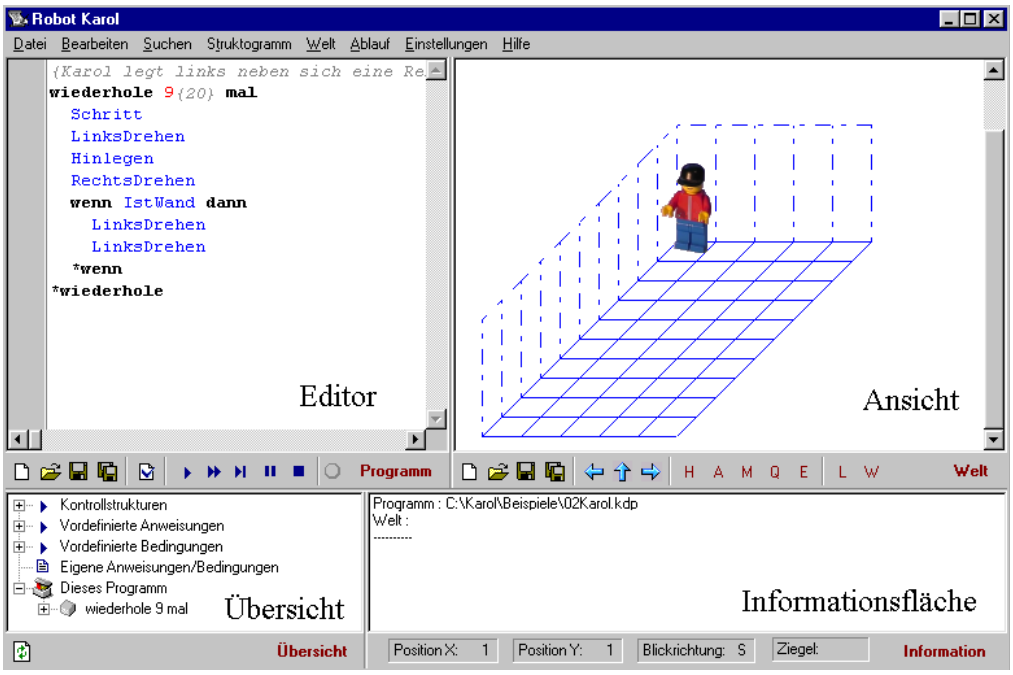
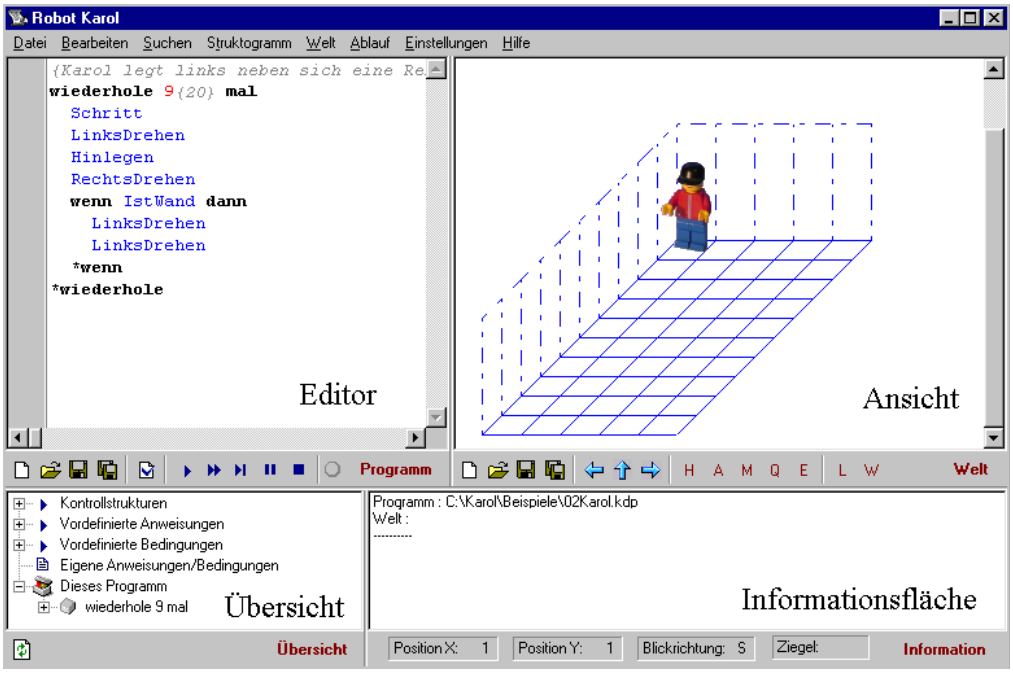
Robot Karol

Eine Programmiersprache
für Schülerinnen und Schüler



Inhaltsverzeichnis

A: Die Oberfläche von Karol



Die Oberfläche von Robot Karol umfasst die vier Teile: Programm, Welt, Übersicht und Informationsfläche.
Die Größe der Bereiche kann durch Ziehen mit der Maus an den Trennlinien eingestellt werden.

Karol kann in der „Welt“ durch Programmanweisungen oder im Direktmodus gesteuert werden. Die direkte Steuerung erfolgt mit Mausklick auf die entsprechenden Schaltflächen oder durch Tasteneingabe.

1: Schaltflächen zur Direktsteuerung

 Linksdrehen
 Schritt
 Rechtsdrehen
H vor sich Ziegel hinlegen
A Ziegel vor sich aufheben
M an der aktuellen Stelle Marke setzen/löschen
Q vor sich Quader aufstellen
E Quader vor sich entfernen
L Welt löschen, alle Ziegel/Quader entfernen
W Welt wiederherstellen, in den Zustand vor dem letzten Programmstart bzw. wenn die Welt schon abgespeichert wurde in den gespeicherten Zustand.Programmierumgebung
2D Welt zweidimensional darstellen (Welt von oben betrachtet)

2: Befehle zum Programmieren von Karol

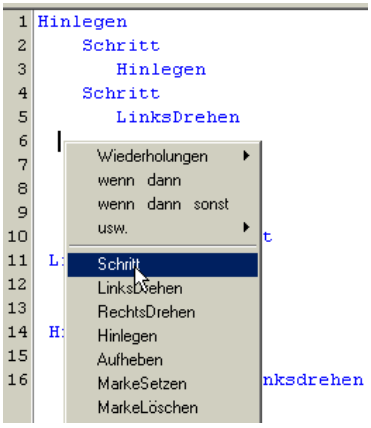
Zum Programmieren von Karol dient der linke Teil des Bildschirmes. Hier schreibt man die Befehle hinein. Zunächst „hört“ Karol zu.

Diese Befehle kennt Karol:	Schritt LinksDrehen RechtsDrehen Hinlegen Aufheben	MarkeSetzen MarkeLöschen Ton Warten Beenden

Hat Karol „beim Zuhören“ einen Befehl „verstanden“, erscheint er in blauer Schrift. Befehle, die Karol nicht erkennt, weil sie z. B. einen Tippfehler enthalten, bleiben grün.

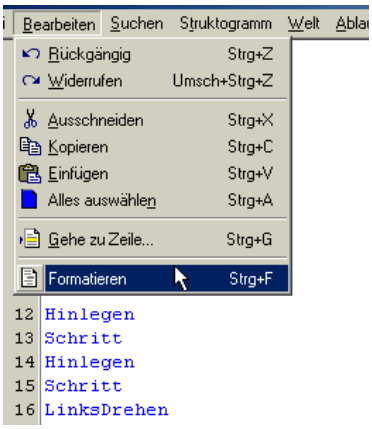
3: Tricks zum schnelleren Programmieren

Befehle mit der Maus einfügen



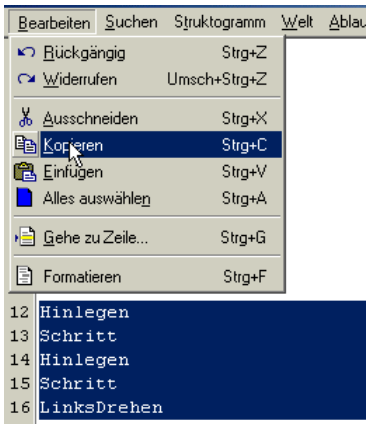
- Mit rechts auf die Stelle klicken, an der der Befehl stehen soll
- Befehl aus dem Kontext-Menü wählen

Befehlsliste formatieren



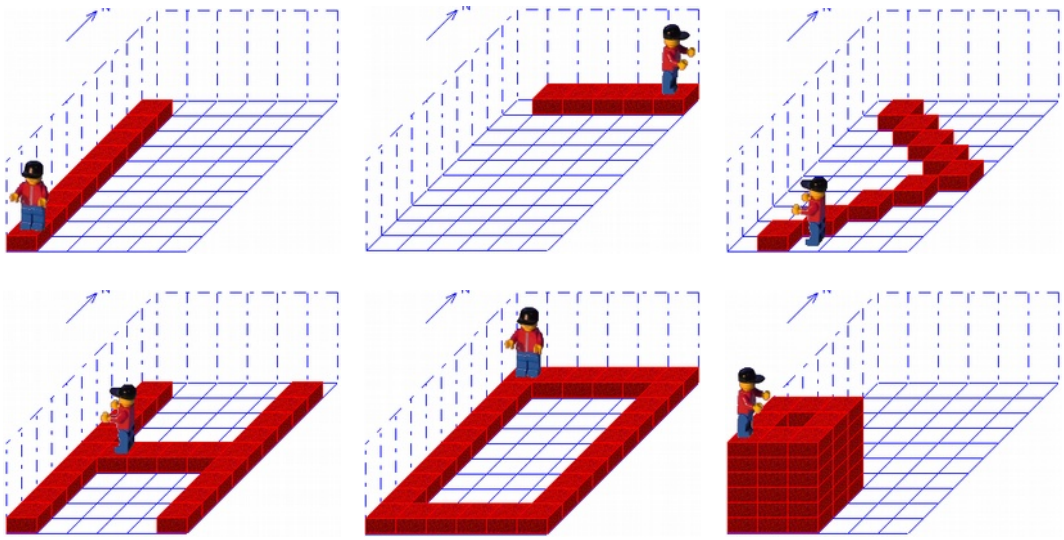
- Menü BEARBEITEN-FORMATIEREN oder Strg + F
- Befehle werden automatisch sauber angeordnet

