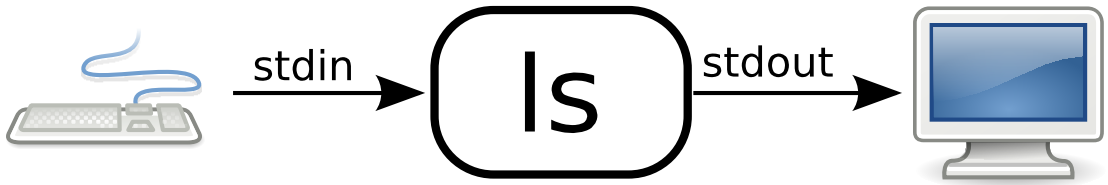
Pipes

Pipes

Ein- und Ausgabe-Streams

Programme...

- lesen ihre Eingabe von der Standard-Eingabe (*stdin*)
- schreiben auf die Standard-Ausgabe (*stdout*)

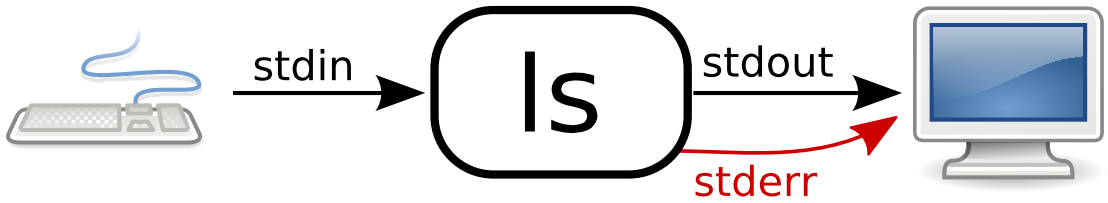


Pipes

Ein- und Ausgabe-Streams

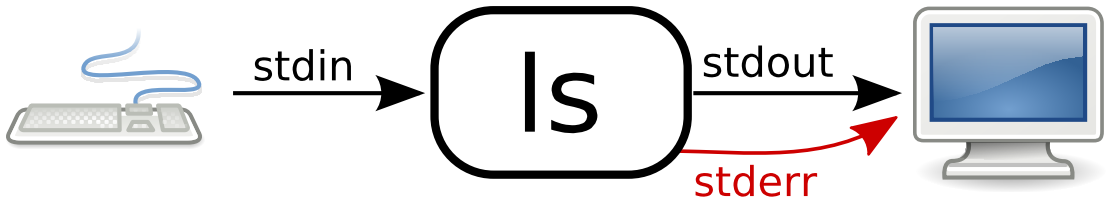
Programme...

- lesen ihre Eingabe von der Standard-Eingabe (*stdin*)
- schreiben auf die Standard-Ausgabe (*stdout*)
- und schreiben Fehlermeldungen auf die Fehler-Ausgabe (*stderr*)



