Робота з файлами в PASCAL

Що таке файл?

- Оксфордський тлумачний словник англійської мови подає слово "файл" так:
  - *це будь-який із різноманітних типів висувних шухляд, полиць, коробок тощо, як правило, з металевим стержнем для тримання паперів разом і в порядку так, щоб їх легко було відшукати;*
  - *це ряд людей або речей, розташованих одне за одним.*
- Мова програмування Pascal:
  - ***Файл*** *— це самостійна послідовність символів, записана в постійну пам'ять комп'ютера. Це певна виділена область інформації.*
  - Існування файлів не залежить від роботи якої-небудь програми і вони нікуди не зникають навіть при включенні і виключенні комп'ютера.
  - Файли можуть зберігати в собі різну інформацію. Це тексти, програми, віруси, картинки, інтернет сторінки і так далі.

Коли і навіщо треба використати файли?

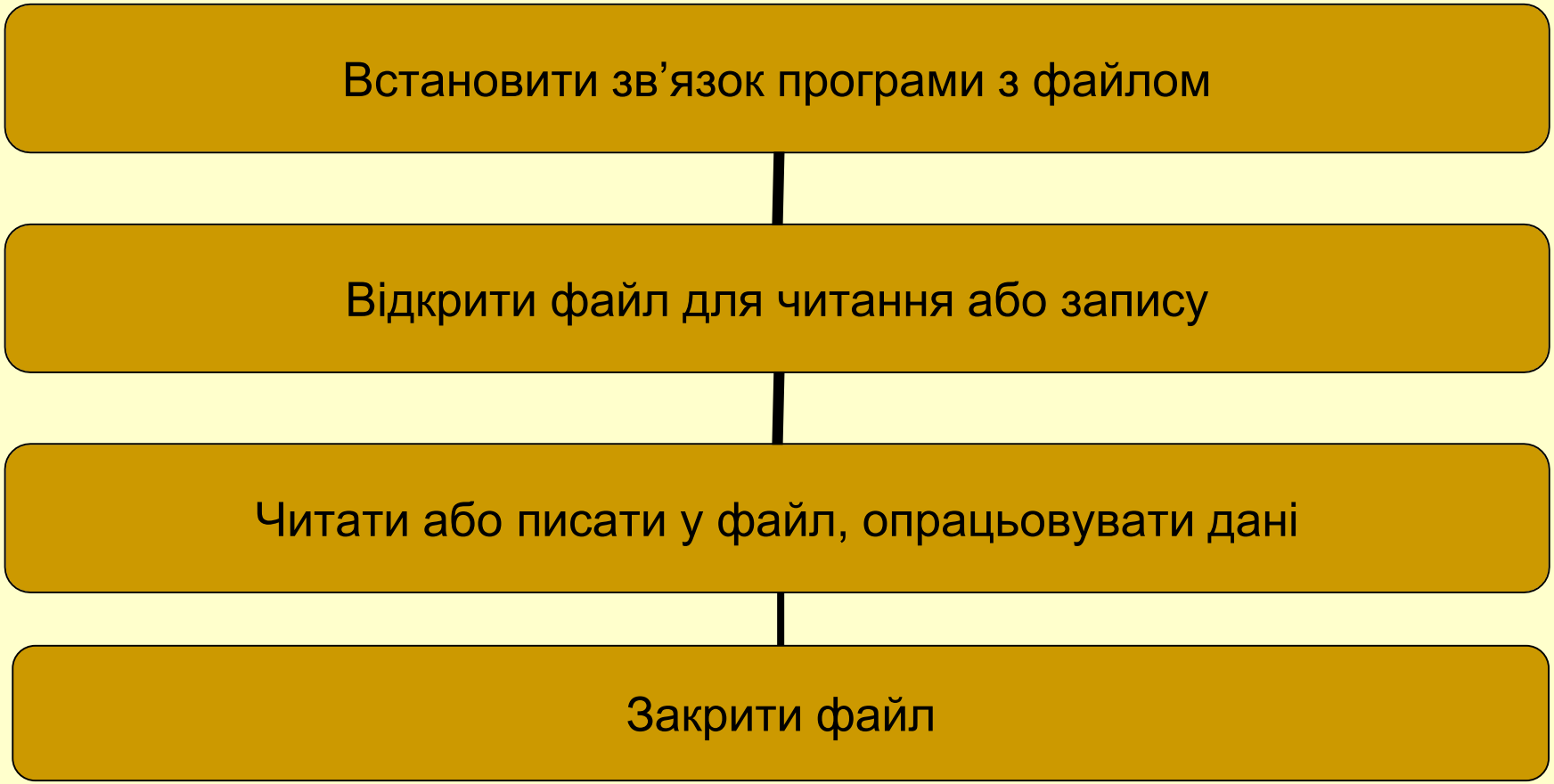
- необхідно зберігати вихідні дані при відладці;
- велика кількість вхідних даних (той, хто вручну тестував програми із заповненням матриць, мене розуміє);
- багатократне введення однієї і тієї ж інформації, з мінімальними змінами або зовсім без змін;

