Inhaltsverzeichnis

Der Puffer

- Definition: Zwischenspeicher für Daten, die zwischen verschiedenen Bereichen übertragen werden müssen.
- Beispiele: Beim E/A Operationen
  - Eingabe: Tastatur -> Puffer -> Programm
  - Ausgabe: Programm -> Puffer -> Datei

Der Puffer: Vorteile

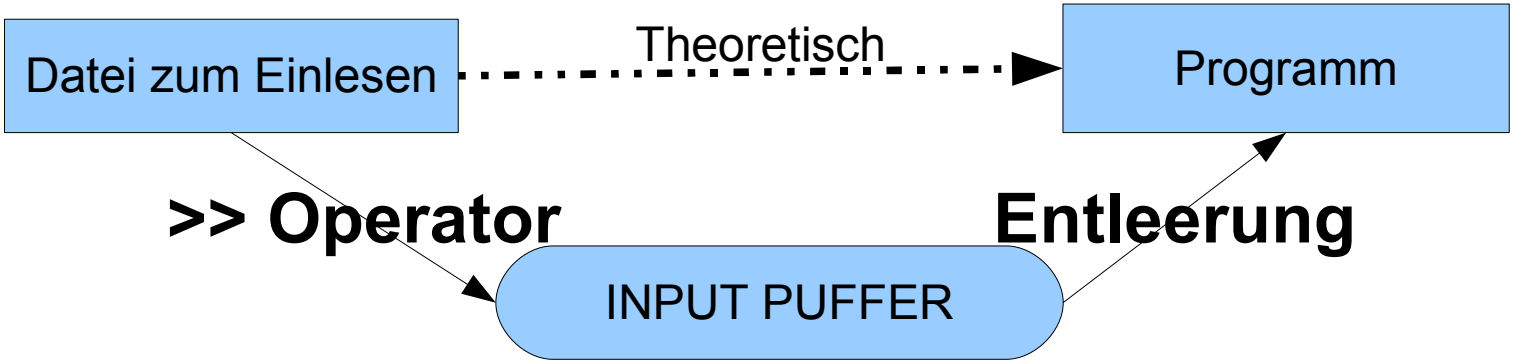
- Sanfte Schnittstelle zwischen Bereichen mit unterschiedlichen Geschwindigkeiten
  - Optimiert die Geschwindigkeit der gesamten Übertragung
  - Optimiert die Kosten (Energie) der gesamten Übertragung.

Der Puffer: in C++ I/O Streams

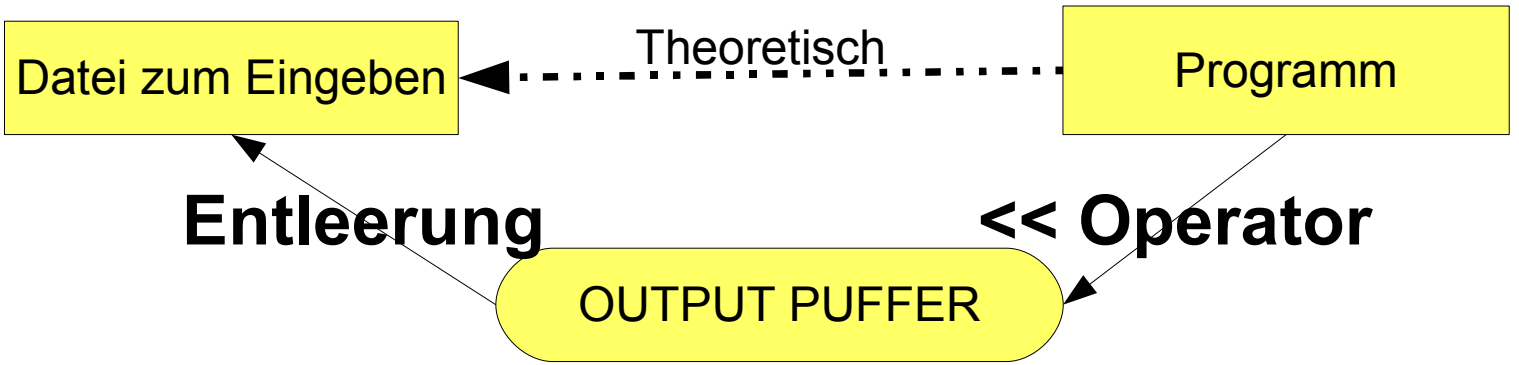
- Sowohl istreams (cin bzw. ifstreams) als ostreams (cout bzw. ofstreams) besitzen eigene Puffer.