Befehle kopieren



- Befehle markieren
- Menü BEARBEITEN-FORMATIEREN oder Strg + C
- Cursor neu positionieren
- BEARBEITEN-EINFÜGEN oder STRG + V

4: Vorschläge zum Nachprogrammieren



5: Ein Programm starten

Beim Programmieren von Karol „hört“ Karol nur zu. Soll er die erteilten Befehle ausführen, hat man mehrere Möglichkeiten:

Ausführung der Befehle:
Langsamer Modus

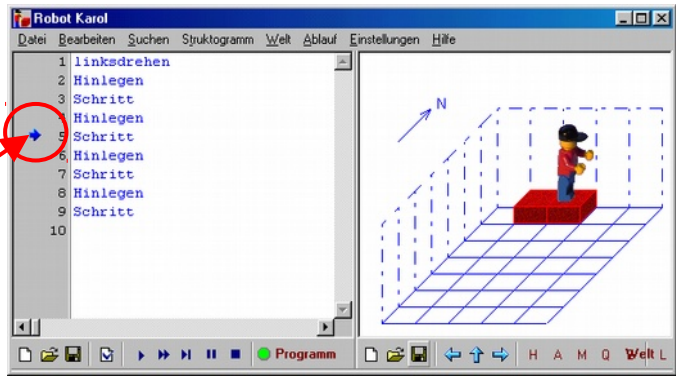
1. Auf ▶ klicken *oder*
2. Menü ABLAUF-START *oder*
3. Funktionstaste F9

Ausführung der Befehle
Schneller Modus

1. Auf ⏩ klicken *oder*
2. Menü ABLAUF-SCHNELLALBAUF *oder*
3. Tastenkombination ⇩ + F9

6: Programmablauf verfolgen

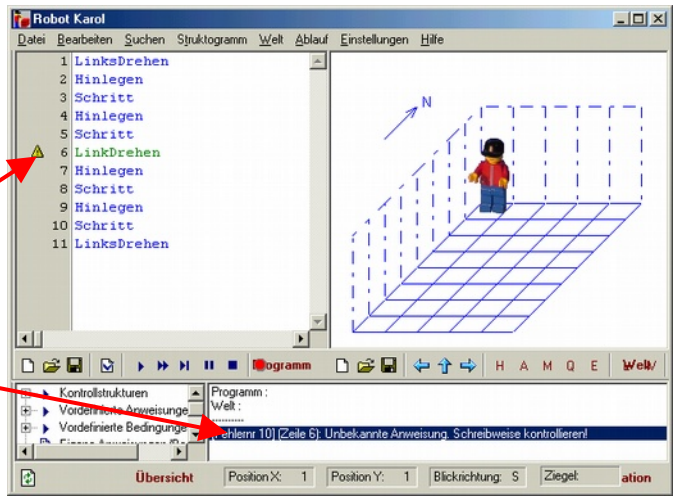
Hat man ein Programm gestartet, kann man im Programmfenster verfolgen, welchen Befehl Karol gerade ausführt. Ein blauer Pfeil ➡ zeigt auf die Zeile mit dem Befehl.



7: Fehlermeldungen verstehen

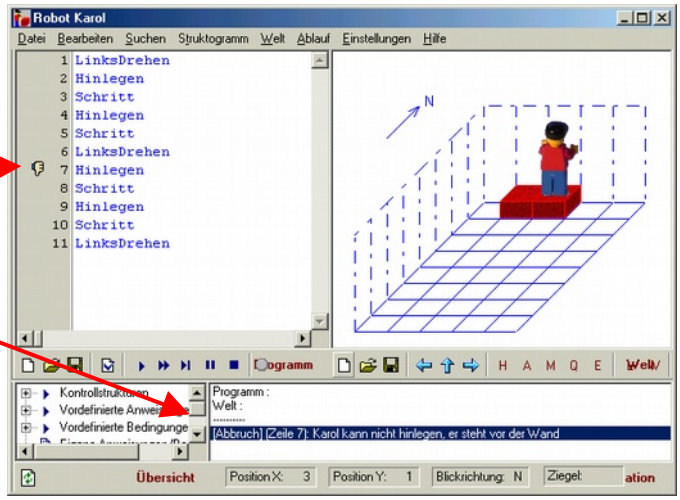
a) <u>Karol hat einen Befehl nicht verstanden</u>

- Links neben der Zeile mit dem falschen Befehl erscheint ein ⚠
- Im Informationsfenster rechts wird man darauf aufmerksam gemacht, dass man vielleicht einen Tippfehler gemacht hat



b) <u>Karol hat den Befehl zwar verstanden, kann ihn aber nicht ausführen</u>

- Links neben der Zeile mit dem Befehl erscheint ein 👎
- Im Informationsfenster rechts unten wird der genauere Grund angegeben



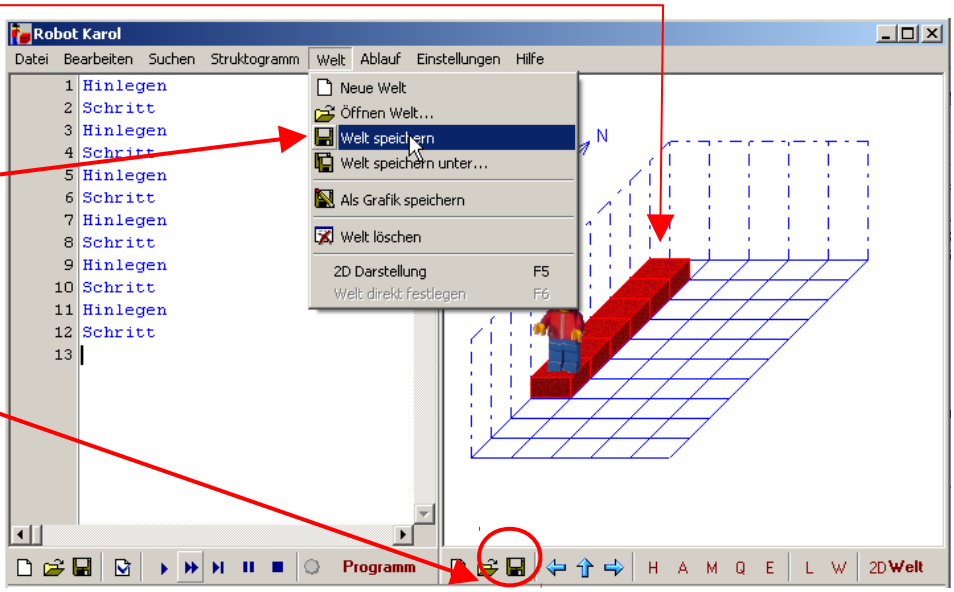
8: Speichern von Welten

Karol kann das speichern, was er gebaut hat.

+ Dazu benutzt man den Menü-Befehl „Welt – Welt speichern“

oder

+ die Schaltfläche „Welt speichern“ im Fenster „Welt“.



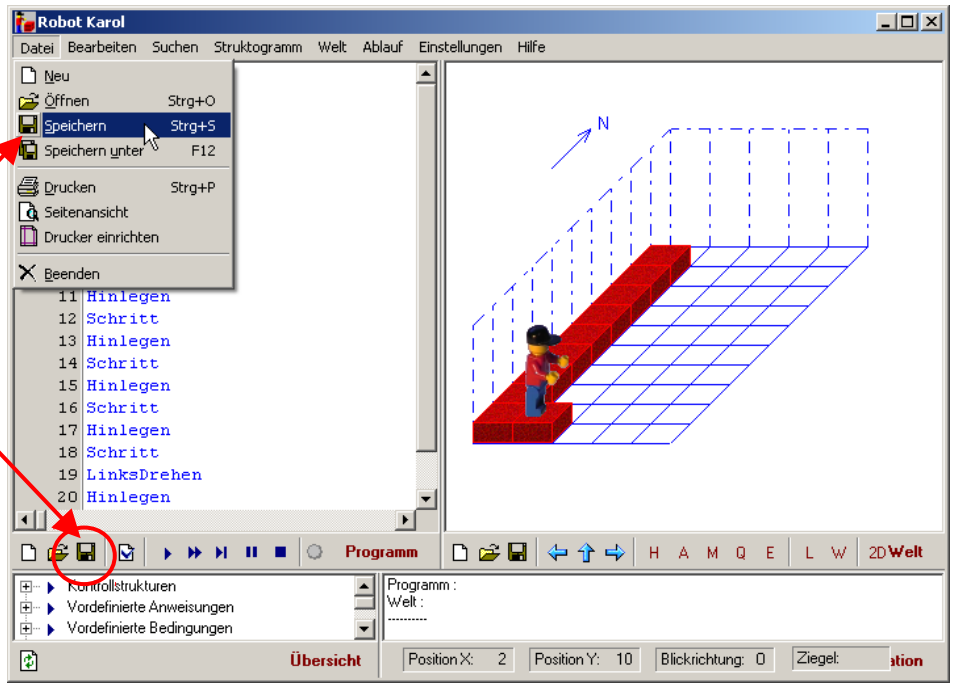
9: Speichern von Programmen

Karol kann auch speichern, wie er etwasgebaut hat.

