

Programmiersprachen  
Vorlesung 12 vom 21.12.23  
Weihnachtsvorlesung

Christoph Lüth

Universität Bremen

Wintersemester 2023/24

# Was machen wir heute?

- ▶ Eine kleine Sprache
- ▶ Eine besondere Sprache

# Eine kleine Sprache

# Kleine Sprachen

- ▶ Wieviele Anweisungen braucht man **mindestens**, um turingmächtig zu sein?

# Kleine Sprachen

- ▶ Wieviele Anweisungen braucht man **mindestens**, um turingmächtig zu sein?
- ▶ **Eine** reicht.
- ▶ Beweis: Subleq (Subtract and Branch on result Less than or Equal to zero)
- ▶ Beispiel für OISC (One Instruction Set Computer)

# Subleq Basics

► Syntax:

```
subleq a b c
```

► Semantik:

$$\begin{aligned}r &\leftarrow \text{mem}[B] - \text{mem}[A] \\ \text{mem}[B] &\leftarrow r \\ pc &\leftarrow \begin{cases} pc + C & r \leq 0 \\ pc + 1 & \text{otherwise} \end{cases}\end{aligned}$$

# Das ist genug?

- ▶ Invariante:  $mem[0] = 0$  (genannt  $Z$ )
- ▶ Unbedingte Sprünge: `branch c`

```
subleq 0, 0, c    ;; 0 := 0- 0 = 0  
                ;; if 0 <= 0 then PC := PC+ C
```

- ▶ Addition: `add a b`

```
subleq a, 0, 0    ;; Z := 0- a = -a  
subleq 0, b, 0    ;; b := b- Z = b+ a  
subleq 0, 0, 0    ;; Z := Z- Z = 0
```

- ▶ Kopieren (Move): `mov a b`

```
subleq b, b, 0    ;; a := b- b = 0  
subleq a, 0, 0    ;; Z := Z- a = -a  
subleq 0, b, 0    ;; b := b- Z = 0+ a  
subleq 0, 0, 0    ;; Z := 0
```

Source: [https://en.wikipedia.org/wiki/One-instruction\\_set\\_computer#Subtract\\_and\\_branch\\_if\\_less\\_than\\_or\\_equal\\_to\\_zero](https://en.wikipedia.org/wiki/One-instruction_set_computer#Subtract_and_branch_if_less_than_or_equal_to_zero)

# Ist das Turing-mächtig?

- ▶ Nicht ganz, aber es demonstriert das Prinzip.
- ▶ Weitere Instruktionen: **add** a b c, **beq** a c
- ▶ Formaler Beweis: Übersetzung einer Registermaschine nach Subleq.
- ▶ Nützlichkeit von Subleq:
  - ① Einfach in Hardware zu implementieren
  - ② Basis für kompakte Mikroarchitektur

# Eine besondere Sprache

# Brainfuck

- ▶ Brainfuck wurde 1993 von Urban Müller erfunden, um den “kleinstmöglichen Compiler für eine Turing-vollständige Sprache” zu schreiben.
- ▶ Abgeleitet von  $P''$  (Corrado Boehm, 1964)
- ▶ Acht Kommandos, Turing-vollständig
- ▶ Kryptisch und von keinem praktischen Nutzen