Der Puffer: in C++ I/O Streams

- Eingabe aus Datei (ifstream)



- Ausgabe in Datei (ofstream)



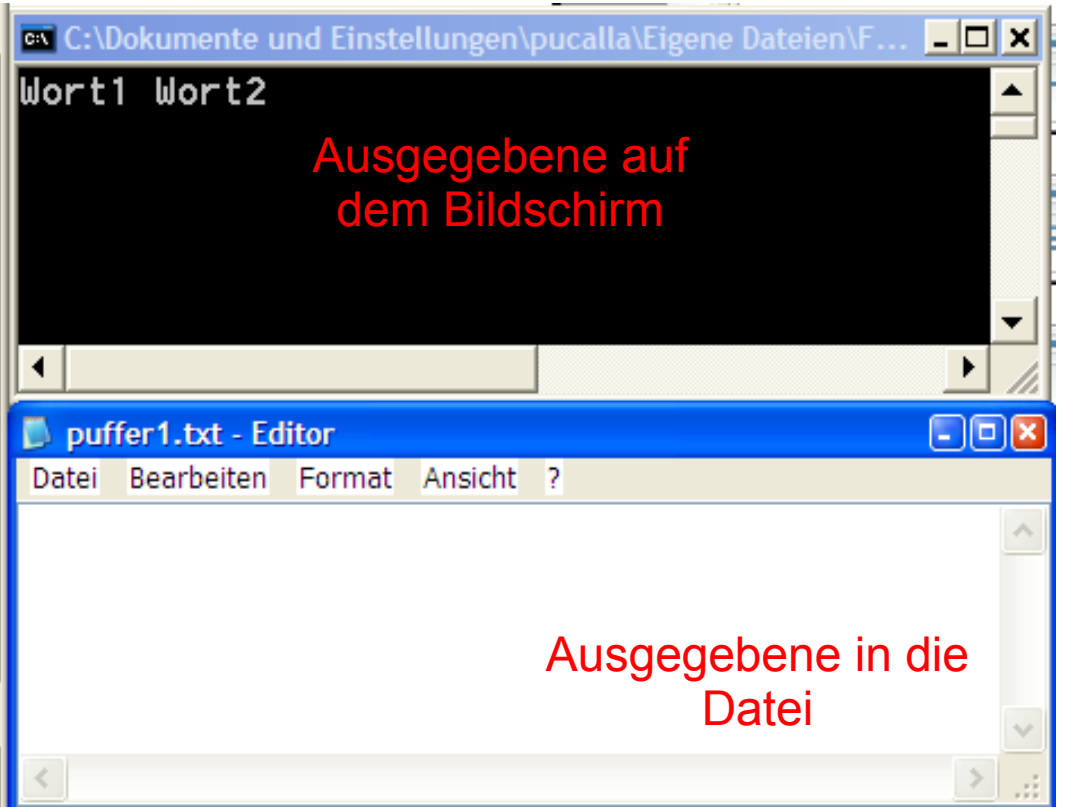
Der Puffer: C++ Funktionen

- Automatische Entleerung des Puffers:
  - Der Puffer wird entleert, wenn der voll ist.
    (weil Puffer eine gewisse Speichergröße haben)
  - Bei Dateien, wenn der Datei geschlossen wird
    (`close()`).
  - Bei der Verwendung von `endl`
- Absichtliche Entleerung
  - ostreams mit der Methode `flush`
  - istreams mit der Methode `sync`

Der Puffer: Beispiele

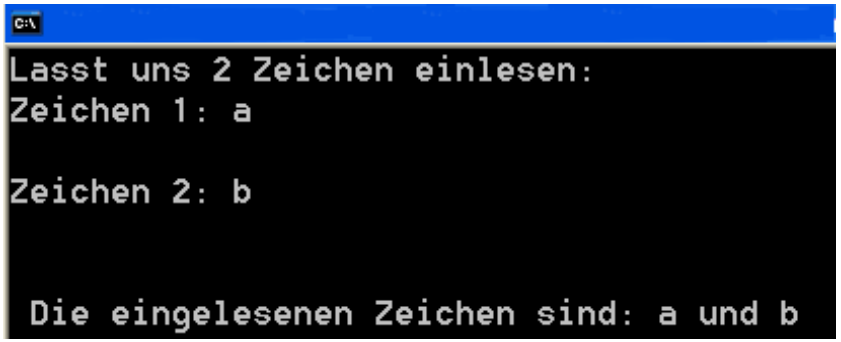
Siehe folgende Programme

- **Puffer1.cpp:** Die Textmeldungen werden auf dem Bildschirm augenblicklich ausgegeben jedoch die Ausgabe in die Datei erfolgt erst nach dem Schließen des Stroms. Davor wird momentan kein Text in der Datei landen. Der bleibt im Puffer (Dies wird sichtbar wenn mit Debugger ausgeführt).