+ Dazu benutzt man den Menü-Befehl „Datei – Speichern“

oder

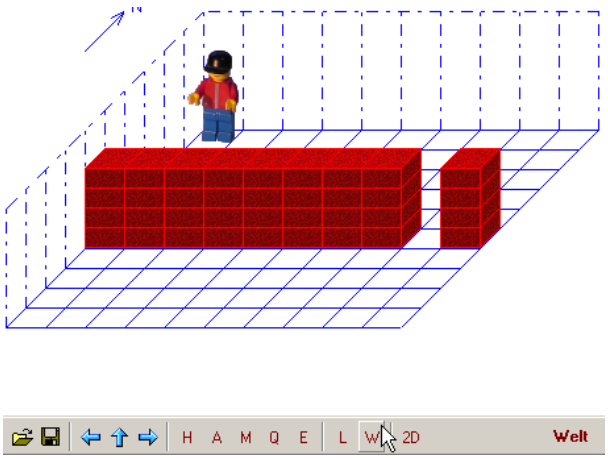
+ die Schaltfläche „Programm speichern“ im Fenster „Welt“.



Meistens ist es wichtiger zu speichern, wie Karol etwas gebaut hat. So kann man jederzeit Karol seine Welt neu bauen lassen.

Nur manchmal benötigt man eine bestimmte Welt, in der Karol ein Problem lösen soll. Dann ist es vorteilhaft, wenn man diese Welt gespeichert hat, damit sie nicht vor jedem Programmstart neu gebaut werden muss.

Ein Beispiel dafür ist die Aufgabe rechts. Hier soll Karol das Loch in der Wand finden. Es wäre mühsam, die Wand jedes Mal neu zu bauen.



10: Eine Welt für Karol einrichten

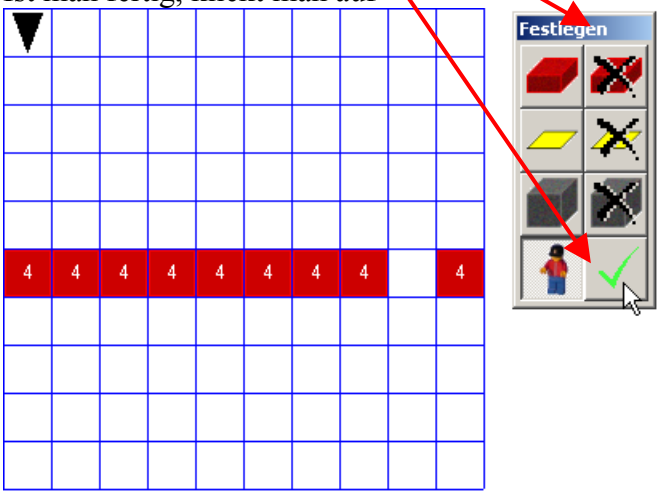
Zunächst wechselt man in die 2D-Darstellung: Menü „Welt- 2D-Darstellung“

o d e r F5

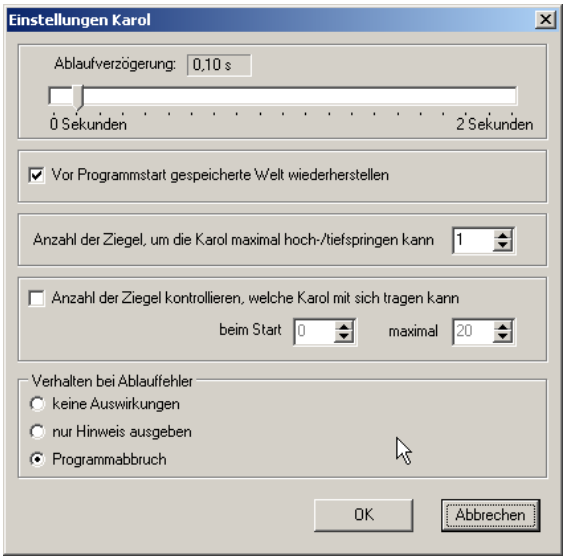
Dann kann man über “Welt direkt festlegen“ oder F6 für Karol eine „Problem-Welt“ selbst bauen.

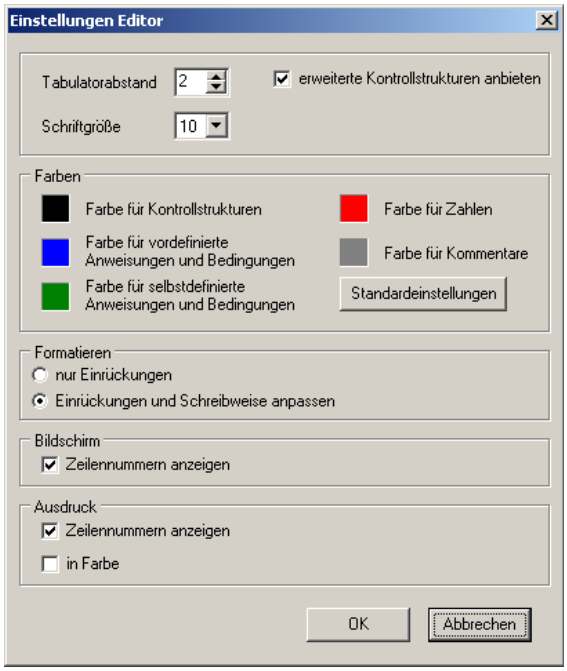
Im Modus „Welt direkt festlegen“ wählt man in einem zusätzlichen Fenster das Werkzeug, das man zum Bauen der Welt braucht.

Ist man fertig, klickt man auf



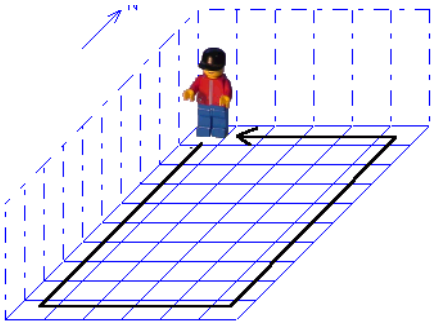
11: Grundeinstellungen





12: Ein Programm vereinfachen: Die gezählte Wiederholung

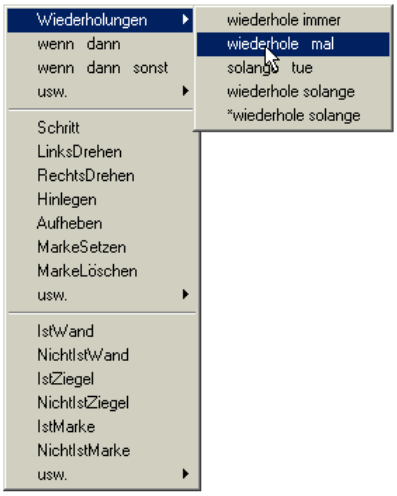
Karols Welt ist 6 x 10 groß. Er soll sie einmal umrunden.



Dazu muss er ...
9 mal einen Schritt machen, sich nach links drehen,
5 mal einen Schritt machen, sich nach links drehen,
9 mal einen Schritt machen, sich nach links drehen,
5 mal einen Schritt machen, sich nach links drehen,

Das sind 32 Befehle, die du ihm geben musst.

Einfacher geht es, wenn du ihm sagst, wie oft er seinen Schritt wiederholen soll.



Im Kontextmenü des Programmierfensters findest du die Befehlsstruktur unter

WIEDERHOLUNGEN – WIEDERHOLE MAL.

Im Programmfenster werden drei Zeilen eingefügt:

```
1 wiederhole | mal
2
3 *wiederhole
```

Der Cursor blinkt an der Stelle, an der du die Zahl der Wiederholungen angibst.

Zwischen die Zeilen WIEDERHOLE .. MAL und *WIEDERHOLE schreibst du, was Karol wiederholen soll

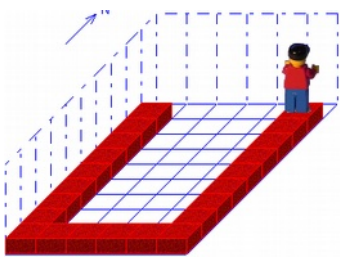
So kommst du mit 16 statt 32 Befehlen aus. Bei längeren Programmieraufgaben kannst du noch viel mehr sparen!

```
 1 wiederhole 9 mal
 2   Schritt
 3 *wiederhole
 4 LinksDrehen
 5 wiederhole 5 mal
 6   Schritt
 7 *wiederhole
 8 LinksDrehen
 9 wiederhole 9 mal
10   Schritt
11 *wiederhole
12 LinksDrehen
13 wiederhole 5 mal
14   Schritt
15 *wiederhole
16 LinksDrehen
```

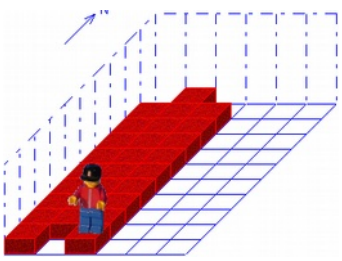
13: Aufgaben zur gezählten Wiederholung

Um die Befehlsliste möglichst kurz zu halten verwende WIEDERHOLE ...