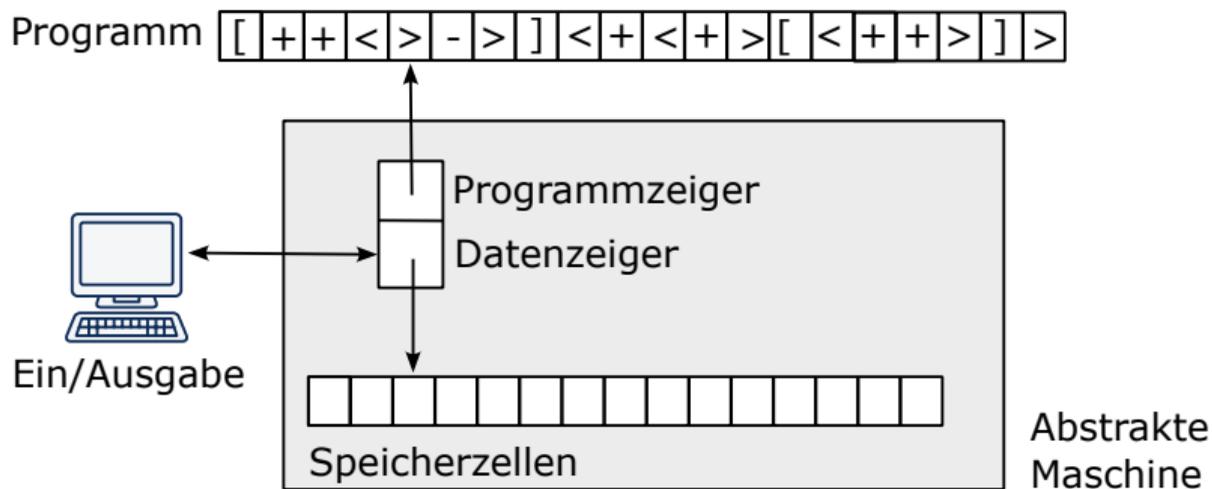
# Ein Beispielprogramm

► Hello, world:

```
+++++++  
[>++++[>+>+>+>+>+<<<<-]>+>+>->>+[<]<-]  
>>.>— .+++++..++.>>.<-.  
<.+ + . ——— . ——— .>>+.>+ + .
```

# Die Sprache

- ▶ Syntax: acht Kommandozeichen
- ▶ Lexikalik: alles andere ist Kommentar
- ▶ Ausführungsmodell:



# Kommandos

<	Datenzeiger erhöhen
>	Datenzeiger erniedrigen
+	Wert der aktuellen Zelle erhöhen
-	Wert der aktuellen Zelle erniedrigen
.	Wert der aktuellen Zelle ausgeben
,	Wert der aktuellen Zelle einlesen
[	Springt hinter das entsprechende ] wenn Wert der aktuellen Zellen 0 ist
]	Springt hinter das entsprechende [ wenn Wert der aktuellen Zellen nicht 0 ist

► [P] ist Iteration von P solange aktueller Zellenwert ungleich 0 ist.

# Einfache Programme

▶ , [ . , ]

# Einfache Programme

▶ `,[ . ,]`      Echo

▶ `[>+<-]`

# Einfache Programme

- ▶ `,[ . ,]`      Echo
- ▶ `[>+<-]`      Addition  $p[i+1] = p[i+1] + p[i]$
- ▶ `[>-<-]`

# Einfache Programme

- ▶ `,[ . ,]`      Echo
- ▶ `[>+<-]`      Addition  $p[i+1] = p[i+1] + p[i]$
- ▶ `[>-<-]`      Subtraktion  $p[i+1] = p[i+1] - p[i]$
- ▶ `>[-]<[>+<-]`

# Einfache Programme

- ▶ `,[ . ,]` Echo
- ▶ `[>+<-]` Addition  $p[i+1] = p[i+1] + p[i]$
- ▶ `[>-<-]` Subtraktion  $p[i+1] = p[i+1] - p[i]$
- ▶ `>[-]<[>+<-]` Verschieben  $p[i+1] = p[i]; p[i] = 0$