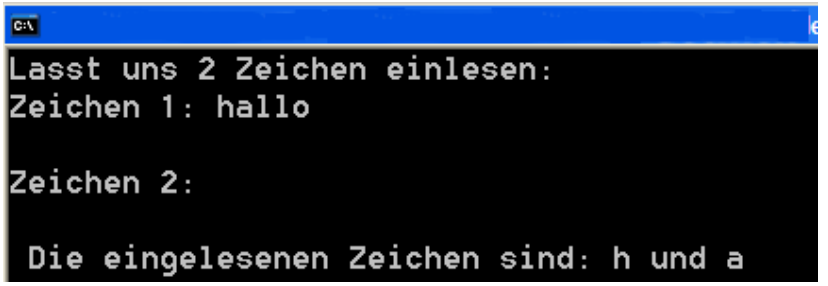


Der Puffer: Beispiele

- Puffer2.cpp: Bei Eingabe von zwei Zeichen kann man die Wirkung eines Puffers wahrnehmen.
  - Normale Eingabe



  - Eingabe eines ganzen Wortes („`hallo`“). Daraus wird „`h`“ eingelesen. Der cin-Puffer behält aber die restliche Zeichen („`allo`“). Daher wird „`a`“ als zweites Zeichen ausgegeben ohne die Möglichkeit weiteres durch Tastatur einzugeben.



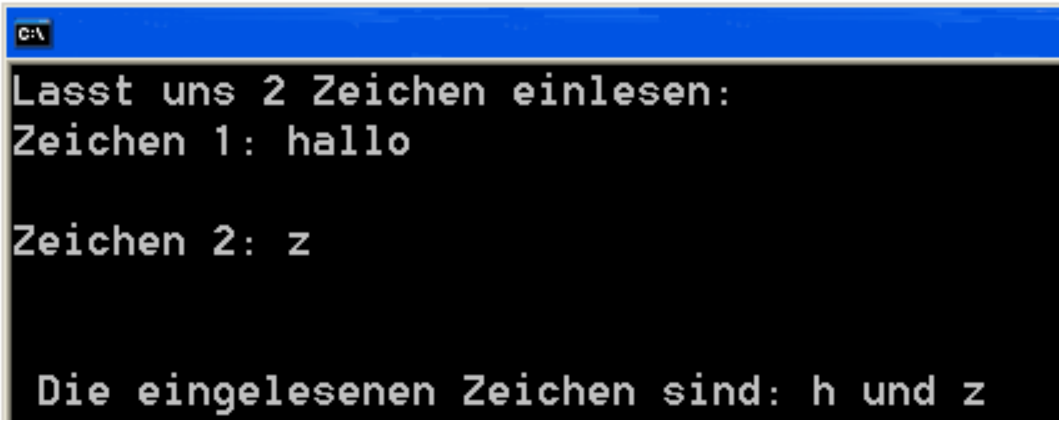
Der Puffer: Beispiele

- Puffer2sync.cpp:

  - Eingabe eines ganzen Wortes („`hallo`“). Daraus wird „`h`“ eingelesen. Der cin-Puffer behält aber die restlichen Zeichen („`allo`“).

    Der cin-Puffer wird nun entleert durch die Methode `sync`

    Daher wird jetzt nicht „`a`“ ausgegeben sondern ein neues Zeichen eingelesen.



Der Debugger („Fehlerbeseitiger“)

- Ein Bug ist auf Englisch ein „Programmierungsfehler“
- Debug bezeichnet auf Englisch die „Fehlerbeseitugung“ auf Deutsch kann man auch den Begriff „Debuggen“ nutzen.
- Ein Debugger ist daher die Anwendung, um Fehler zu beseitigen.

Der Debugger

- Ein Debugger ist Teil einer IDE (Integrierten Entwicklungsumgebung).
- Die Wichtigste Ausführungsfunktionen eines Debuggers sind
  - „Go to cursor“: „Gehe zum Kursor“ um sich in eine bestimmte Position zu stellen.
  - „Next Step“: „Nächster Schritte“ um die Anweisungen einzeln der Reihe nach auszuführen.
  - „Step into“: „Trete hinein“ um in eine Funktion hineinzutreten.

Der Debugger in der IDE Dev-C++

- In der IDE Dev-C++ wird der Debugger so ausgeführt

  Die erste Bedingung ist, dass die .cpp Datei ausführbar sein muss. Dann kann man:

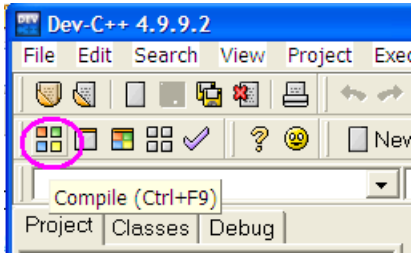
  1. Die Datei kompilieren.
  2. Ein Brechpunkt einsetzen
  3. Debugmodus ausführen.

