Kommandozeilenargumente

- Namen von Eingabedateien können
  - fest im Code stehen:
    `std::string namen_der_einzulesenden_datei;`
    $\Rightarrow$ Sehr unschöne Lösung.
  - mit `std::cin` zu Laufzeit eingelesen werden: Bessere Lösung.
- Möglich ist auch ein Aufruf von außen in der Form:
  `./a.out dateiname`
- Für diese Variante ersetzt man die Deklaration von `int main()` durch:
  `int main (int count, char **args);`
- Dann liefert `count` die Zahl der Argumente beim Aufruf des Programms (der Programmname wird mitgezählt).
- `args` liefert einen Pointer auf einen Pointer von `char`, also einen Pointer auf Strings.
- Mit `args[0]`, ..., `args[count-1]` kann auf die einzelnen Strings zugegriffen werden.

Fehlersuche mit gdb I

Idee des Debuggers gdb

- Programm Schritt für Schritt durchgehen oder an wichtige Stellen springen
- Werte von Variablen anzeigen

Starten von gdb

- Kompilieren im Debug-Modus mit `-g`:
  `g++ -std=c++11 -Wall -Wpedantic -g -o EXECUTABLE QUELLDATEI`
- Ausführen des Programms mit gdb:
  `gdb ./EXECUTABLE`
- Es öffnet sich eine gdb-Konsole. Hier ggf. Breakpoints setzen (siehe nächste Folie), dann Programm starten:
  `run`
- Ggf. Eingabeparameter eingeben.

Fehlersuche mit gdb II

Nützliche Befehle

- Breakpoint in Zeile 37 der Datei `program.cpp` setzen:
  `b program.cpp:37`
- Wechsel zwischen gdb-Output-Ansicht und Code-Ansicht:
  Steuerung + x + a
- Fokus-Wechsel zwischen Konsole und Code:
  Steuerung + x + o
- Variable x ausgeben (z.B. Standarddatentypen, std::vector, ...):
  `p x`
- Gehe zur nächsten Zeile der aktuellen Datei:
  `n`
- Gehe zur nächsten Zeile der aktuellen Datei, aber springe in andere Funktionen rein:
  `s`
- Fahre mit dem Programm fort bis zum nächsten Breakpoint:
  `c`

Die Standardbibliothek

Nützliches aus der Standardbibliothek:

- Kleine Helfer, wie die Funktion `std::min(a,b)` bzw. `std::max(a,b)` oder auch `std::swap`
- Sortierfunktion `std::sort`
- Komplexe Zahlen (mit `#include<complex>`)
- Zahlreiche Datenstrukturen, um Mengen zu verwalten.

$\Rightarrow$ Es lohnt sich, dort nach Lösungen zu suchen.