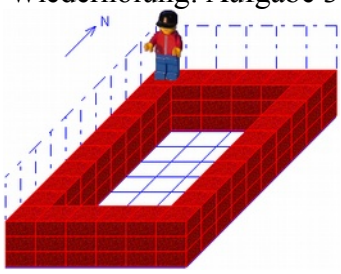
Wiederholung: Aufgabe 1



Wiederholung: Aufgabe 2



Wiederholung: Aufgabe 3

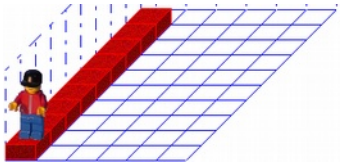


Fach:	Name:	Datum:

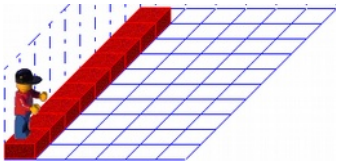
14: Lösungen zum Thema „Gezählte Wiederholung“

Aufgabe 1

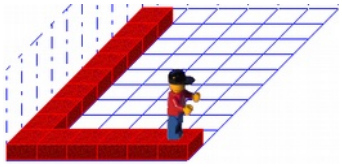
```
wiederhole 9 mal
  hinlegen
  Schritt
*wiederhole
```



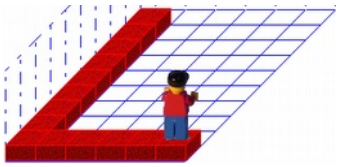
```
linkdrehen
```



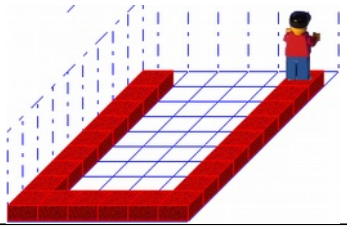
```
wiederhole 5
mal
   hinlegen
   Schritt
*wiederhole
```



```
linksdrehen
```

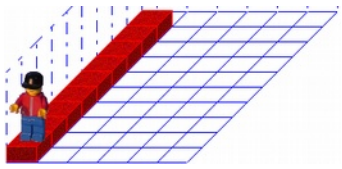


```
wiederhole 9 mal
  hinlegen
Schritt
*wiederhole
```

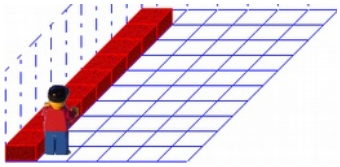


Aufgabe 2

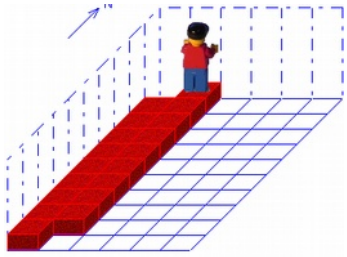
```
wiederhole 9 mal
  hinlegen
  Schritt
*wiederhole
```



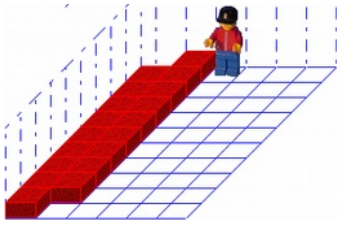
```
linksdrehen
Schritt
linksdrehen
```



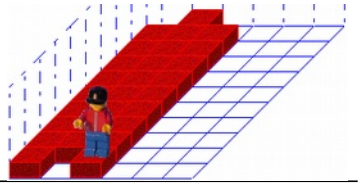
```
wiederhole 9 mal
  hinlegen
  Schritt
*wiederhole
```



```
rechtsdrehen
Schritt
rechtsdrehen
```

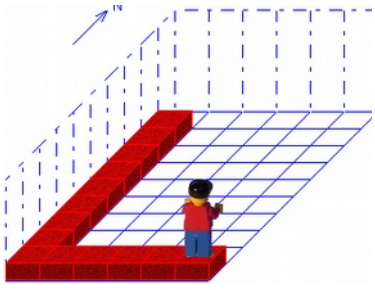


```
wiederhole 9 mal
  hinlegen
  Schritt
*wiederhole
```

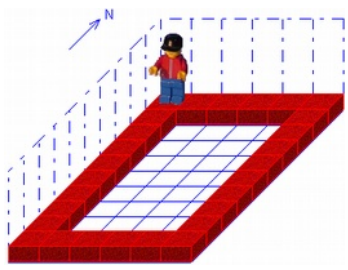


Aufgabe 3

```
wiederhole 9 mal
  Hinlegen
  Schritt
*wiederhole

LinksDrehen

wiederhole 5 mal
  Hinlegen
  Schritt
*wiederhole

LinksDrehen
```



```
wiederhole 2 mal

  wiederhole 9 mal
    Hinlegen
    Schritt
  *wiederhole

  LinksDrehen

  wiederhole 5 mal
    Hinlegen
    Schritt
  *wiederhole

  LinksDrehen

*wiederhole
```

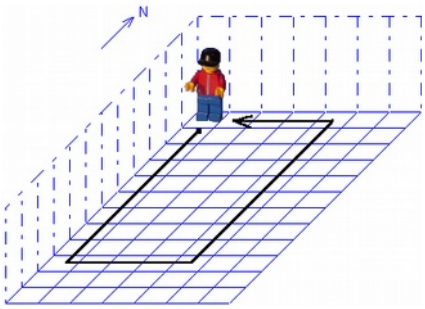


15: Karol entscheidet wie oft er wiederholt: Die bedingte Wiederholung

Karols Welt ist beliebig groß. Er soll sie einmal umrunden.

Problem:
Bei der gezählten Wiederholung ...
- ist seine Runde in einer größeren Welt zu klein
- stößt er in einer kleineren Welt beim Befehl SCHRITT gegen die Wand, weil er sich nicht rechtzeitig nach links dreht

.

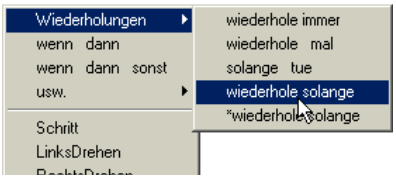


Bei der gezählten Wiederholung reagiert Karol nicht auf die veränderte Größe der Welt.

Lösung:
Karol soll den Befehl SCHRITT wiederholen, solange er nicht vor einer Wand steht.

Im Kontextmenü des Programmierfensters findest du die passende Befehlsstruktur unter

WIEDERHOLUNGEN – WIEDERHOLE SOLANGE.



Lösung: WIEDERHOLE SOLANGE ...

Es werden drei Zeilen im Programmfenster eingefügt:

```
1 wiederhole solange
2
3 *wiederhole
```

Der Cursor blinkt an der Stelle, an der du angibst, was Karol vor jeder Wiederholung überprüfen soll. Hier:

```
1 wiederhole solange NichtIstWand
2   Schritt
3 *wiederhole
```

Zwischen die Zeilen WIEDERHOLE SOLANGE ... und *WIEDERHOLE gibst du Karol den Befehl SCHRITT.
Hat Karol die Wand erreicht, lässt du ihn LINKSDREHEN:

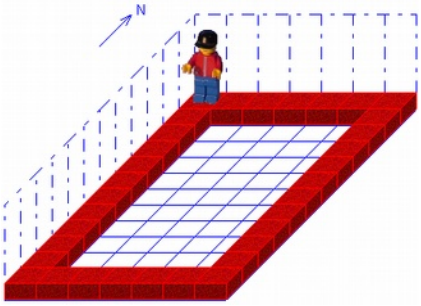
So läuft Karol in jeder beliebigen Welt an den Außenwänden entlang.

```
1  // Westwand entlang
2  wiederhole solange NichtIstWand
3    Schritt
4  *wiederhole
5  LinksDrehen
6  // Südwand entlang
7  wiederhole solange NichtIstWand
8    Schritt
9  *wiederhole
10 LinksDrehen
11 // Ostwand entlang
12 wiederhole solange NichtIstWand
13   Schritt
14 *wiederhole
15 LinksDrehen
16 // Nordwand entlang
17 wiederhole solange NichtIstWand
18   Schritt
19 *wiederhole
20 LinksDrehen
```

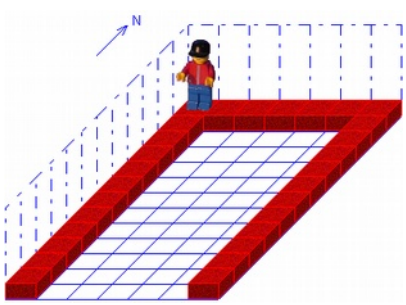
16: Aufgaben zur bedingten Wiederholung

Für beliebige großeWelten verwende WIEDERHOLE SOLANGE ...

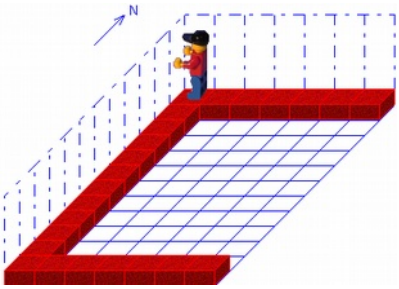
Bedingte Wiederholung: Aufgabe 1



Aufgabe 2



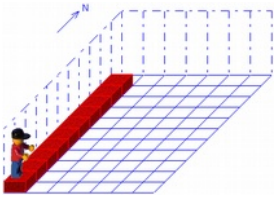
Aufgabe 3



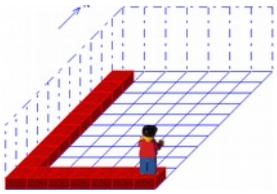
17: Lösungen zum Thema „Bedingte Wiederholung“

Aufgabe 1

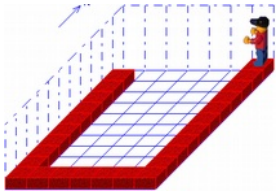
```
// 1. Seite bauen
wiederhole solange NichtIstWand
  Hinlegen
  Schritt
*wiederhole
LinksDrehen
```



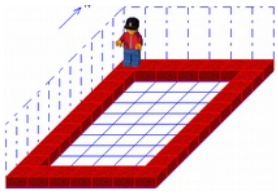
```
// 2. Seite bauen
wiederhole solange NichtIstWand
  Hinlegen
  Schritt
*wiederhole
LinksDrehen
```



```
// 3. Seite bauen
wiederhole solange NichtIstWand
  Hinlegen
  Schritt
*wiederhole
LinksDrehen
```



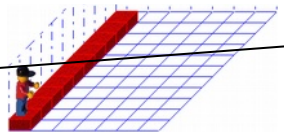
```
// 4. Seite bauen
wiederhole solange NichtIstWand
  Hinlegen
  Schritt
*wiederhole
LinksDrehen
```



Statt die Befehle für die zweite, dritte und vierte Seite extra einzugeben könnte man auch den Bau einer Seite viermal wiederholen lassen.

