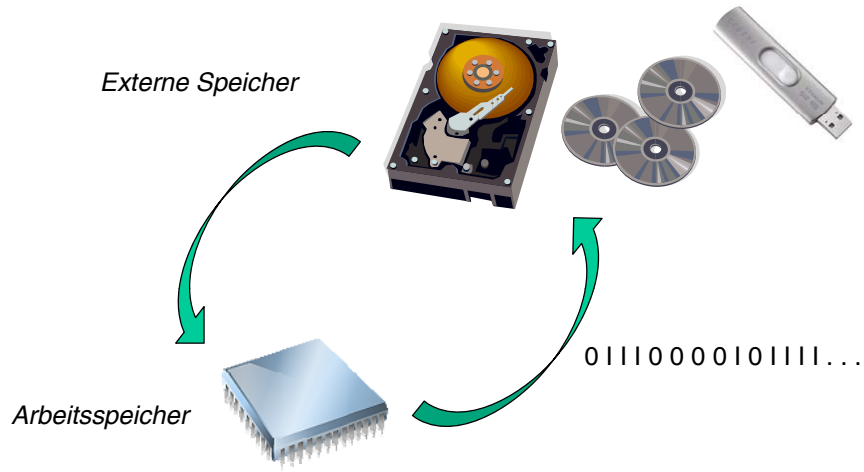
Agenda für heute, 22. April 2010

- **Externe Daten verarbeiten**
- Die wichtigsten Operationen
- Spezialfall Textfiles
- Direkte Files

Lesen Sie den Begleittext Seite 70 - 75

Externe Daten verarbeiten



Der Datentyp `File`

Filename Basistyp

```
var
   Datei1: file of integer;
   Datei2: file of boolean;
   Datei3: file of char;
```

Zuordnung eines Files zu einer Datei

```
AssignFile(Datei3, 'G:\Meindok.txt');
```

Filevariable — Verzeichnispfad und Dateiname

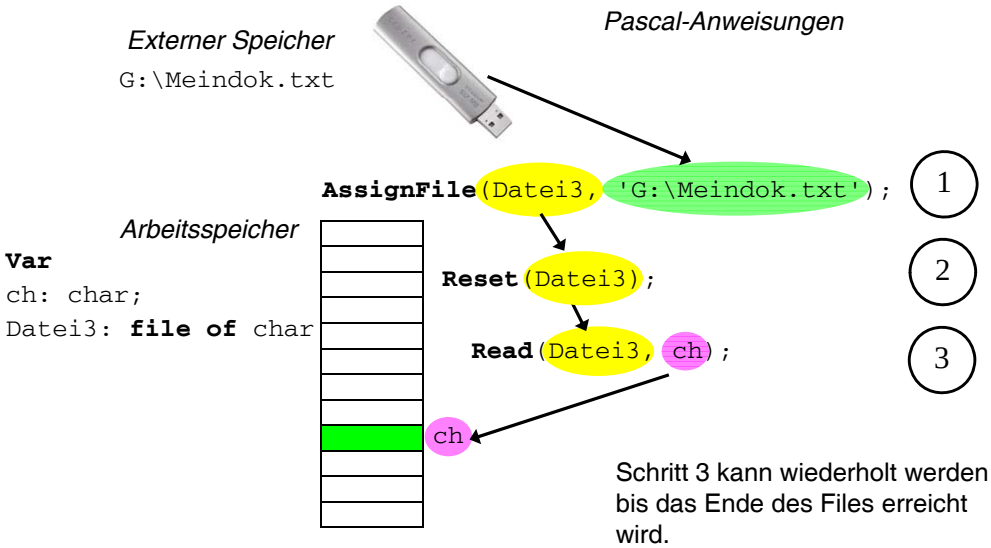
In Turbo Pascal: `Assign(Datei3, 'G:\Meindok.txt')`

