CARL VON OSSIETZKY universität OLDENBURG

Teil

Imperative Programmierung

Unterrichtseinheit 5

Prozeduren (Hamster-Modell)

Dr. Dietrich Boles

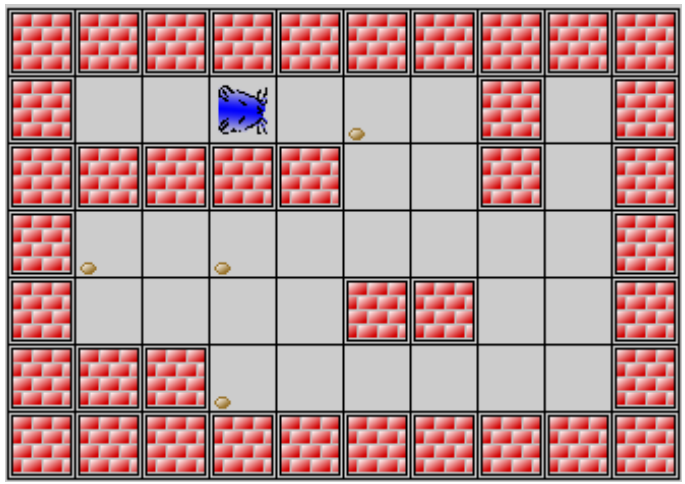
Gliederung

- Motivation
- Prozedurdefinition
- Prozeduraufruf
- Programme mit Prozeduren
- Beispiele
- Vorteile von Prozeduren
- Codekonventionen
- Zusammenfassung

Prozeduren / Motivation

Motivation:

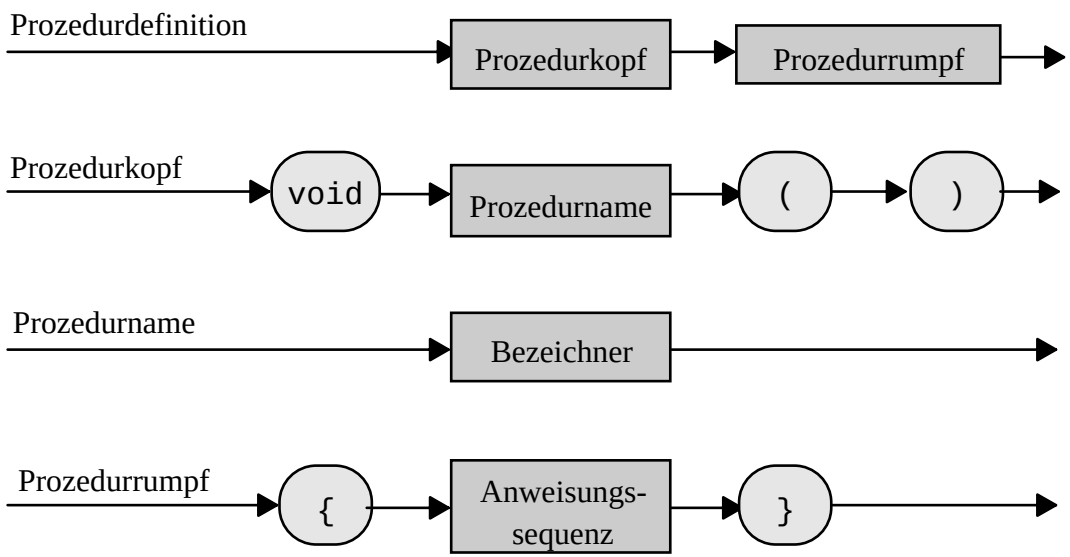
Der Hamster soll zwei Körner einsammeln.



```
void main() {
    vor(); vor();
    nimm();
    linksUm(); linksUm(); linksUm();   <-- rechtsUm();
    vor(); vor();
    linksUm(); linksUm(); linksUm();   <-- rechtsUm();
    vor(); vor();
    nimm();
}
```

Prozedurdefinition: Vereinbarung eines neuen Befehls

Syntax:

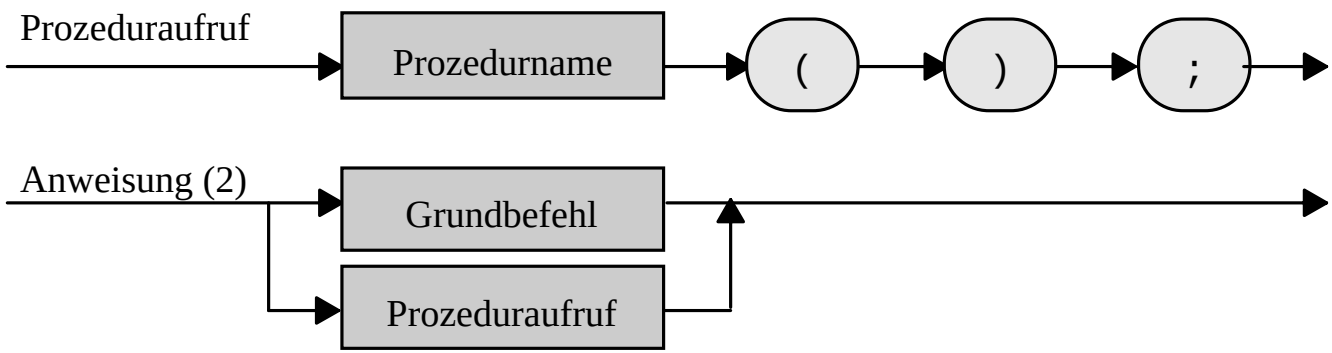


Semantik:

Es wird ein neuer Befehl eingeführt.