Aufgabe 2

```
// 1. Seite bauen
wiederhole solange NichtIstWand
  Hinlegen  Schritt
*wiederhole
LinksDrehen
```

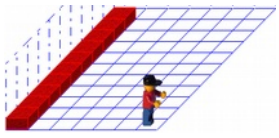


Hier stehen zwei Befehle in einer Zeile.

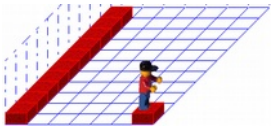
Vorteil:
Das Programm hat so weniger Zeilen.

Nachteil:
Bei einem Fehler in dieser Zeile sagt Karol nur, dass die Zeile einen Fehler enthält, aber nicht, welcher Befehl in dieser Zeile der Schuldige ist.

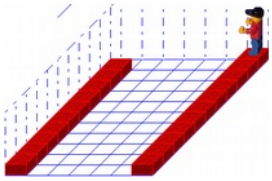
```
// Zur nächsten Wand
wiederhole solange NichtIstWand
  Schritt
*wiederhole
```



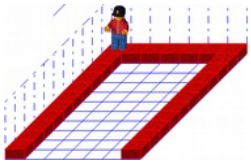
```
// Stein in Ecke setzen
LinksDrehen LinksDrehen
Schritt
LinksDrehen LinksDrehen
Hinlegen
Schritt LinksDrehen
```



```
// Rest bauen
wiederhole solange NichtIstWand
  Hinlegen   Schritt
*wiederhole
LinksDrehen
```



```
wiederhole solange NichtIstWand
  Hinlegen   Schritt
*wiederhole
LinksDrehen
```



18: Karol lagert öfter benötigte Befehlsblöcke aus: Die ANWEISUNG

Problem:
Je umfangreicher die Aufträge für Karol, desto länger wird die Befehlsliste. Oft werden ganze Befehlsblöcke mehrfach benötigt.

Schaut man sich die Befehlsliste rechts an, sieht man, dass Karol zur Wand laufen, ein Quadrat bauen, zur Wand laufen, ein Quadrat bauen ... muss.

Lösung (Grundeinstellung in maximal 10 Ziegel hochsteigen/tief springen geändert):
Wir erstellen eine Anweisung aus den Befehlen für „Zur Wand laufen" und geben ihr den Namen ZURWAND. So brauchen wir im Programm jedes Mal nur den Namen der Anweisung aufrufen. Karol führt dann die Anweisung aus, weil er weiß, was zu tun ist.

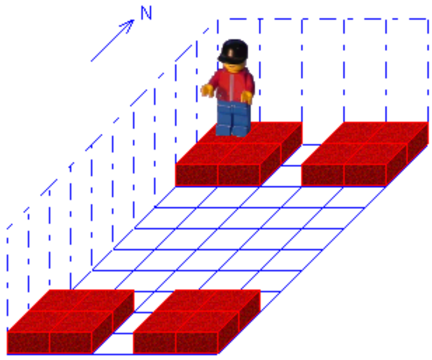
Im Kontextmenü des Programmierfensters findest du die Befehlsstruktur ANWEISUNG unter

1. Im Programmfenster werden drei Zeilen eingefügt. Du gibst der Anweisung in der 1. Zeile einen Namen.
2. Dann Schreibst du zwischen ANWEISUNG ... und *ANWEISUNG, was Karol bei dieser Anweisung alles tun soll.
3. Im Programm selbst reicht es, die Anweisung aufzurufen.

```
1 Anweisung ZurWand                  ← (1)
2   wiederhole solange NichtIstWand  }
3     Schritt                        } ← (2)
4   *wiederhole                      }
5 *Anweisung
6 // Hier beginnt das Programm
7 ZurWand                            ← (3)
```

Wenn du auch noch eine Anweisung für das Quadrat erstellst, wird dein Programm wesentlicher kürzer und übersichtlicher.

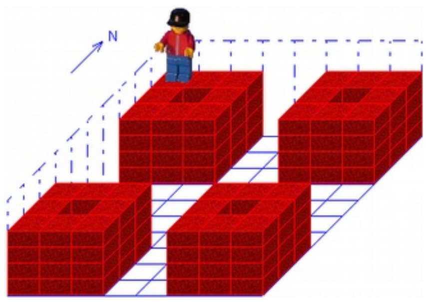
```
Anweisung ZurWand
  wiederhole solange NichtIstWand
    Schritt
  *wiederhole
*Anweisung

Anweisung Quadrat
  wiederhole 4 mal
    LinksDrehen Hinlegen Schritt
  *wiederhole
  linksdrehen
*Anweisung

// Hier beginnt das Programm
wiederhole 4 mal
  ZurWand Quadrat
*wiederhole
```



Karols Aufgabe (oben) und ein Programm dazu (unten)..

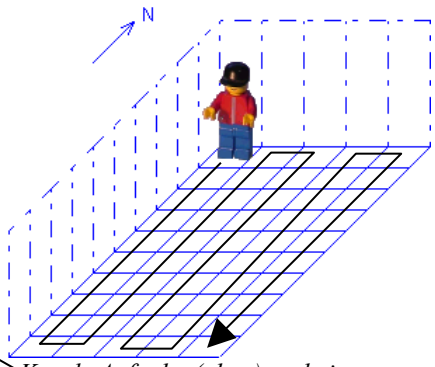
```
// zur Wand laufen
wiederhole solange NichtIstWand
   Schritt
*wiederhole
// Quadrat bauen
wiederhole 4 mal
  LinksDrehen  Hinlegen  Schritt
*wiederhole
LinksDrehen
// zur Wand laufen
wiederhole solange NichtIstWand
   Schritt
*wiederhole
// Quadrat bauen
wiederhole 4 mal
  LinksDrehen  Hinlegen  Schritt
*wiederhole
LinksDrehen
// zur Wand laufen
wiederhole solange NichtIstWand
   Schritt
*wiederhole
// Quadrat bauen
wiederhole 4 mal
  LinksDrehen  Hinlegen  Schritt
*wiederhole
LinksDrehen
// zur Wand laufen
wiederhole solange NichtIstWand
   Schritt
*wiederhole
// Quadrat bauen
wiederhole 4 mal
  LinksDrehen  Hinlegen  Schritt
*wiederhole
LinksDrehen
```

Vorschlag: Erstelle eine Anweisung TURM und verwende sie für das Bauwerk unten



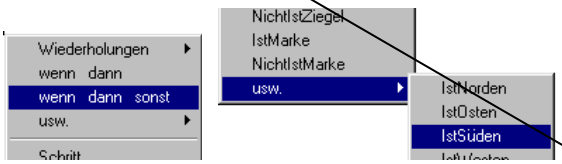
19: Karol entscheidet selbst, was zu tun ist: WENN ... DANN ... SONST ...

Problem:
Karol soll die Bahnen seiner Werkstatt ablaufen. Vor der Südwand soll er sich nach Norden drehen, vor der Nordwand nach Süden. Er soll selbst entscheiden, welche Drehung richtig ist.

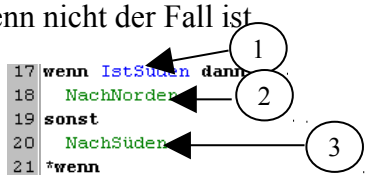


Karols Aufgabe (oben) und ein erstes Programm dazu (unten)..

Lösung:
Wir erstellen die Anweisungen ZURWAND, NACHSÜDEN und NACHNORDEN. Jetzt kann Karel selbst überprüfen, ob er eine Drehung NACHSÜDEN oder NACHNORDEN machen muss.
Im Kontextmenü des Programmierfensters findest du die Befehlsstruktur WENN ... DANN ... SONST ...



1. Hinter WENN gibst du an, was Karol prüfen soll, nämlich WENN IST SÜDEN
2. Dann Schreibst du zwischen WENN ... und SONST die Anweisung(en) dafür, wenn Karol nach Süden schaut.
3. Zwischen SONST und *WENN schreibst du die Anweisung(en) dafür, wenn nicht der Fall ist.



Wenn du auch noch eine Anweisung für das Quadrat erstellst, wird dein Programm wesentlicher kürzer und übersichtlicher.

```
1  Anweisung ZurWand
2    wiederhole solange NichtIstWand
3      Schritt
4    *wiederhole
5  *Anweisung
6
7  Anweisung NachNorden
8    LinksDrehen
9    Schritt
10   LinksDrehen
11 *Anweisung
12
13 Anweisung NachSüden
14   RechtsDrehen
15   Schritt
16   RechtsDrehen
17 *Anweisung
18
19 // *** Programmbeginn ***
20 wiederhole 5 mal
21   ZurWand
22   wenn IstSüden dann
23     NachNorden
24   sonst
25     NachSüden
26   *wenn
27 *wiederhole
```

Lösungsbeispiel, das mit einer Fehlermeldung endet

Testet man das Programm rechts, taucht dieser Fehler auf:

```
 7 Anweisung NachNorden
 8   LinksDrehen
 9   Schritt
10   LinksDrehen
11 *Anweisung
```

[Abbruch] [Zeile 9]: Karol ist an der Wand angestoßen

Bei der 5. Wiederholung will Karol wieder nach Norden, aber er kann keinen Schritt mehr machen, weil er schon vor der Wand steht.