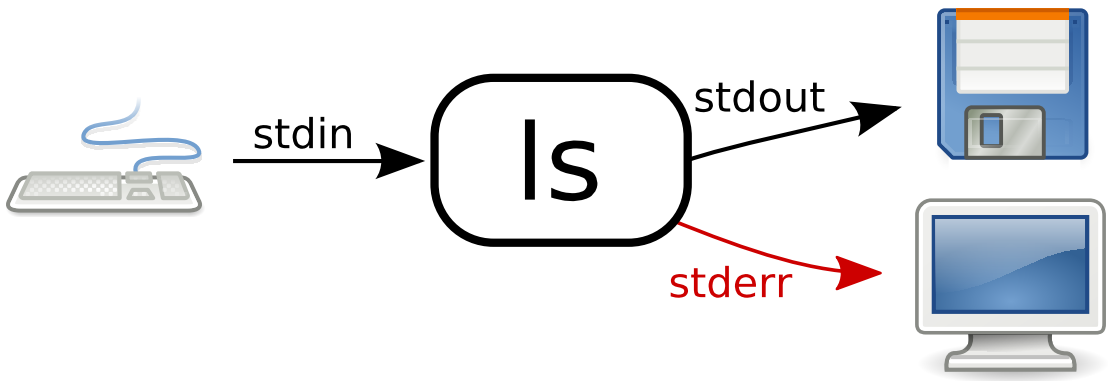
Pipes

Umleiten von Aus- und Eingabe-Streams



Pipes

Umleiten von Aus- und Eingabe-Streams



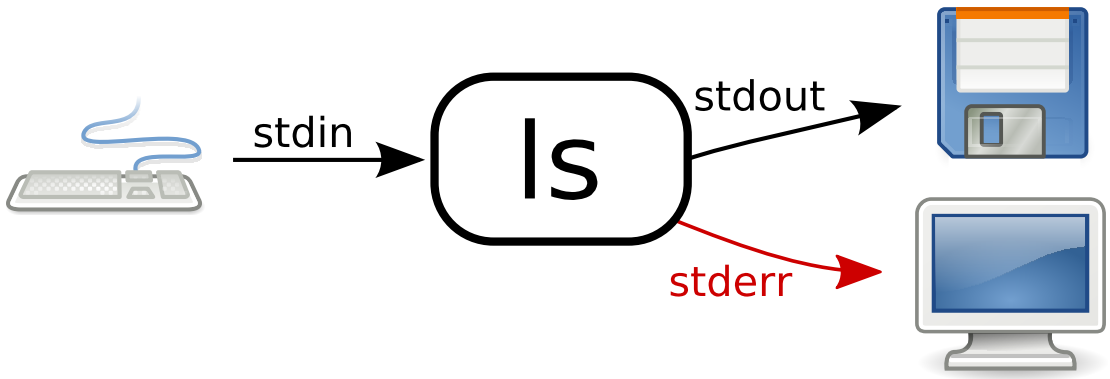
> – Ausgabe umleiten

> leitet *stdout* in eine Datei um.

⟨Befehl⟩ > ⟨Datei⟩

Pipes

Umleiten von Aus- und Eingabe-Streams



Beispiel: Erstellen einer Liste aller Dateien in einem Verzeichnis

```
$ ls
```

. . . und dann die Liste abtippen, oder:

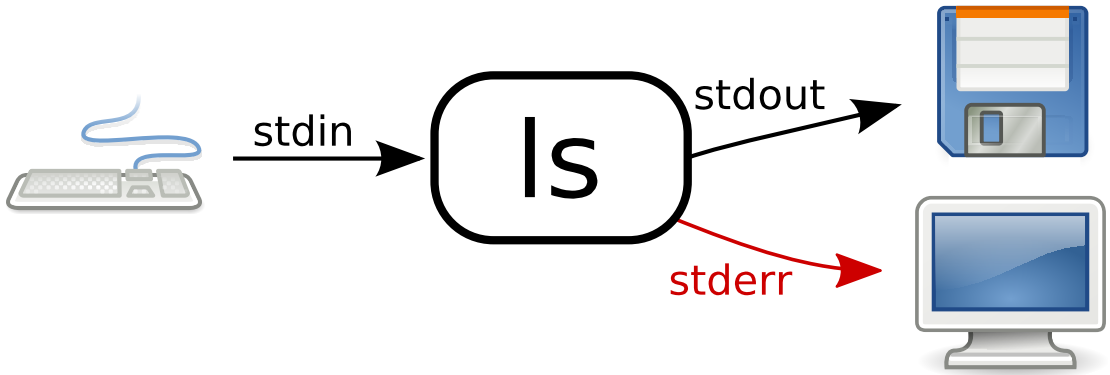
```
$ ls > listeMitDateien.txt
```

Vorsicht

> überschreibt den Inhalt einer Datei!