Standardprozeduren für Files

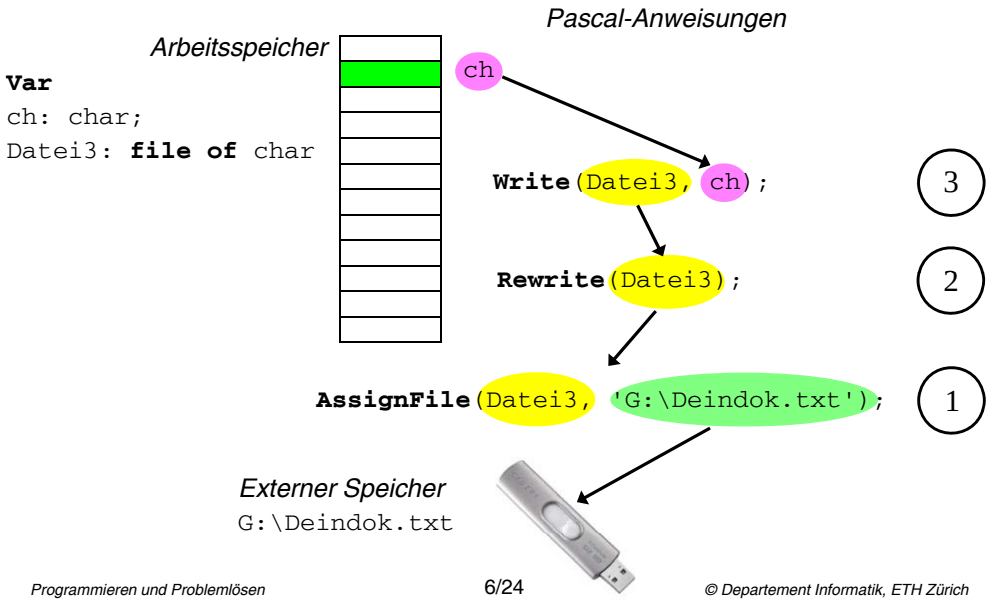
Verarbeitungsschritte

1. File öffnen: `Reset(Filevariable)`
   `Rewrite(Filevariable)`
   `Append(Filevariable)`
2. File lesen: `Read(Filevariable, Variable)`
3. File schreiben: `Write(Filevariable, Variable)`
4. File-Ende testen: `Eof(Filevariable)`
5. File schliessen: `Close(Filevariable)`