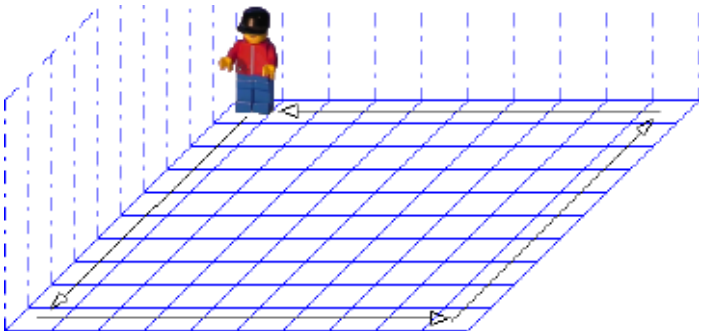
Lösung:
Wir ergänzen die Anweisungen NACHNORDEN und NACHSÜDEN so, dass Karol vor dem Schritt und dem Linksdrehen erst schaut, ob er nicht vor einer Wand steht.

```
 7 Anweisung NachNorden
 8   LinksDrehen
 9   wenn NichtIstWand dann
10     Schritt LinksDrehen
11   *wenn
12 *Anweisung
13
14 Anweisung NachSüden
15   RechtsDrehen
16   wenn NichtIstWand dann
17     Schritt RechtsDrehen
18   *wenn
19 *Anweisung
```

Verbessern der Anweisungen NACHNORDEN und NACHSÜDEN

B: Übungen zu Robot Karol

1: Karol geht rund um die Welt



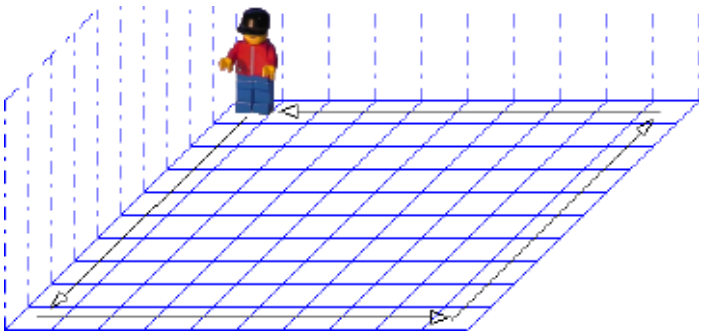
Lösung 1: Viele Wiederholungen

Karol soll einmal um seine Welt laufen.

Welt: 10 x 10
Weltdateien: 01RundUmWelt1.kdw
Lösung: 01RundUmWelt1.kdp

Befehle:	`Schritt, Linksdrehen`		
Bedingungen:	`---`		

Kontrollstrukturen:	`wiederhole x mal`
	`  ...`
	`* wiederhole`
		`Anweisung xyz`
		`  ...`
		`*Anweisung`
			`Verschachtelte Wiederholung`



01RundUmWelt1.kdp

```
wiederhole 9 mal
  Schritt
*wiederhole
LinksDrehen
```

Karol läuft 9 Schritte und dreht dann nach links.

```
wiederhole 9 mal
  Schritt
*wiederhole
LinksDrehen
```

Karol läuft 9 Schritte und dreht dann nach links.

```
wiederhole 9 mal
  Schritt
*wiederhole
LinksDrehen
```

Karol läuft 9 Schritte und dreht dann nach links.

```
wiederhole 9 mal
  Schritt
*wiederhole
LinksDrehen
```

Karol läuft 9 Schritte und dreht dann nach links.

Lösung 2: Wiederholungen wiederholt

01RundUmWelt2.kdp

```
wiederhole 4 mal
  wiederhole 9 mal
    Schritt
  *wiederhole
  LinksDrehen
*wiederhole
```

Trick: Man kann in eine Wiederholung eine andere Wiederholung einbauen.

Karol wiederholt 4mal
die 9malige Wiederholung und den Linksdreh.

Lösung 3: Anweisungen erklären

01RundUmWelt3.kdp

```
// --- Anweisungen -------
Anweisung ZurWand
  wiederhole 9 mal
    Schritt
  *wiederhole
  LinksDrehen
*Anweisung
// === Hauptprogram ======
wiederhole 4 mal
  ZurWand
*wiederhole
```

Karol bekommt in einer Anweisung erklärt, wie er zur Wand geht und nach links dreht.

Im Hauptprogramm braucht man deswegen nicht alle Befehle neu schreiben. Es reicht, wenn man den Namen der Anweisung angibt, die Karol ausführen soll.

Struktogramm und Befehlsliste 1

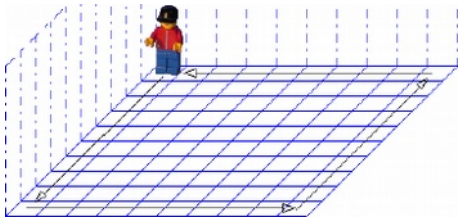
01RundUmWelt3.kdp

Lösung mit vier aufeinander folgende Wiederholungen

Hauptprogramm

wiederhole 9 **mal**
Schritt
LinksDrehen
wiederhole 9 **mal**
Schritt
LinksDrehen
wiederhole 9 **mal**
Schritt
LinksDrehen
wiederhole 9 **mal**
Schritt
LinksDrehen

```
wiederhole 9 mal
  Schritt
*wiederhole
LinksDrehen

wiederhole 9 mal
  Schritt
*wiederhole
LinksDrehen

wiederhole 9 mal
  Schritt
*wiederhole
LinksDrehen

wiederhole 9 mal
  Schritt
*wiederhole
LinksDrehen
```



01RundUmWelt3.kdp

Lösung mit einer Wiederholung, die viermal wiederholt wird (geschachtelte Wiederholung)

Hauptprogramm

wiederhole 4 **mal**	
	wiederhole 9 **mal**
		Schritt
	LinksDrehen

```
wiederhole 4 mal
  wiederhole 9 mal
    Schritt
  *wiederhole
  LinksDrehen
*wiederhole
```

01RundUmWelt3.kdp

Lösung mit einer Anweisung, die viermal wiederholt wird

Hauptprogramm

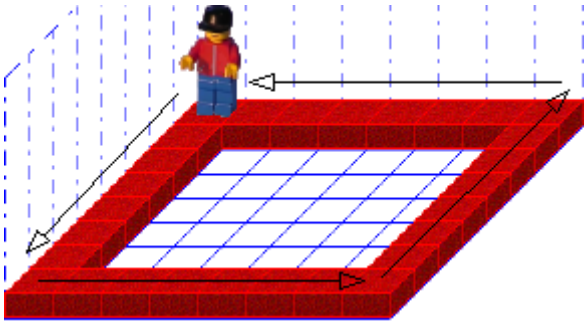
wiederhole 4 **mal**
ZURWAND

Anw.: ZURWAND

wiederhole 9 **mal**
Schritt
LinksDrehen

```
// --- Anweisungen -------
Anweisung ZurWand
  wiederhole 9 mal
    Schritt
  *wiederhole
  LinksDrehen
*Anweisung


// === Hauptprogram ======
wiederhole 4 mal
  ZurWand
*wiederhole
```

2: Karol legt Ziegel beim Gang rund um die Welt



Aufgabe 1: Funktioniert nicht immer

Karol soll einmal um seine Welt laufen und dabei Ziegel auf den Boden legen.

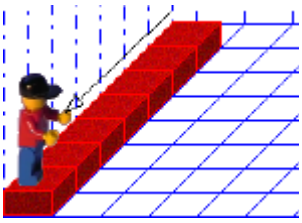
Welt: 8 x 8
Weltdateien: ---
Lösung: 02ZiegelRunde1.kdp, 02ZiegelRunde2.kdp, 02ZiegelRundgang3.kdp

Befehle: `Schritt, Linksdrehen, Hinlegen`

Bedingungen: `NichtIstWand`

Kontrollstrukturen:

```
wiederhole x mal        solange ... tue        Anweisung xyz
    ...                     ...                    ...
* wiederhole            *solange               *Anweisung
```



Karol legt sieben Ziegel zur Wand und dreht nach links.

// Verwendete Anweisung

```
Anweisung ZiegelZurWand
  wiederhole 7 mal
    Hinlegen
    Schritt
  *wiederhole
  LinksDrehen
*Anweisung
```

// Hauptprogramm

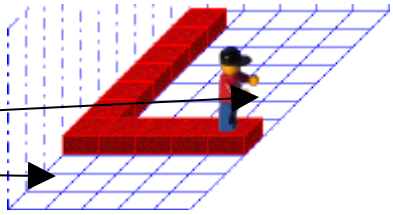
```
wiederhole 4 mal
  ZiegelZurWand
*wiederhole
```

Bei der Anweisung *ZiegelZurWand* bekommt Karol erklärt, dass er 7mal einen Ziegel vor sich hinlegen und einen Schritt darauf machen soll.
Danach soll er sich nach links drehen

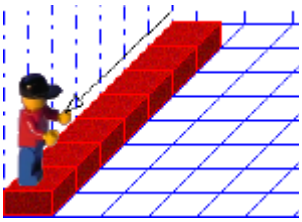
Für einen Ziegelring muss Karol 4mal *ZiegelZurWand* legen.