Pipes

Umleiten von Aus- und Eingabe-Streams



>> – Ausgabe umleiten (und anhängen)

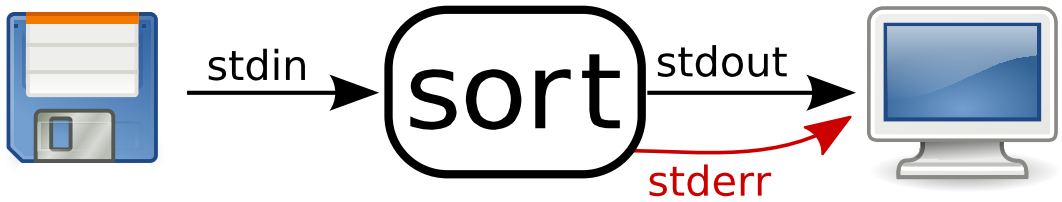
>> leitet *stdout* in eine Datei um, dabei wird alles ans Ende der Datei angehängt.

Beispiel: Erstellen einer Liste aller Dateien aus zwei Verzeichnissen

```
$ ls bilder/ > listeMitDateien.txt
$ ls urlaubsbilder/ >> listeMitDateien.txt
```

Pipes

Umleiten von Aus- und Eingabe-Streams



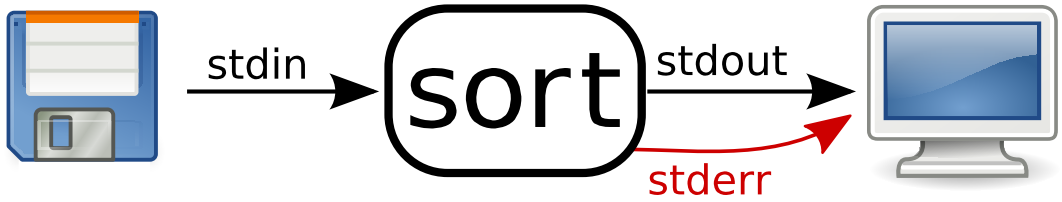
< – Eingabe umleiten

< stellt den Inhalt einer Datei dem Programm auf *stdin* zur Verfügung.

⟨`Befehl`⟩ < ⟨`Datei`⟩

Pipes

Umleiten von Aus- und Eingabe-Streams



Beispiel: Sortieren einer Liste von Dateien.

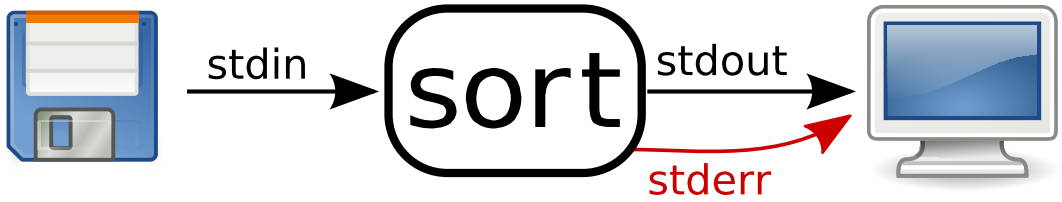
`sort` sortiert die Zeilen, die von *stdin* gelesen werden.

```
$ sort
```

. . . und dann die Liste der Dateien manuell eintippen oder:

Pipes

Umleiten von Aus- und Eingabe-Streams



Beispiel: Sortieren einer Liste von Dateien.

`sort` sortiert die Zeilen, die von *stdin* gelesen werden.