Prozeduraufruf: Aufruf eines selbstdefinierten Befehls

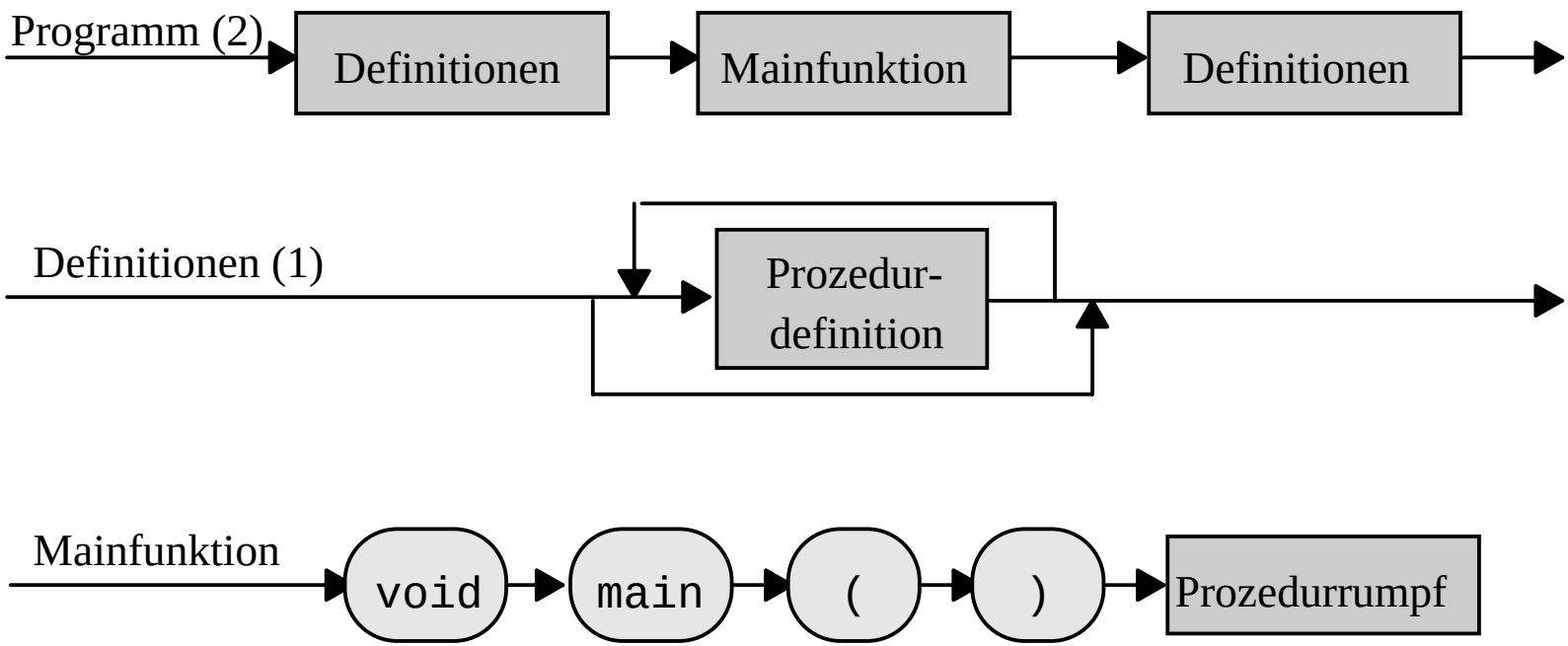
Syntax:



Semantik:

Beim Aufruf einer Prozedur wird die Anweisungssequenz des Prozedurrumpfes ausgeführt.