Aufgabe 2: Funktioniert in jeder Welt

Ändert sich die Größe von Karols Welt, funktioniert das Programm ZiegelRundgang1.kdp nicht mehr richtig. Entweder verunglückt Karol, weil er gegen eine Wand rennt
oder der Ziegelring wird für die Welt zu klein.



Lösung



Solange keine Wand kommt, legt Karol Ziegel. Dann dreht er nach links.

// Verwendete Anweisung

```
Anweisung ZiegelBisWandIst
  solange NichtIstWand tue
    Hinlegen
    Schritt
  * solange
  LinksDrehen
*Anweisung
```

// Hauptprogramm

```
wiederhole 4 mal
  ZiegelBisWandIst
*wiederhole
```

Bei der Anweisung `ZiegelBisWandIst` bekommt Karol erklärt, dass er, solange keine Wand kommt, einen Ziegel vor sich hinlegen und einen Schritt darauf machen soll.
Danach soll er sich nach links drehen

Für einen Ziegelring muss Karol 4mal `ZiegelBisWandIst` legen.

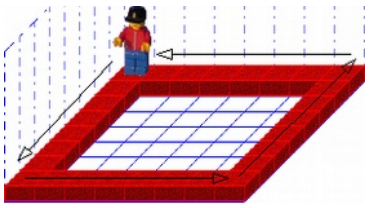
Aufgabe 3: Anweisungen erleichtern kompliziertere Aufgaben

Karol soll fünf Ziegelringe übereinander legen

Struktogramm und Befehlsliste 2

02ZiegelRundgang1.kdp

Welt ist 8 x 8 Ziegel
Lösung mit einer Anweisung, die viermal wiederholt wird



Hauptprogramm

wiederhole 4 **mal**
ZIEGELZURWAND

Anw.: ZIEGELZURWAND

wiederhole 7 **mal**
Hinlegen
Schritt
LinksDrehen

```
// --- Anweisungen -------
Anweisung ZiegelZurWand
  wiederhole 7 mal
    Hinlegen
    Schritt
  *wiederhole
  LinksDrehen
*Anweisung
// === Hauptprogram ======
wiederhole 4 mal
  ZiegelZurWand
*wiederhole
```

02ZiegelRundgang2.kdp

Welt ist beliebig groß
Lösung mit einer Anweisung, die viermal wiederholt wird

Hauptprogramm

wiederhole 4 **mal**
ZIEGELBISWANDIST

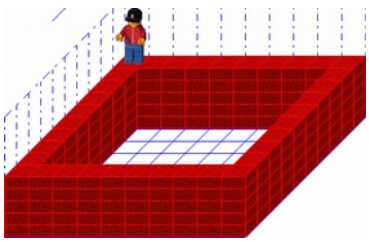
Anw.: ZIEGELBISWAND

solange NichtIstWand
Hinlegen
Schritt
LinksDrehen

```
// --- Anweisungen -------
Anweisung ZiegelBisWandIst
  solange NichtIstWand tue
    Hinlegen
    Schritt
  *solange
  LinksDrehen
*Anweisung

// === Hauptprogram ======
wiederhole 4 mal
  ZiegelBisWandIst
*wiederhole
```

02ZiegelRundgang3.kdp

Welt ist beliebig groß, 5 Ziegelringe übereinander
Lösung mit einer Wiederholung, die eine Anweisung aufruft und fünfmal wiederholt wird (geschachtelte Wiederholung)



Hauptprogramm

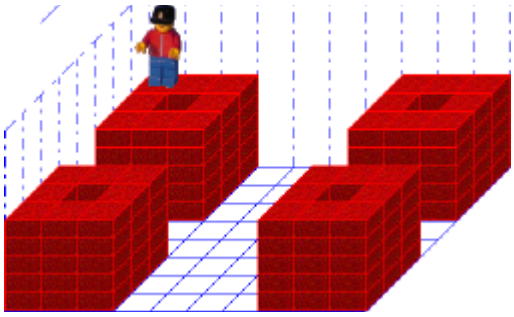
wiederhole 5 **mal**	
	wiederhole 4 **mal**
	ZIEGELBISWANDIST

Anw.: ZIEGELBISWANDIST

solange NichtIstWand
Hinlegen
Schritt
LinksDrehen

```
// --- Anweisungen -------
Anweisung ZiegelBisWandIst
  solange NichtIstWand tue
    Hinlegen
    Schritt
  *wiederhole
  LinksDrehen
*Anweisung

// === Hauptprogram ======
wiederhole 5 mal
  wiederhole 4 mal
    ZiegelBisWandIst
  *wiederhole
*wiederhole
```

3: Karol baut 4 Türme in die 4 Ecken der Welt (schwindelfrei!)



Aufgabe 1: Mehrere Anweisungen verwenden

Karol soll in jede der 4 Ecken seiner Welt einen Turm bauen. Der Turm soll 3 Ziegel lang, 3 Ziegel breit und 5 Ziegel hoch sein. Karol soll mindestens 5 Ziegel herab springen können (Einstellungen – Karol).

Welt: mindestens 7 lang, 7 breit und 5 hoch
Weltdateien: ---
Lösung: 03Türme1.kdp, 03Türme2.kdp

Befehle: `Schritt, Linksdrehen,Hinlegen`

Bedingungen: `NichtIstWand`

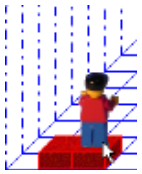
Kontrollstrukturen:

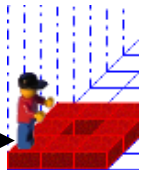
```
wiederhole x mal        solange ... tue        Anweisung xyz
    ...                     ...                    ...
* wiederhole            *solange               *Anweisung
```

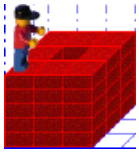
// Verwendete Anweisung

```
Anweisung GeheZurWand
  solange NichtIstWand tue
    Schritt
  *solange
  LinksDrehen
*Anweisung
```

Bei der Anweisung GeheZurWand bekommt Karol erklärt, dass er, solange keine Wand kommt, einen Schritt darauf machen soll.
Danach soll er sich nach links drehen


Hinlegen
Schritt
Hinlegen
Schritt
LinksDrehen


4mal wiederholt


20mal wiederholt ergibt 5 Ziegel Höhe

```
Anweisung
  wiederhole 20 mal
    Hinlegen Schritt
    Hinlegen Schritt
    LinksDrehen
  *wiederhole
*Anweisung
```

Bei der Anweisung BaueTurm bekommt Karol erklärt, wie er am besten einen Turm baut

// Hauptprogramm

```
wiederhole 4 mal
  GeheZurWand
  BaueTurm
*wiederhole
```

Für vier Türme in den Ecken muss Karol 4mal zur Wand gehen und dann einen Turm bauen.

Aufgabe 2: Weitere Anweisungen einfügen

Karol soll auf jeden Turm noch Zinnen setzen. Dazu erstellen wir einfach eine neue Anweisung zum Bauen der Zinnen und ergänzen das Hauptprogramm.

// Neue Anweisung

```
Anweisung BaueZinnen
  wiederhole 4 mal
    Schritt Hinlegen
Schritt
    LinksDrehen
```

Neue Anweisung zum Bauen der Turmzinnen

Ergänztes Hauptprogramm

// Hauptprogramm

```
wiederhole 4 mal
  GeheZurWand
  BaueTurm
  BaueZinnen
*wiederhole
```

```
  *wiederhole
*Anweisung
```

Struktogramm und Befehlsliste 3

03Türme1.kdp

Welt ist mindestens 7 Ziegel lang und breit, Karol kann 10 Ziegel tief hüpfen
Lösung mit zwei Anweisungen, die viermal wiederholt werden

Hauptprogramm

wiederhole 4 **mal**
GEHEZURWAND
BAUETURM

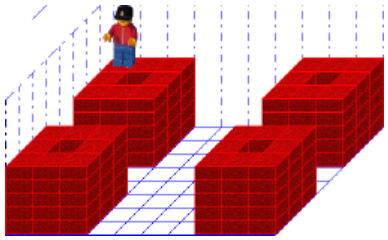
Anw.: GEHEZURWAN

solange NichtIstWand
Schritt
LinksDrehen

Anw.: BAUETURM

wiederhole 20 **mal**
Hinlegen
Schritt
Hinlegen
Schritt
LinksDrehen

```
// --- Anweisungen -------
Anweisung GeheZurWand
  solange NichtIstWand tue
    Schritt
  *solange
  LinksDrehen
*Anweisung
// ----------------------
Anweisung BaueTurm
  wiederhole 20 mal
    Hinlegen Schritt
    Hinlegen Schritt
    LinksDrehen
  *wiederhole
*Anweisung

// === Hauptprogram ======
wiederhole 4 mal
  GeheZurWand
  BaueTurm
*wiederhole
```



03Türme2.kdp

Welt ist mindestens 7 Ziegel lang und breit
Lösung mit drei Anweisungen, die viermal wiederholt werden