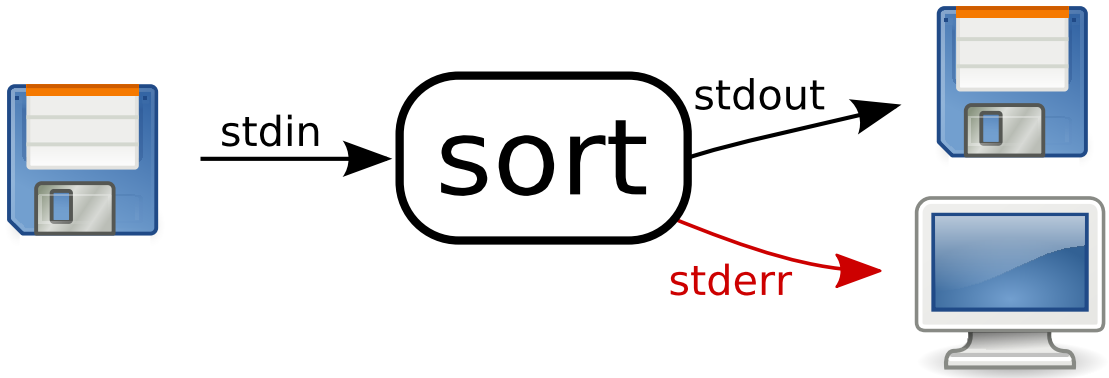
```
$ sort
```

. . . und dann die Liste der Dateien manuell eintippen oder:

```
$ sort < listeMitDateien.txt
alex.jpg
bruno.jpg
...
```

Pipes

Umleiten von Aus- und Eingabe-Streams




```
$ sort < liste.txt > ausgabe.txt
```

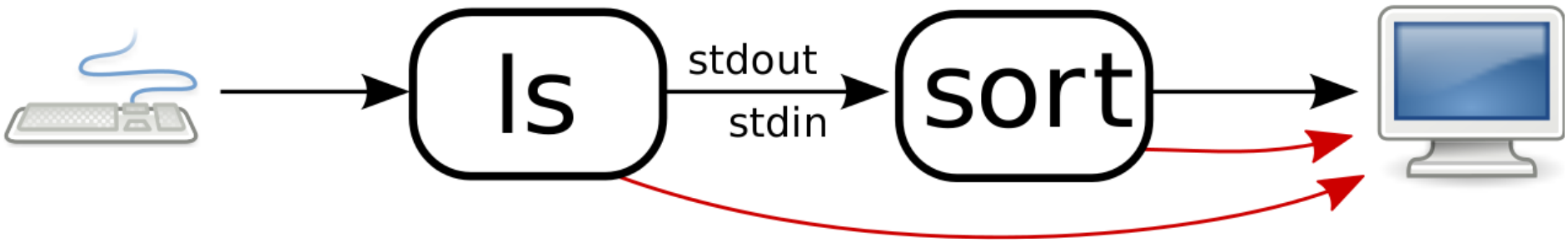
Vorsicht

Falls Eingabe- und Ausgabedatei identisch sind, geschehen seltsame Dinge!

Pipes

Umleiten von Aus- und Eingabe-Streams

Natürlich kann man auch zwei Programme miteinander verbinden.



| – Ausgabe an ein anderes Programm weiterleiten

| („Pipe“) leitet *stdout* von einem Programm zum *stdin* eines anderen Programmes um.

⟨`Befehl 1`⟩ | ⟨`Befehl 2`⟩

Pipes

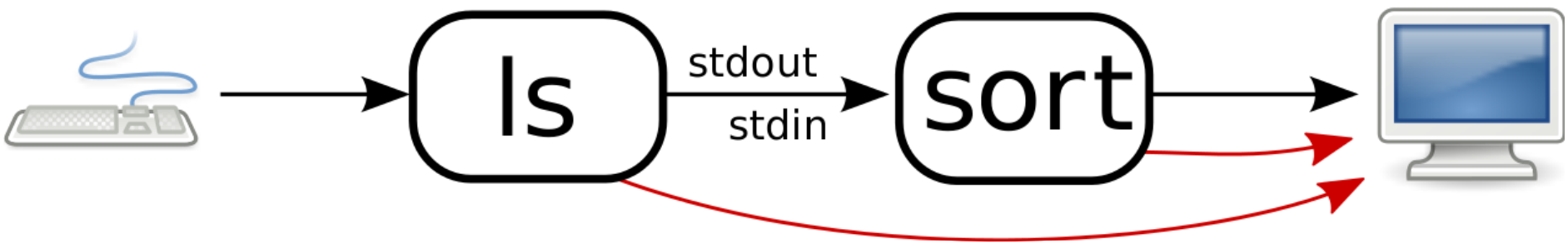
Umleiten von Aus- und Eingabe-Streams

Umständlich: Sortieren einer Liste aller Dateien aus zwei Verzeichnissen.

```
$ ls bilder/ urlaubsbilder/ > listeMitDateien.txt
$ sort < listeMitDateien.txt
```