Як описати файлові змінні?

- Турбо Паскаль підтримує три файлових **типи**:
  - текстові файли;
  - типізовані файли;
  - нетипізовані файли.
- Доступ до файлу в програмі відбувається за допомогою змінних файлового типу. Змінну файлового типу **описують** одним з трьох способів:
  - file of тип - типізований файл (вказаний тип компоненти);
  - text - текстовий файл;
  - file - нетипізований файл.
- **Приклади** опису файлових змінних:

```
var f1: file of char;
    f2: file of integer;
    t: text;
    f3: file;
```

Послідовність роботи з файлом



Стандартні процедури та функції

- Assign(f, FileName);

- пов'язує файлову змінну f з фізичним файлом, повне ім'я якого задано в рядку *FileName*. Наприклад:

  Assign (F1, 'a:\Tp5\DAT\St629.DAT');
  Assign (F2, 'Dannye.DAT'); - *буде розташовано в поточному каталозі*

Увага!

Встановлений зв'язок буде чинним до кінця роботи програми, або до тих пір, поки не буде зроблено перепризначення.

Стандартні процедури та функції

- Reset(f)

- відкриває для читання файл, з яким пов'язана файлова змінна f.

Після успішного виконання процедури Reset файл готовий до читання з нього першого елемента.

Увага!

Процедура завершується з повідомленням про помилку, якщо зазначений файл не знайдений.

Якщо f - типізований файл, то процедурою reset він відкривається для читання і запису одночасно.

Стандартні процедури та функції

- Rewrite(f)

- відкриває для запису файл, з яким пов'язана файлова змінна *f*.

Після успішного виконання цієї процедури файл готовий до запису в нього першого елемента.

Увага!

Якщо вказаний файл вже існував, то всі дані з нього видаляються.

Стандартні процедури та функції

- Close(f)

- закриває відкритий до цього файл з файлової змінної *f*.

Увага!

Виклик процедури *Close* необхідний при завершенні роботи з файлом. Якщо з якоїсь причини процедура Close не буде виконана, файл усе-таки буде створений на зовнішньому пристрої, але вміст останнього буфера в нього не буде перенесено.

Стандартні процедури та функції

- **EOF(f): boolean**

- повертає значення TRUE, коли при читанні досягнутий кінець файлу.

- **EOLn(f): boolean**

- повертає значення TRUE, коли при читанні досягнутий кінець рядка текстового файлу.

- **Rename(f, NewName)**

- дозволяє перейменувати фізичний файл на диску, пов'язаний з файловою змінною f. Увага! Перейменування можливо після закриття файлу.

- **Erase(f)**

- видаляє фізичний файл на диску, який був зв'язаний з файловою змінною f. Увага! Файл до моменту виклику процедури Erase має бути закритий.

- **Append(f)**

- Процедура відкриває текстовий файл для додавання інформації до його кінця. Використовуйте цю процедуру замість Rewrite.

