Pliki. Operacje na plikach w Pascalu

ścieżka zapisu, pliki elementowe, tekstowe, operacja plikowa, etapy, assign, zmienna plikowa, skojarzenie, tryby otwarcia, reset, rewrite, append, read, write, buforowanie operacji dyskowych, close, fsearch

Obsługa plików w Pascalu

Wszelkie dane przechowywane w pamięci RAM komputera ulegają zniszczeniu po zakończeniu wykonywania programu. Pisząc coraz bardziej zaawansowane aplikacje, napotkamy w końcu na sytuację w której zachodzi konieczność zapisania wartości niektórych zmiennych, bądź efektów pracy programu - np. wykonanych obliczeń.

Podobnie w grach komputerowych nie chcemy utracić przebiegu gry - np. rozwoju naszej postaci:

O plikach słów kilka (1)

Plik (file) to porcja informacji, stanowiąca całość, zapisana w pamięci masowej w postaci zbioru bajtów. Pełna nazwa pliku składa się z dwóch członów:

- nazwy, nadawanej przez autora pliku o długości do 255 znaków (nazwa nie może zawierać znaków: / \ * ? < >)
- rozszerzenia, określającego typ pliku, np. txt, avi, cpp, pdf

Ścieżka dostępu do pliku to ciąg znaków określający położenie dowolnego obiektu w strukturze folderów na dysku twardym lub innym nośniku. Najczęściej ma postać listy folderów odseparowanych ukośnikami. Ścieżka dostępu może być określana jako ścieżka *bezwzględna* albo *względna*:

- bezwzględna ścieżka dostępu rozpoczyna się od folderu głównego poprzedzonego symbolem dysku, na którym umiejscowiony jest ten folder, np. *C:\programowanie\omg\srlsly\szkola\1TI\plik.txt*

- względna ścieżka dostępu przedstawia lokalizację pliku lub folderu względem folderu bieżącego - np: *..\1TI\plik.txt*

O plikach słów kilka (2)

Dane przechowywane w pliku mogą mieć reprezentację binarną (taką samą, jak w pamięci komputera) lub tekstową (taką, jaka używana jest do wprowadzania informacji z klawiatury i wyprowadzania jej na ekran monitora lub drukarkę). Reprezentacjom tym odpowiadają w Pascalu pliki:

- elementowe (inaczej nazywane binarnymi)
- tekstowe

Zawartość plików elementowych jest na ogół nieczytelna dla użytkownika, natomiast treść pliku tekstowego daje się łatwo odczytać, np. w Notatniku

Wszystkie dane przechowywane w plikach elementowych muszą być tego samego typu.

Pliki tekstowe mogą być użyte do przechowywania mieszanych typów danych (np. tekstów i liczb), gdyż wszelka informacja przechowywana jest w nich w postaci kolejnych linii z zawartością.

O plikach słów kilka (3)

Pliki tekstowe umożliwiają również formatowanie zapisu i korzystanie z procedur **readln** i **writeln**, które są niedostępne dla plików binarnych.

Pliki elementowe umożliwiają tzw. dostęp swobodny, co znaczy iż w każdym momencie można odwołać się do dowolnego elementu pliku.

Pliki tekstowe są plikami o dostępie sekwencyjnym, co oznacza, że aby dostać się do wybranego elementu pliku, należy przeczytać wszystkie elementy znajdujące się przed nim.

Aby móc używać pliku deklaruje się tzw. zmienną plikową:

```
var
  plik : file of byte //plik elementowy liczb rzeczywistych
  plik : text  // plik tekstowy
```

Schemat operacji dyskowej

Ogólny schemat operacji plikowej w Pascalu obejmuje 4 etapy:

1. Skojarzenie zmiennej plikowej z odpowiednim plikiem (znajdującym się na dysku lub nowo tworzonym)

2. Otwarcie pliku, przygotowujące go do zapisywania lub odczytywania informacji

3. Jedna lub więcej operacji zapisu lub odczytu danych

4. Zamknięcie pliku

Funkcje związane z obsługą plików (1)

assign(zmienna_plikowa, nazwa_pliku) - funkcja ta służy do skojarzenia zmiennej plikowej z plikiem fizycznie znajdującym się na dysku.

nazwa_pliku określa tu ścieżkę do pliku na dysku, do którego chcemy się odwoływać - może zawierać nazwę dysku, katalogu i ewentualnych podkatalogów zawierających plik).