  Ab diesem Augenblick kann man die Debugfunktionen nutzen.

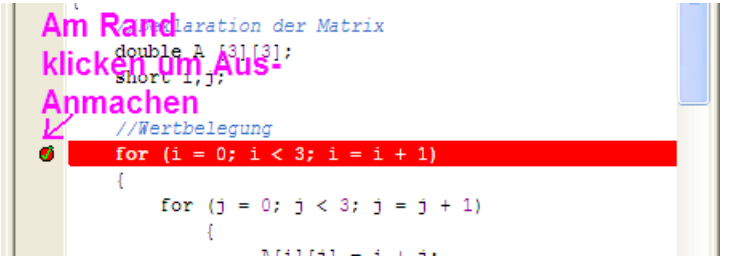
Der Debugger in der IDE Dev-C++

- Veranschaulichung des Prozesses

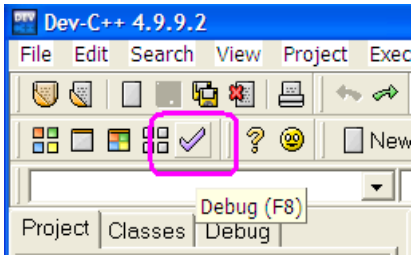
1. Die Datei kompilieren.



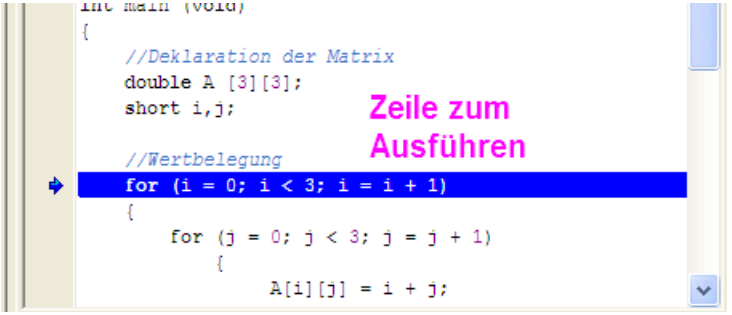
2. Ein Brechpunkt einsetzen



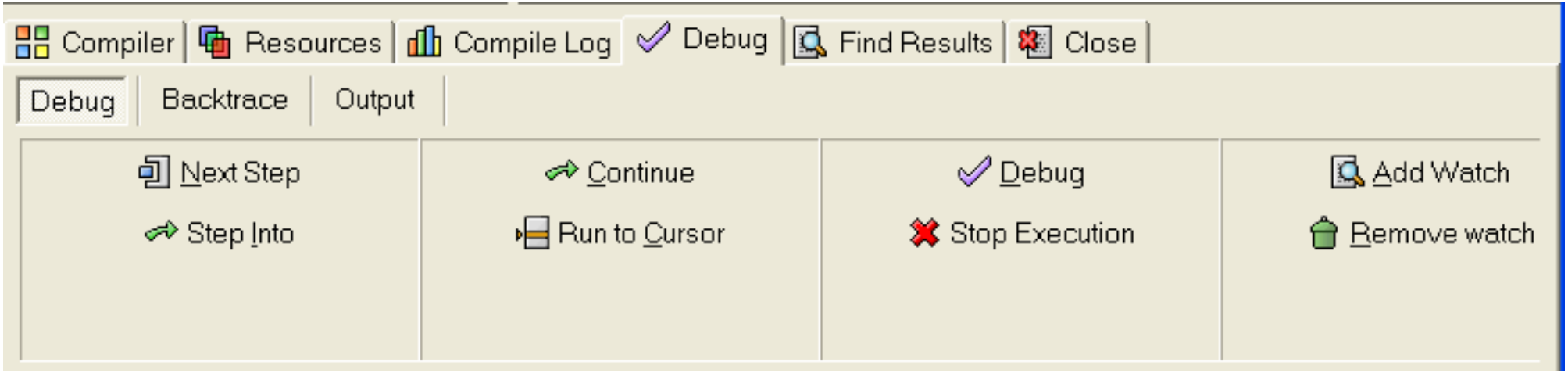
3. Debugmodus ausführen.



Der Debugger in der IDE Dev-C++



Ab diesem Augenblick kann man die Debugfunktionen nutzen.



Der Debugger: „Watch“ - Überwachung

- Die wichtigste Überwachungsfunktion sind die „Watches“
- Ein „Watch“ (beobachten) ist die Anzeige einer bestimmten Variable des Programms.