Робота з текстовими файлами

Текстовий файл - це сукупність рядків, розділених мітками кінця рядка. Сам файл закінчується міткою кінця файлу. Доступ до кожного рядка можливий лише послідовно, починаючи з першого. Одночасний запис і читання заборонені.

Робота з текстовими файлами

Читання з текстового файлу:

Read(f, список змінних);
ReadLn(f, список змінних);

Спосіб читання залежить від типу змінних, що стоять в списку. У змінну char записуються символи з файлу. Запис у числову змінну відбувається за таким принципом: пропускаються символи-роздільники, початкові пробіли і зчитується значення числа до появи наступного роздільника. У змінну типу string записується кількість символів, яка дорівнює довжині рядка, але тільки в тому випадку, якщо до цього не зустрілися символи кінця рядка або кінця файлу.

Відмінність ReadLn від Read полягає в тому, що перша команда після прочитання даних пропустить решту символів рядку, включаючи мітку кінця рядка. Якщо список змінних відсутній, то процедура ReadLn(f) пропускає рядок при читанні текстового файлу.

Робота з текстовими файлами

Запис в текстовий файл:

Write(f, список змінних);
WriteLn(f, список змінних);

Спосіб запису залежить від типу змінних в списку (як і при виведенні на екран). Враховується формат виводу. WriteLn від Write відрізняється тим, що після запису всіх значень зі змінних записує ще й символ кінця рядка (формується закінчений рядок файлу).

Приклади 1-2

Запис у текстовий файл

```
Var F: Text;
      S: byte;
Begin
  Assign (F,'XXX.TXT');
  Rewrite(F); {відкриття файла}
  ReadLn (S); {введення з
                          клавіатури}
  WriteLn (F,S); {запис у файл}
  Close (F) {закриття файла}
End.
```

Читання з текстового файла з виведенням на екран

```
Var   F: Text;
        S: byte;
Begin
  Assign (F,'XXX. TXT');
  Reset(F): {відкриття файла}
  ReadLn(F, S); {введення
                          з файлу}
  WriteLn (S); {виведення на екран}
  Close(F) {закриття файла}
End.
```

Перевірити роботу програм для s: string; - задати значення s='1 2 3'

Приклади 3-4

Файл xxx.in:

1 2 3

4 5 6

7 8 9

Читання 3 чисел з текстового файла з виведенням на екран

```
Var   F: Text;
        a,b,c: byte;
Begin
  Assign (F,'XXX. in');
  Reset(F): {відкриття файла}
  ReadLn(F, a,b,c); {введення
                          з файлу}
  WriteLn (a,' ',b,' ',c); {виведення на
     екран}
  Close(F) {закриття файла}
End.
```

На екрані буде 1 4 7

```
Var   F: Text;
        a,b,c: byte;
Begin
  Assign (F,'XXX. in');
  Reset(F): {відкриття файла}
  Read(F, a,b,c); {введення
                          з файлу}
  WriteLn (a,' ',b,' ',c); {виведення на
     екран}
  Close(F) {закриття файла}
End.
```

На екрані буде 1 2 3

Приклади 5-6

Запис у текстовий файл 10 чисел

```
Var F: Text;
      I:byte; a:real;
Begin
  Assign (F,‘task2.out’);
  Rewrite(F); {відкриття файла}
  For i:=1 to 10 do
    Begin
      ReadLn (a); {введення з клавіатури}
       WriteLn (F,a); {запис у файл}
    end;
  Close (F) {закриття файла}
End.
```

Читання з текстового файла 10 чисел та визначення їх суми

```
Var   F: Text;
        I:byte; S, a:real;
Begin
  Assign (F, ‘task2.out’);
  Reset(F): {відкриття файла} S:=0;
For i:=1 to 10 do
    Begin
     ReadLn(F, a); {введення з файлу}
     s:=s+a;
    end;
  WriteLn (S); {виведення на екран}
  Close(F) {закриття файла}
End.
```