Pipes

Umleiten von Aus- und Eingabe-Streams



Umständlich: Sortieren einer Liste aller Dateien aus zwei Verzeichnissen.

```
$ ls bilder/ urlaubsbilder/ > listeMitDateien.txt
$ sort < listeMitDateien.txt
```

Besser: In einem Schritt mit Pipe:

```
$ ls bilder/ urlaubsbilder/ | sort
alex.jpg
bruno.jpg
```

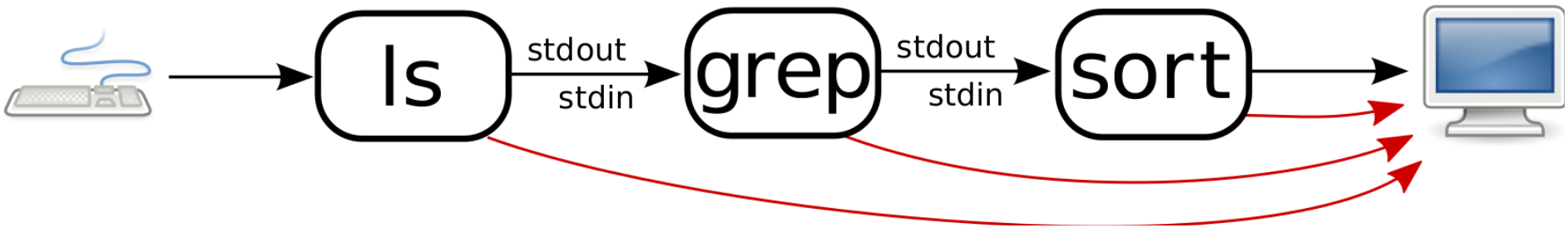
Pipes

Umleiten von Aus- und Eingabe-Streams – beliebig erweiterbar!

Beliebig erweiterbar!

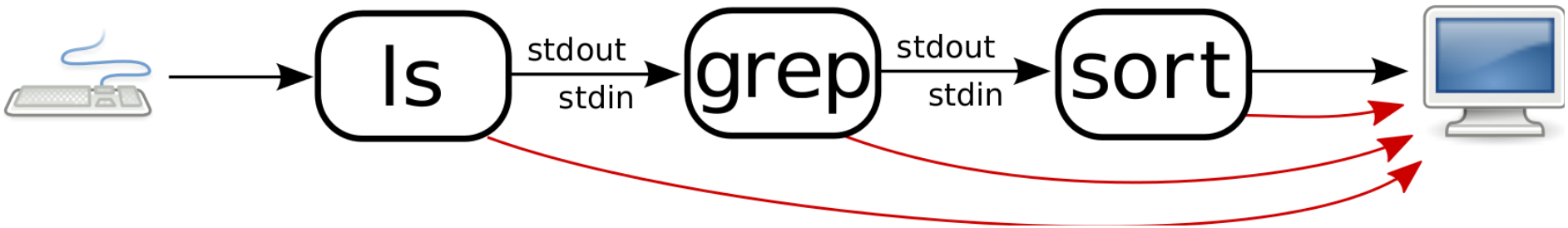
- z. B. können wir eine sortierte Liste von Bildern nach allen Bildern von Peter durchsuchen!

  → `grep` filtert die Liste



Pipes

Umleiten von Aus- und Eingabe-Streams – beliebig erweiterbar!



Beispiel: Eine Liste von Bildern erstellen und diese durchsuchen

```
$ ls bilder/ | grep peter | sort
peter.jpg
peter_muede.jpg
rainer-und-peter.jpg
...
```