Zeichen aus einer Textdatei lesen

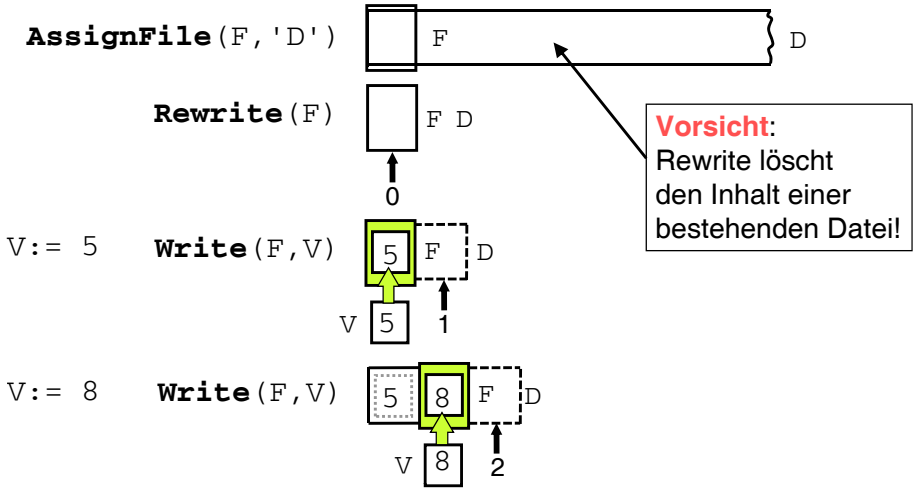


Zeichen in eine Textdatei schreiben



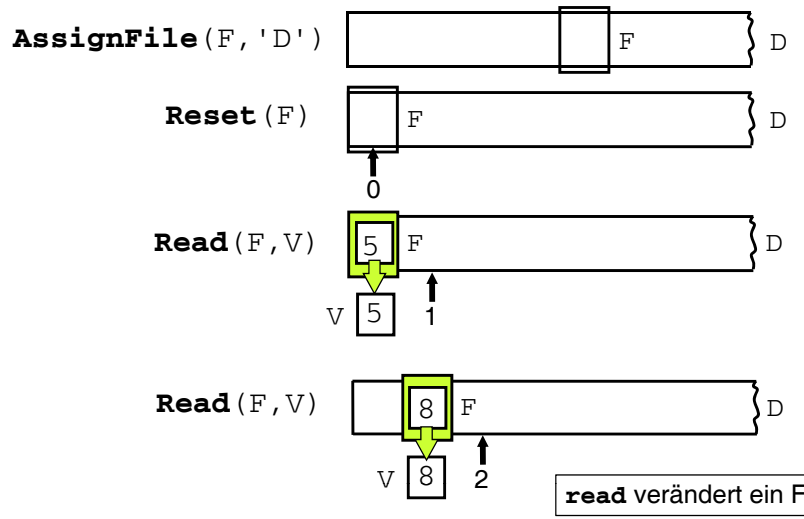
Schreiben einer sequentiellen Datei

Schematische Darstellung



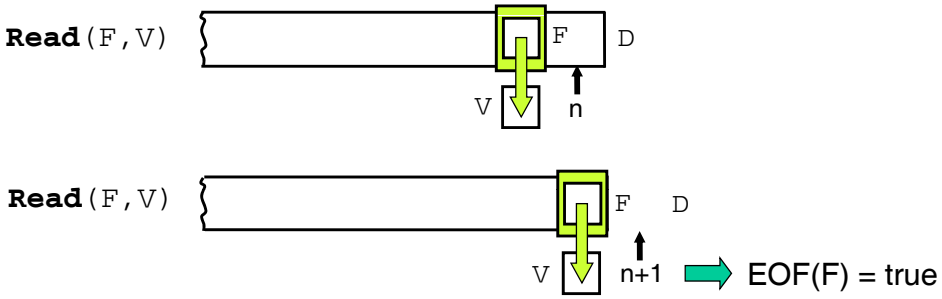
Lesen einer sequentiellen Datei

Schematische Darstellung



Lesen einer sequentiellen Datei

Schematische Darstellung



Eine weitere Read-Anweisung würde zu einem Laufzeitfehler führen

Nicht-typisierte Files

```
var
   Eingabe: file;        Basistyp Byte
   Eingabe: Textfile;    Basistyp Text (zeilenweise)
```


- Externe Daten verarbeiten
- **Die wichtigsten Operationen**
- Spezialfall Textfiles
- Direkte Files

Öffnen eines bestehenden Files mit `Reset`

```
var
   Messwert: Real;
   Eingabe: file of Real;
begin
   AssignFile(Eingabe, 'C:\Messungen');
   Reset(Eingabe);
```

- Nicht vorhandene Datei führt zu Eingabe/Ausgabe-Fehler.
- Bereits offene Files werden zuerst geschlossen und dann erneut geöffnet, mit Sicht auf die erste Komponente.
- Auf die Filevariable kann in ihrem Sichtbarkeitsbereich zugegriffen werden.

Lesen eines Files mit **Read**

```
AssignFile(Eingabe, 'C:\Messungen')
Reset(Eingabe);
if NOT EOF(Eingabe)
then Read(Eingabe, Messwert);
```

- File und Variable müssen vom gleichen Typ sein.
- Mit **EOF** (End Of File) prüfen ob überhaupt eine Filekomponente vorhanden ist.
- Nochmaliges Aufrufen von **Read** liest die nächste Filekomponente.

Öffnen eines neuen Files zum Schreiben mit **Rewrite**

```
var
ch: char
Ausgabe: file of char;
begin
  AssignFile(Ausgabe, 'C:\Meintext.txt');
  Rewrite(Ausgabe);
```

- Ein File darauf vorbereiten, Daten in eine neue Datei zu schreiben.
- Der Inhalt einer gleichnamigen Datei, wird gelöscht und an ihrer Stelle die neue Datei angelegt.
- Bereits offene Files werden zuerst geschlossen und dann erneut geöffnet, mit Positionszeiger am Anfang des Files.

Schreiben eines Files mit **Write**

```
AssignFile(Ausgabe, 'C:\Meintext.txt');
Rewrite(Ausgabe);
Write(Ausgabe, ch);
```

- File und Variable müssen vom gleichen Typ sein.
- Nochmaliges Aufrufen von **Write** schreibt die nächste Filekomponente.

Schliessen eines Files mit **CloseFile**

```
AssignFile(Eingabe, 'C:\Messwerte');
Reset(Eingabe);
   { Datei Messwerte verarbeiten }
CloseFile(Eingabe);
AssignFile(Eingabe, 'D:\NeueWerte');
Reset(Eingabe);
   { Datei NeueWerte verarbeiten }
```

- CloseFile stellt sicher, dass *alle* Daten in die Datei geschrieben werden (In Turbo Pascal: Close).
- Nach CloseFile steht die Filevariable für andere (oder nochmals die gleiche) Datei zur Verfügung.