Приклади 7-8

Запис у текстовий файл N чисел

```
Var F: Text;
      I, N:byte; a:real;
Begin
  Assign (F,‘task3.out’);
  Rewrite(F); {відкриття файла}
   readln(N); Writeln(F,N);
  For i:=1 to N do
    Begin
      ReadLn (a); {введення з
      клавіатури}
       WriteLn (F,a); {запис у файл}
    end;
  Close (F) {закриття файла}
End.
```

Читання з текстового файла N чисел та визначення їх суми

```
Var   F: Text;
        I, N:byte; S, a:real;
Begin
  Assign (F,’ task3.out’);
  Reset(F): {відкриття файла} S:=0;
  readln(F, N);
For i:=1 to N do
    Begin
     ReadLn(F, a); {введення з файлу}
     s:=s+a;
    end;
  WriteLn (S); {виведення на екран}
  Close(F) {закриття файла}
End.
```

Приклади 9

Читання з текстового файла невизначеної кількості чисел та обчислення їх суми

```
Var   F: Text;
        I, N:byte; S, a:real;
Begin
  Assign (F,' task4.out');
  Reset(F): {відкриття файла}
 S:=0;
  While not(eof(F)) do {доки не досягнуто мітки кінця файлу}
    Begin
     ReadLn(F, a); {введення з файлу}
     s:=s+a;
    end;
  WriteLn (S); {виведення на екран}
  Close(F) {закриття файла}
End.
```

Запитання: Що буде обчислювати програма, якщо команду s:=s+a; замінити на If (a>1) and (a<=5) Then s:=s+a;

Приклади 10

Читання з текстового файла N чисел та визначення найменшого з них

```
Var   F: Text;
        I, N:byte; min, a:real;
Begin
  Assign (F,' task3.out');
   Reset(F); {відкриття файла}
 readln(F, N);
 read(F, min); {прочитати з файлу перший елемент як змінну min}
For i:=2 to N do
    Begin
     Read(F, a); {введення з файлу інших елементів}
     If a<min then min:=a;
    end;
  WriteLn (min); {виведення на екран}
  Close(F) {закриття файла}
End.
```

Приклади 11-12

Читання з текстового файла масива з N чисел та визначення найменшого елементу

```
Var   F: Text;
        I, N:byte; min:real;
        a:array [1..100] of real;
Begin
  Assign (F,' task3.out');
  Reset(F);
  readln(F, N);
  read(F, min);
  For i:=2 to N do
    Begin
     Read(F, a[I]);
     If a[I]<min then min:=a[I];
    end;
 Close(F);
```

WriteLn ('min=',min:8:4); ***{виведення на екран}***

```
 End.
```

Читання з текстового файла масива з N чисел та визначення найменшого елементу. Відповідь вивести наприкінці файлу даних.

```
Var   F: Text;
        I, N:byte; min:real;
        a:array [1..100] of real;
Begin
  Assign (F,' task3.out');
  Reset(F);
  readln(F, N);
  read(F, min);
  For i:=2 to N do
    Begin
     Read(F, a[I]);
     If a[I]<min then min:=a[I];
    end;
 Close(F);
```

Append(f)

WriteLn (f, ' min=',min:8:4); ***{дозапис у файл}***

Close(F);

```
 End.
```

Завдання

1. Дано текстовий файл, який містить цілі числа. Визначити:
   - Кількість парних елементів
   - Суму додатних елементів
   - Середнє арифметичне найменшого і найбільшого елементів
2. Дано файл F, компонентами якого є цілі числа. Переписати у файл G ті з них, які:
   - Є парними числами
   - Менші за задане ціле число
   - Діляться на 3 і не діляться на 7

Робота з типізований файлами

- **Типізований файл** - це послідовність компонент будь-якого заданого типу (окрім типу "файл"). Доступ до компонент файлу здійснюється по їх порядковими номерами. Компоненти нумеруються, починаючи з 0. Після відкриття файлу покажчик (номер поточної компоненти) стоїть на його початку, на нульовому компоненті. Після кожної операції читання або запису покажчик зсувається до наступного компоненту.

