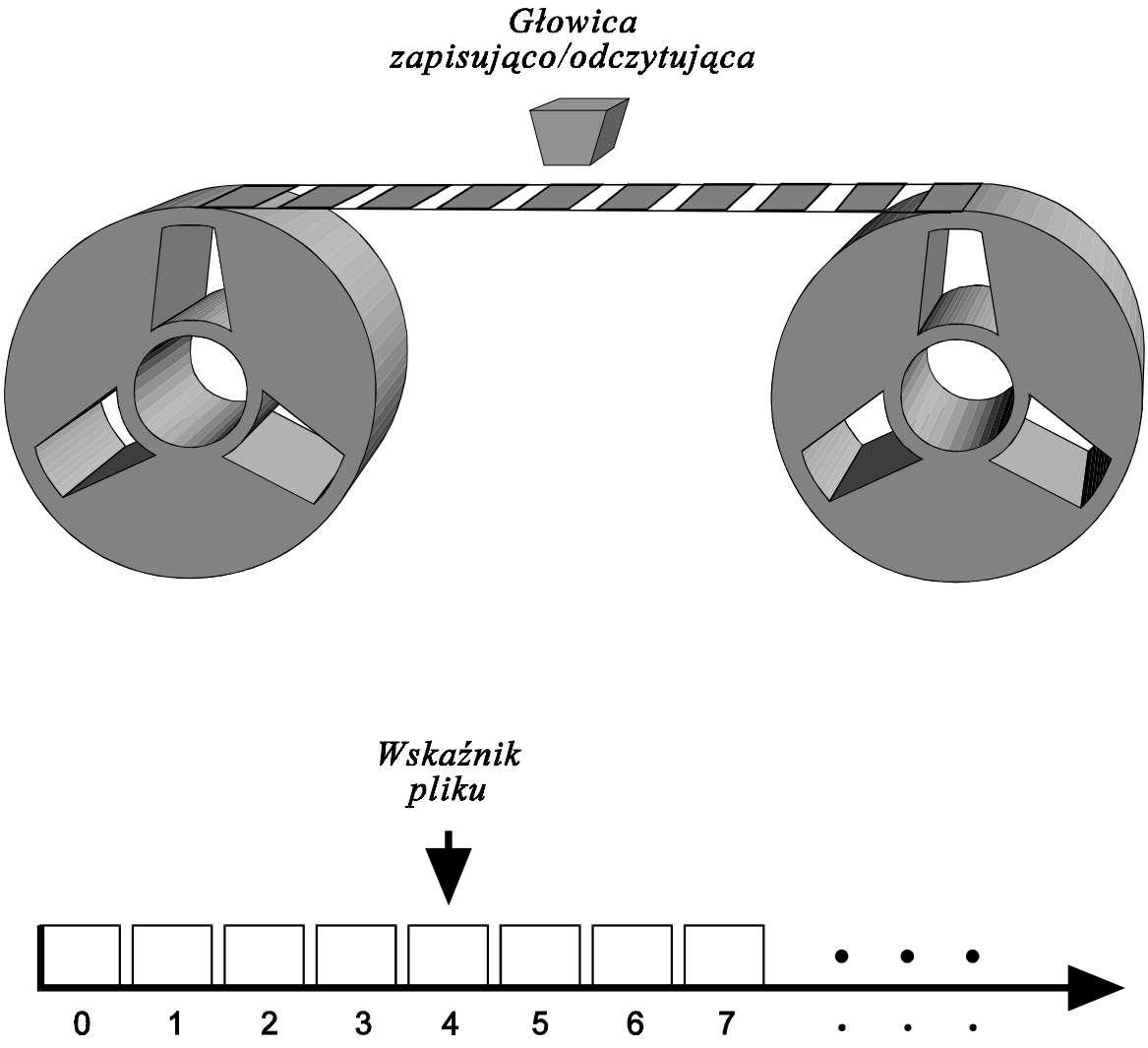
Wewnętrzna struktura pliku fizycznego

→ ciąg elementów zapisanych w postaci binarnej

→ w przypadku zmiennej plikowej zdefiniowanej (***file of*** *typ_elementów_pliku*) typ elementu i jego rozmiar określa definicja typu,