Przykładowe skojarzenie zmiennej plikowej z plikiem może mieć następującą postać:

```
assign(plik, 'save\plik.dat');
```

Funkcje związane z obsługą plików (2)

- **`reset(zmienna_plikowa)`** umożliwia otwarcie już istniejącego pliku, ustawiając tzw. wskaźnik plikowy na jego początku. W przypadku, gdy otwierany plik nie istnieje, wywołanie procedury reset zakończy się błędem wykonania.

- **`rewrite(zmienna_plikowa)`** umożliwia otwarcie pliku niezależnie od tego, czy istniał on poprzednio: jeśli plik nie istnieje utworzony zostanie nowy plik o danej nazwie, zaś jeśli plik istniał - zeruje długość istniejącego pliku i ustawia wskaźnik plikowy na jego początku (czego efektem jest utracenie wszystkich danych zawartych w pliku)

- **`append(zmienna_plikowa)`** otwiera plik do dopisywania, tj. otwiera go do zapisu nie niszcząc poprzedniej zawartości i ustawia wskaźnik plikowy na jego końcu. Umożliwia to dodawanie danych do plików tekstowych.

Funkcje związane z obsługą plików (3)

Warto pamiętać, że w przypadku plików tekstowych procedura **`reset`** otwiera plik wyłącznie do odczytu, zaś **`rewrite`** - wyłącznie do zapisu (nie ma zatem możliwości mieszania odczytów i zapisów w jednym cyklu otwarcia).

Zasada ta nie obowiązuje dla plików binarnych, które można odczytywać i zapisywać bez ograniczeń niezależnie od tego, czy zostały otwarte za pomocą procedury reset, czy rewrite (w tym ostatnim przypadku trzeba najpierw zapisać do pliku jakieś dane).

Funkcje związane z obsługą plików (4)

Do wymiany danych pomiędzy programem a plikiem służą znane nam już procedury **read** (odczyt) i **write** (zapis). Ponieważ w "standardowej" wersji obsługują one ekran monitora i klawiaturę, niezbędne jest podanie dodatkowego argumentu określającego plik, z/do którego informacja ma być odczytana lub zapisana. Argumentem tym jest właśnie nazwa odpowiedniej zmiennej plikowej:

```
read(zmienna_plikowa, lista_elementów);
write(zmienna_plikowa, lista_elementów);
```

Powyższe operacje odnoszą się zarówno do plików elementowych, jak i tekstowych. Dla tych ostatnich możliwe jest ponadto użycie procedur readln i writeln, odczytujących lub zapisujących dane wraz ze znakami końca wiersza. Ponieważ pliki elementowe przechowują wyłącznie dane określonego typu i nie mogą zawierać znaków końca wiersza, użycie procedur readln i writeln jest w ich przypadku niedozwolone.

Funkcje związane z obsługą plików (5)

Po wykonaniu żądanych operacji zapisu i odczytu danych plik należy zamknąć. Ta bardzo ważna operacja dająca pewność, że dane zostały zapisane na dysku. W systemie operacyjnym zachodzi tzw. *buforowanie operacji dyskowych*, czyli technika polegająca na odczytywaniu i zapisywaniu danych nie pojedynczo, lecz całymi paczkami, za pośrednictwem specjalnego obszaru pamięci - tzw. bufora dyskowego.

Wykorzystanie bufora pozwala na zredukowanie liczby fizycznych odczytów i zapisów na dysku, a przez to zmniejszenie jego mechanicznego obciążenia. Ponieważ jednak podczas zapisu zawartość bufora wysyłana jest na dysk dopiero po jego zapełnieniu lub w chwili zamknięcia pliku, przerwanie wykonywania programu może spowodować utratę danych.

Zamknięcie pliku realizuje procedura **`close`**:

```
close(zmienna_plikowa);
```

Funkcje związane z obsługą plików (6)

fsearch(sciezka_do_pliku) to funkcja, która sprawdza czy plik o nazwie skojarzonej ze zmienną plikową rzeczywiście istnieje na dysku.

Przykład wykorzystania funkcji file search:

```
if (fsearch('dane/plik.txt','')='') then
begin
  writeln('PLIK NIE ISTNIEJE');
  readln;
end
else
begin
  //procedury odczytu lub zapisu do pliku
end;
```

Uwaga: aby użyć tej funkcji należy dodać do listy uses moduł DOS