Робота з типізований файлами

- **Запис** у файл: Write(f, список змінних);
- **Читання** з файлу: Read(f, список змінних);

Тип файлових компонент має відповідати типу змінних. Якщо буде зроблена спроба читання неіснуючих компонент, то відбудеться помилкове завершення програми. Необхідно або точно розраховувати кількість компонент, або перед кожним читанням даних робити перевірку їх існування (функція eof)

- **Зсув покажчика файлу** на n-нну позицію: Seek(f, n);

Нумерація у файлі починається з 0.

- Визначення **кількості компонент**: FileSize(f): longint;
- Визначення **позиції покажчика**: FilePos(f): longint;
- **Відсікання останніх компонент** файлу, починаючи з поточної позиції включно: Truncate(f)

Приклади 13-14

Переглянути всі елементи файла в зворотному порядку

```
Var   F: file of real;
        I, N:byte; a:real;
Begin
  Assign (F,' task5.in');
 Reset(F);
 readln(F, N);
For i:=1 to N do
    Begin
     Seek(f,n-i);
     ReadLn(F, a);
      WriteLn (a);
  end;
 Close(F)
End.
```

Читання з типізованого файла всіх чисел та визначення їх суми

```
Var   F: file of real;
        I, N:byte; S, a:real;
Begin
  Assign (F,' task6.in');
  Reset(F);
  S:=0;
For i:=1 to FileSize(f) do
    Begin
     ReadLn(F, a);
     s:=s+a;
    end;
  WriteLn (S);
  Close(F)
End.
```

Для типізованих файлів дані можна занести у файл лише програмно!

Завдання

1. Дано типізований файл, який містить цілі числа. Визначити:
   - Кількість парних елементів
   - Суму додатних елементів
   - Середнє арифметичне найменшого і найбільшого елементів
2. Дано типізований файл F, компонентами якого є цілі числа. Переписати у файл G ті з них, які:
   - Є парними числами
   - Менші за задане ціле число
   - Діляться на 3 і не діляться на 7

Робота з нетипізованими файлами

Нетипізовані файли - це послідовність компонент довільного типу.

- Відкриття нетипізовані файлу:

  Reset(f, BufSize)
  Rewrite(f, BufSize)

Параметр BufSize задає число байтів, що записуються в файл за одне звертання або зчитуються з нього. Мінімальне значення BufSize - 1 байт, максимальне - 64 Кбайт. Якщо пропустити цей параметр, то він отримає типове значення рівне 128 байтам.

Робота з нетипізованими файлами

Читання даних з нетипізований файлу:

- BlockRead(f, X, Count, QuantBlock);

Ця процедура здійснює за одне звертання зчитування в змінну *X* такої кількості блоків, яка задана параметром *Count*, при цьому довжина блоку дорівнює довжині буфера. Значення Count не може бути менше 1. За одне звернення не можна прочитати більше, ніж 64 Кбайт.

Необов'язковий параметр QuantBlock повертає число блоків, прочитаних поточною операцією BlockRead. У разі успішного завершення операції читання QuantBlock = Count, у разі аварійної ситуації параметр QuantBlock буде містити число вдало прочитаних блоків. Звідси випливає, що за допомогою параметра QuantBlock можна контролювати правильність виконання операції читання.

Робота з нетипізованими файлами

Запис даних у нетипізований файл:

- BlockWrite(f, X, Count, QuantBlock);

Ця процедура здійснює за одне звертання запис зі змінної X кількості блоків, яка задана параметром Count, при цьому довжина блоку дорівнює довжині буфера.

Необов'язковий параметр QuantBlock повертає число блоків, записаних успішно поточною операцією BlockWrite.

- Для нетипізованих файлів можна використовувати процедури Seek, FIlePos і FileSize, аналогічно до відповідних процедур для типізованих файлів.