→ w przypadku zmiennej plikowej niezdefiniowanej (***file***) rozmiar elementu określany jest poprzez parametry odpowiednich procedur

→ pojęcie ***pozycji bieżącej*** pliku (***wskaźnika pliku***, *pozycji wskaźnika* pliku),



OGÓLNY SCHEMAT PRZETWARZANIA PLIKÓW

1. **Deklaracja zmiennych** plikowych oraz buforów programowych

```
type        element_pliku   =  definicja_typu;
          Tplik_przykladowy =  file of  element_pliku;
var        plik_przykladowy  :  Tplik_przykladowy;
```

---

2. **Skojarzenie** zmiennej plikowej z plikiem fizycznym

```
assign( plik_przykladowy, 'c:\wyklad\przyklad_1.dat' );
```

---

3. **Otwarcie pliku**

```
rewrite( plik_przykladowy );      {otwarcie do zapisu, utworzenie}
```

lub

```
reset( plik_przykladowy );      {otwarcie do odczytu, aktualizacji}
```

---

4. **Przetwarzanie** zawartości pliku fizycznego

```
for  i := 1  to  10  do
  begin
    read( dana );
    write( plik_przykladowy , dana );
  end;
```

lub

```
while not  eof( plik_przykladowy ) do
  begin
    read( plik_przykladowy , dana );
    writeln( dana );
  end;
```

---

5. **Zamknięcie pliku**

```
close( plik_przykladowy );
```

PROCEDURY I FUNKCJE DO OPERĄCJI NA PLIKACH

ASSIGN(zp,nazwa);	←	Skojarzenie zmiennej plikowej z plikiem fizycznym
REWRITE(zp);	←	Otwarcie do zapisu / utworzenie pliku
RESET(zp);	←	Przygotowanie pliku do odczytu
WRITE(zp,dane);	←	Zapis / dopisanie elementu do pliku
READ(zp,zmienna);	←	Odczyt elementu z pliku
EOF(zp):Boolean;	←	Sygnalizacja końca pliku
CLOSE(zp);	←	Zamykanie pliku

Dodatkowe funkcje dotyczące pliku elementowego

FILESIZE(zp):Longint;	←	liczba elementów pliku
FILEPOS(zp):Longint;	←	aktualne położenie wskaźnika pliku
SEEK(zp, pozycja);	←	ustawienie wskaźnika na zadanej pozycji
TRUNCATE(zp);	←	„Obcięcie” / skrócenie długosci pliku
ERASE(zp);	←	Kasowanie pliku
RENAME(zp,NowaNazwa);	←	Zmiana nazwy pliku

Ochrona przed błędami

Ustawianie opcji komplilacji → dyrektywy kompilatora {$I−}, {$I+}

Np.:

```
{$I−}
RESET(zp) ;
{$I+}
kod_bledu := IOResult ;
IF kod_bledu <> 0 THEN
        Error( kod_bledu ) ;
```

SKŁADOWANIE STRUKTURY DANYCH W PLIKU

```
CONST
   N = 100;
TYPE
   t_tablica = array[ 1 .. N ] of integer;
VAR
   tablica : t_tablica;
   i, il_elem : integer;
   plik_liczb : file of integer;
   plik_tablic : file of t_tablica;
```

Jednorazowe składowanie całej struktury:

```
Assign( plik_tablic, 'nazwa' );
Rewrite( plik_tablic );
Write( plik_tablic, tablica );
Close( plik_tablic );
```

Składowanie struktury pojedynczymi elementami:

```
Assign(plik_liczb, 'nazwa');
Rewrite(fe);
for i := 1 to il_elem do
   Write( plik_liczb, tablica[ i ] );
Close( plik_liczb );
```

Odzyskiwanie struktury elementami:

```
Assign( plik_liczb, 'nazwa' );
Reset( plik_liczb );
i := 1;
while not EOF( plik_liczb ) do
   begin
      Read( plik_liczb, tab[ i ] );
      i := i + 1;
   end;
Close( plik_liczb );
il_elem := i;
```