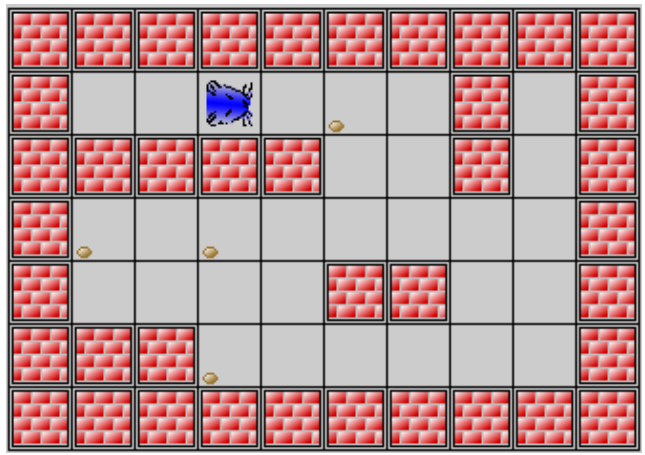
Programm (mit Prozeduren):



Prozeduren / Beispiel 1 (1)

Lösung 1 des Motivationsproblems:

Der Hamster soll zwei Körner einsammeln.



```
void main() {
    vor(); vor();
    nimm();
    rechtsUm();
    vor(); vor();
    rechtsUm();
    vor(); vor();
    nimm();
}
```

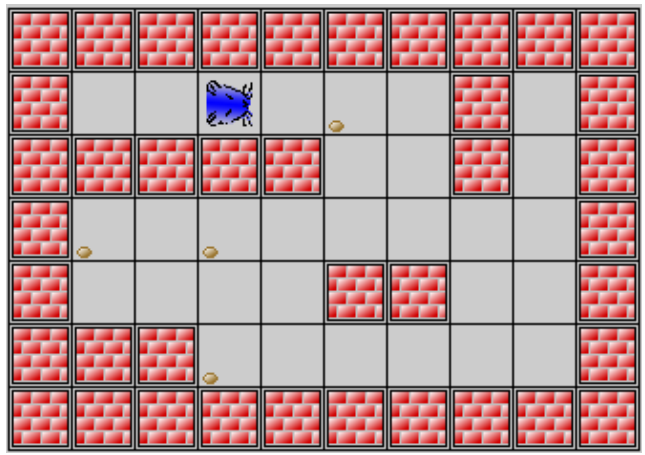
Demo

```
void rechtsUm() {
    linksUm();
    linksUm();
    linksUm();
}
```

Prozeduren / Beispiel 1 (2)

Lösung 2 des Motivationsproblems:

Der Hamster soll zwei Körner einsammeln.



```
void main() {
    zweiVor();
    nimm();
    rechtsUm();
    zweiVor();
    rechtsUm();
    zweiVor();
    nimm();
}

void zweiVor() {
    vor();
    vor();
}
```

```
void rechtsUm() {
    kehrt();
    linksUm();
}

void kehrt() {
    linksUm();
    linksUm();
}
```

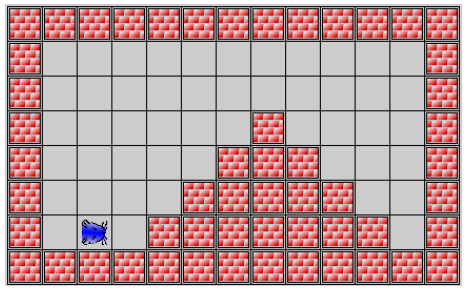
Prozeduren / Beispiel 2

Aufgabe:

Gegeben sei das folgende Territorium.

Der Hamster soll den Berg erklimmen.

Landschaft:



Demo

Programm:

```
void main() {
    laufeZumBerg();
    erklimmeErsteStufe();
    erklimmeZweiteStufe();
    erklimmeDritteStufe();
    erklimmeGipfel();
}

void laufeZumBerg() {
    vor();
}

void erklimmeErsteStufe() {
    erklimmeStufe();
}
```

```
void erklimmeZweiteStufe() {
    erklimmeStufe();
}
void erklimmeDritteStufe() {
    erklimmeStufe();
}
void erklimmeGipfel() {
    erklimmeStufe();
}
void erklimmeStufe() {
    linksUm(); vor();
    linksUm();linksUm();linksUm();
    vor();
}
```

Prozeduren / Vorteile

Vorteile von Prozeduren:

- bessere Übersichtlichkeit von Programmen
- separate Lösung von Teilproblemen
- Platzeinsparung
- einfachere Fehlerbeseitigung
- Flexibilität
- Wiederverwendbarkeit

Codekonventionen

- Prozedurname: Anfangsbuchstabe klein; Anfangsbuchstaben neuer Wortbestandteile groß
- Prozedurname: aussagekräftig !!!!!
- Prozedurkopf und `{` in eine Zeile
- `}` unterhalb des `v` von `void`
- innere Anweisungen um 4 Spalten einrücken
- Hinter Prozedurnamen kein Leerzeichen
- Prozedurdefinition und -aufruf: Zwischen `( )` kein Leerzeichen
- Prozedurdefinition: Hinter `( )` ein Leerzeichen
- Prozeduraufruf: Hinter `( )` kein Leerzeichen
- zwei Prozedurdefinitionen durch Leerzeile trennen

Zusammenfassung

- Mit Hilfe von Prozeduren können neue Befehle definiert werden
- Bei der Prozedurdefinition wird der neue Befehl vereinbart
- Beim Prozeduraufruf wird der neue Befehl ausgeführt