- Externe Daten verarbeiten
- Die wichtigsten Operationen
- **Spezialfall Textfiles**
- Direkte Files

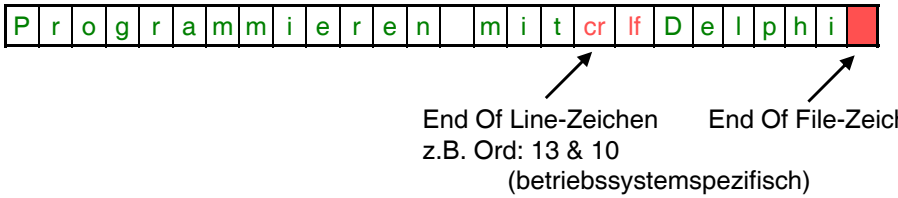
Eigenschaften von Textfiles

Textfiles speichern lineare Folgen von Textzeichen. Sie haben zusätzlich eine Zeilenstruktur, die durch spezielle Zeichen, welche das Ende einer Zeile markieren, gegeben ist.

Der Text:

Programmieren mit
Delphi

würde wie folgt gespeichert:



Textfiles lesen

```
var
  Dokument: Textfile;
  Zeile: String;
  Zeichen: Char;
begin
  Reset(Dokument);
  Read(Dokument, Zeile); {liest ganze Zeile}
  Readln(Dokument);  {liest Zeilenende-Zeichen}
{ oder }
  Readln(Dokument, Zeile);  {liest beides}
```

In Turbo Pascal: `Dokument: Text;`

Textfiles lesen

Ein Zeichen nach dem anderen bis zum Ende einer Zeile lesen:

```
while not EOLN(Dokument) do
  Read(Dokument, Zeichen);
Readln(Dokument);
```

Die Standardfunktion EOLN (für End Of Line) liefert "TRUE" sobald das nächste zu lesende Zeichen ein Zeilenende-Zeichen ist.

!! Vorsicht: ein **Readln** innerhalb der Schleife würde beim ersten Aufruf ein Zeichen lesen, gleichzeitig alle Zeichen bis zum End Of Line ignorieren und das Zeilenende-Zeichen lesen.

Textfiles lesen

- Aus Textfiles gelesene Zahlen werden automatisch in die entsprechenden numerischen Werte vom Typ Integer oder Real konvertiert.
- Entspricht die Zahl nicht dem erwarteten Format, tritt ein E/A-Fehler auf.

```
var
  Dokument: Textfile;
  Wert_1, Wert_2: Integer;
begin
  Read(Dokument, Wert_1, Wert_2);
```


Textfiles schreiben

Textfiles akzeptieren Werte vom Typ *Char*, *String*, *Boolean*, einem *Integer*-Typ oder einem *Real*-Typ (automatische Konvertierung).

```
var
  Dokument: Textfile;
  Zeile: String; Wert: Real;
begin
  Rewrite(Dokument);    { erzeugt leeres File }
  Zeile:= 'ETH Zuerich';
  Write(Dokument, Zeile);
  Writeln;     { fügt Zeilenende-Zeichen ein }
  Wert:= 37.85;
  Writeln(Dokument, Wert);
```

```
ETH Zuerich
37.85
```


Text anfügen

```
var
  Dokument: Textfile;
  Zeile: String;
  Leitzahl: Integer;
begin
  Append(Dokument);
  Zeile:= 'Ort: ';
  Leitzahl:= 8092;
  Writeln(Dokument, Zeile, Leitzahl, ' Zuerich');
```

Ist das File bereits offen, wird es zuerst geschlossen und dann erneut geöffnet.

```
ETH Zuerich
37.85
Ort: 8092 Zuerich
```


- Externe Daten verarbeiten
- Die wichtigsten Operationen
- Spezialfall Textfiles
- **Direkte files**

Direkte Filezugriffe

```
var
   Eingabe: file of Integer;
   Messung: Integer;
begin
   AssignFile(Eingabe,'C:\Messwerte')
   Reset(Eingabe);
   Seek(Eingabe, 20);
   Read(Eingabe, Messung);
   Seek(Eingabe, 79);
   Write(Eingabe, Messung)
```

Schreibt die 21. Komponente des Files ***Eingabe*** in die 80. Position des selben Files.

Direkte Filezugriffe

Beispiel

Ein File vergrössern:

```
Seek(F, FileSize(Eingabe));
```

Dieser Prozeduraufruf setzt den Positionszeiger ans Ende des Files.

Hinweis

`FileSize` kann nicht für Textdateien verwendet werden.

Zusammenfassung: Arbeiten mit Files

- Für das File muss eine Filevariable deklariert sein.
- Das File muss mit einem dem Betriebssystem bekannten, ev. noch zu kreierenden, Dateinamen in Verbindung gebracht werden.
- Das File muss durch das Programm geöffnet werden.
- Über die *Filevariable* wird in die Datei geschrieben oder aus der Datei gelesen.
- Das File muss durch das Programm geschlossen werden.