WYBRANE SCHEMATY PRZETWARZANIA PLIKÓW

Definicje i deklaracje wykorzystywane w przykładach:

```
TYPE
   Element = xyz ;   { xyz - dowolny typ dozwolony jako element pliku}
VAR
   plik, plik_zrodlowy, plik_wynikowy : file of Element;
   i, pozycja, n : integer
   dane : Element;
   Znaleziono : boolean;
```

Dołączenie elementu do końca istniejącego pliku

```
Assign( plik, 'nazwa' );
Reset( plik );
Seek( plik, FileSize( plik ) );      {ustawienie wskaźnika pliku na końcu}
Write( plik, dane );
Close( plik );
```

Wybieranie elementów w/g wartości (w/g pola rekordu)

```
Assign( plik, 'nazwa' );
Reset( plik );
i := 0;                        {wyzerowanie licznika poszukiwanych elementów }
while not EOF( plik ) do
   begin
      Read( plik, dane );
      {sprawdzenie zgodności wartości badanej z wzorcową}
      if dane = WartoscPoszukiwana then
         begin
            i := i + 1;                  {zwiększenie wartości licznika }
            OpracujDane( dane );     { np. wyświetl dane na ekranie }
         end;
   end;
Writeln( 'Znaleziono ',  i  ,'elementów o poszukiwanej wartosci' );
Close( plik );
```

Wyszukiwanie elementu w/g unikalnej wartości

```
Assign( plik, 'nazwa' );
Reset( plik );
Znaleziono := FALSE;    {inicjacja zmiennej sygnalizującej znalezienie}
while  not (  EOF(plik) or Znaleziono  )  do
   begin
      Read( plik, dane );
      {sprawdzenie zgodności wartości badanej z wzorcową}
      if dane = WartoscPoszukiwana then
         begin
            Znaleziono := TRUE;      {zaznaczenie odnalezienia elementu}
            OpracujDane( dane );     { np. wyświetl dane na ekranie }
         end;
   end;
Close( plik );
```

Po „znalezieniu" wskaźnik pliku znajduje się na pozycji zaraz po znalezionym elemencie. Odczytując pozycję wskaźnika można ustalić położenie/pozycję tego elementu w pliku:

```
pozycja := FilePos( plik ) – 1;
```

Element o znanym położeniu może potem zostać zaktualizowany lub usunięty.

Aktualizacja elementu o znanym położeniu w pliku

```
Assign( plik, 'nazwa' );
Reset( plik );
Seek( plik, pozycja );     {ustawienie wskaźnika pliku przed elementem}
Write( plik, dane );       {zapis nowej / aktualnej wartości}
Close( plik );
```

Usunięcie elementu o znanym położeniu (dla plików nieuporządkowanych)

```
Assign( plik, 'nazwa' );
Reset( plik );

{odczyt elementu znajdującego się na końcu pliku}
Seek( plik, FileSize(plik) - 1 ) );
Read( plik, dane );

{zapis odczytanego ostatniego elementu na 'kasowanej' pozycji }
Seek( plik, pozycja );
Write( plik, dane );

{ 'obcięcie' ostatniego elementu → skrócenie pliku o 1 element}
Seek( plik, FileSize(plik) - 1 ) );
Truncate( plik );
Close( plik );
```

Usunięcie elementu z pliku uporządkowanego wymaga kompresji pliku !

- kompresja przez zsunięcie elementów (czasochłonna!)
- skopiowanie całego pliku do drugiego pliku, (z pominięciem elementu usuwanego).
- logiczne kasowanie elementu w pliku (zaznaczanie, że element jest „usunięty").

Przetwarzanie pliku źródłowego w wynikowy o tej samej nazwie

```
Assign( plik_zrodlowy, 'nazwa_org');
Assign( plik_wynikowy, 'nazwa_rob' )
Reset( plik_zrodlowy );
Rewrite( plik_wynikowy );
while not EOF( plik_zrodlowy ) do
   begin
      Read( plik_zrodlowy, dane);
      OpracujDane( dane );
      Write( plik_wynikowy, dane);
   end;
Close( plik_zrodlowy);
Erase( plik_ zrodlowy);                    {skasowanie pliku źródłowego}
Close( plik_wynikowy);                     {zamknięcie pliku wynikowego}
Rename( plik_wynikowy, 'nazwa_org' );      {przywrócenie nazwy }
```