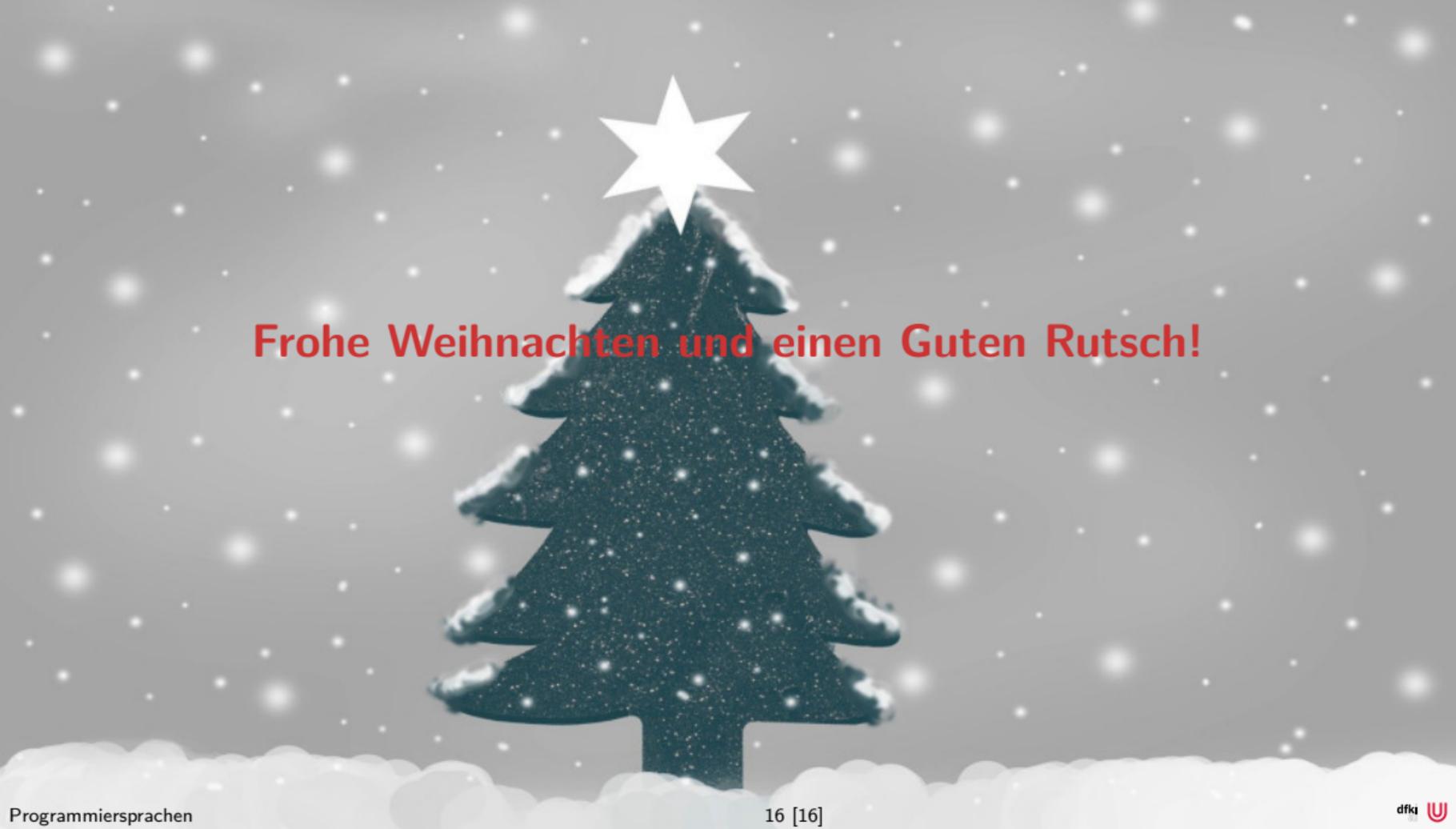
# Größere Programme

## ► Kopieren:

```
>[-]>[-]<<      Initialisierung  
[>+>+<<-]      Verschiebe p[i] nach p[i+1],p[i+2]  
>>[<<+>>-]    Verschiebe p[i+2] nach p[i]  
<<
```

# Zusammenfassung

- ▶ Brainfuck ist:
  - ▶ Turing-vollständig
  - ▶ extrem kompliziert zu benutzen
  - ▶ extrem einfach zu implementieren
  - ▶ in der Praxis unbrauchbar
- ▶ Weitere skurrile Programmiersprachen (siehe <https://esolangs.org/>):
  - ▶ Intercal: eine der ersten dieser Art.
  - ▶ Whitespace: Token sind TAB, LF, Space. Programme sind 'unsichtbar' und können in andere Programmiersprachen (oder Texte) eingebettet werden.



**Frohe Weihnachten und einen Guten Rutsch!**