Hauptprogramm

wiederhole 4 **mal**
GEHEZURWAND
BAUETURM
BAUEZINNEN

Anw.: GEHEZURWAN

solange NichtIstWand
Schritt
LinksDrehen

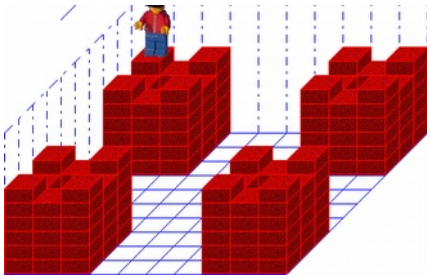
Anw.: BAUETURM

wiederhole 20 **mal**
Hinlegen
Schritt
Hinlegen
Schritt
LinksDrehen

Anw.: BAUEZINNEN

wiederhole 4 **mal**
Schritt
Hinlegen
Schritt
LinksDrehen

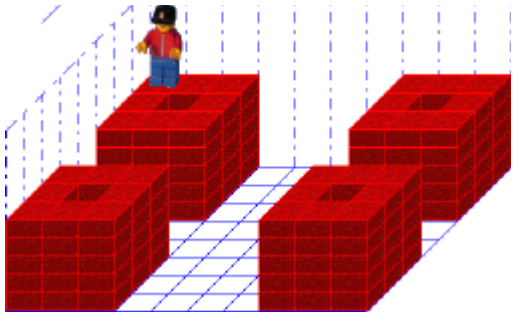
```
// --- Anweisungen -------
Anweisung GeheZurWand
  solange NichtIstWand tue
    Schritt
  *solange
  LinksDrehen
*Anweisung
// ----------------------
Anweisung BaueTurm
  wiederhole 20 mal
    Hinlegen Schritt
    Hinlegen Schritt
    LinksDrehen
  *wiederhole
*Anweisung
// ----------------------
Anweisung BaueZinnen
  wiederhole 4 mal
    Schritt Hinlegen Schritt
    LinksDrehen
  *wiederhole
*Anweisung
// === Hauptprogram ======
wiederhole 4 mal
  GeheZurWand
  BaueTurm
  BaueZinnen
*wiederhole
```



Fach:	Name:	Datum:

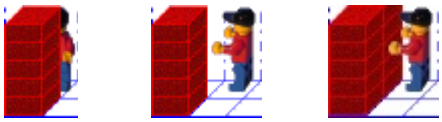
4: Karol baut 4 Türme in die 4 Ecken (nicht schwindelfrei!)

Karol soll in jede der 4 Ecken seiner Welt einen Turm bauen. Der Turm soll 3 Ziegel lang, 3 Ziegel breit und 5 Ziegel hoch sein. Kann Karol nur einen Ziegel herab springen, schafft er zwar den ersten Turm, traut sich aber dann nicht mehr von dieser Höhe herunter zu springen.

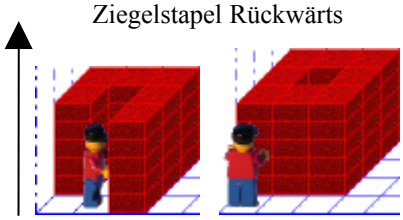


Welt: mindestens 7 lang, 7 breit und 5 hoch
Weltdateien: ---
Lösung: 04Türme.kdp

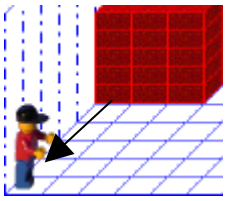
Befehle:	`Schritt, Linksdrehen, Rechtsdrehen, Hinlegen`
Bedingungen:	`NichtIstWand`
Kontrollstrukturen:	`wiederhole x mal ... * wiederhole` `solange ... tue ... *solange` `Anweisung xyz ... *Anweisung`



Ziegelstapel Rechtsdrehen Rückwärts
Ziegelstapel Rückwärts



3mal wiederholt ... und fertig gestellt



// Verwendete Anweisung

```
Anweisung ZiegelStapel
  wiederhole 5 mal
    Hinlegen
  *wiederhole
*Anweisung
```

Karel baut einen 5 Ziegel-hohen Stapel.

```
Anweisung BaueTurm
  wiederhole 3 mal
    ZiegelStapel
    RechtsDrehen
    Rückwärts
    Ziegelstapel
    Rückwärts
  *wiederhole
  Ziegelstapel
  LinksDrehen
  Rückwärts
  Ziegelstapel
*Anweisung
```

Für einen Turm baut Karel dreimal einen Ziegelstapel, geht nach rechts und einen Schritt zurück und baut einen weiteren Ziegelstapel

Dann baut er den Turm fertig.

```
Anweisung GeheZurWand
  solange NichtIstWand tue
    Schritt
  *solange
  LinksDrehen
*Anweisung
```

Solange keine Wand kommt, geht Karol nach vorne.
Dann dreht er nach links.

```
Anweisung Rückwärts
  LinksDrehen LinksDrehen
  Schritt
  RechtsDrehen RechtsDrehen
*Anweisung
```

Bei der Anweisung `Rückwärts` bekommt Karol erklärt, wie er einen Schritt zurück geht

// Hauptprogramm

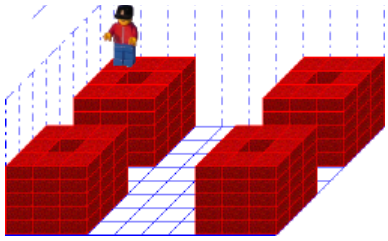
```
wiederhole 4 mal
  BaueTurm
  LinksDrehen LinksDrehen
  GehezurWand
*wiederhole
```

Für vier Türme in den Ecken muss Karol 4mal zur Wand gehen und dann einen Turm bauen.

Struktogramm und Befehlsliste 4

04Türme.kdp

Welt ist mindestens 7 Ziegel lang und breit, Karol kann **nur 1 Ziegel tief** hüpfen
Lösung mit vier Anweisungen, die viermal wiederholt werden



Hauptprogramm

wiederhole 4 **mal**
BAUETURM
LinksDrehen
LinksDrehen
GEHEZURWAND

Anw.: ZIEGELSTAPEL

Schnell
wiederhole 5 **mal**
Hinlegen

Anw.: RÜCKWÄRTS

Schnell
LinksDrehen
LinksDrehen
Schritt
RechtsDrehen
RechtsDrehen

Anw.: BAUETURM

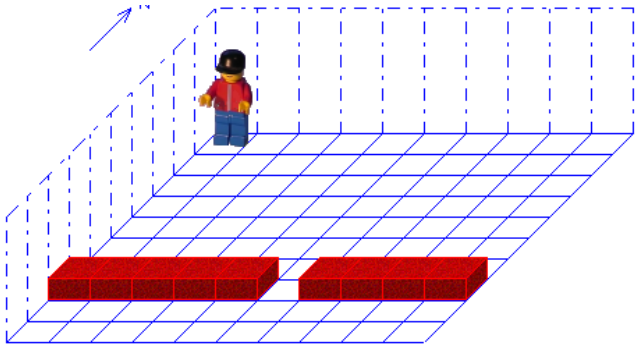
wiederhole 3 **mal**
ZIEGELSTAPEL
RechtsDrehen
RÜCKWÄRTS
ZIEGELSTAPEL
RÜCKWÄRTS
ZIEGELSTAPEL
LinksDrehen
RÜCKWÄRTS
ZIEGELSTAPEL

Anw.: GEHEZURWAN

solange NichtIstWand
Schritt
LinksDrehen

```
// --- Anweisungen -------
Anweisung ZiegelStapel
  Schnell
  wiederhole 5 mal
    Hinlegen
  *wiederhole
*Anweisung

Anweisung Rückwärts
  Schnell
  LinksDrehen LinksDrehen
  Schritt
  RechtsDrehen RechtsDrehen
*Anweisung

Anweisung BaueTurm
  wiederhole 3 mal
    ZiegelStapel
    RechtsDrehen
    Rückwärts
    Ziegelstapel
    Rückwärts
  *wiederhole
  Ziegelstapel
  LinksDrehen
  Rückwärts
  Ziegelstapel
*Anweisung

Anweisung GeheZurWand
  solange NichtIstWand tue
    Schritt
  *solange
  LinksDrehen
*Anweisung

// === Hauptprogram ======
wiederhole 4 mal
  BaueTurm
  LinksDrehen LinksDrehen
  GehezurWand
*wiederhole
```

5: Karol soll durch ein Loch in einer Quermauer zur Südwand laufen

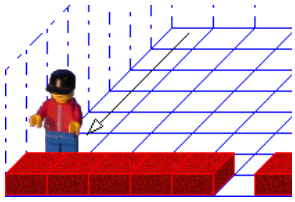


Quer zu Karols Weg von der Nord- zur Süd-Wand ist eine Ziegelmauer aufgebaut. Gott-Sei-Dank hat sie ein Loch. Dieses soll Karol finden und bis zur Südwand laufen.

Welt: Größe beliebig
Weltdateien: wand.kdw
Lösung: 05wand.kdp

Befehle:	`Schritt, Linksdrehen, Rechtsdrehen`
Bedingungen:	`IstZiegel, NichtIstZiegel, NichtIstWand`

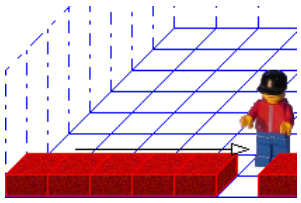
Kontrollstrukturen:	`Solang ... tue`
	`...`
	`*solange`



// 1. Karol läuft zur Ziegelmauer

```
solange NichtIstZiegel tue
  Schritt
*solange
```

Karol läuft bis zur Ziegelmauer

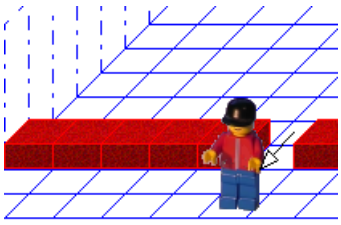


// 2. Er läuft die Ziegelmauer entlang und sucht ein Loch

```
solange IstZiegel tue
    LinksDrehen
    Schritt
    RechtsDrehen
*solange
```

Solange Karol vor einem Ziegel steht, läuft er einen Schritt weiter an der Mauer entlang.

// 3. Er läuft bis zur Südwand



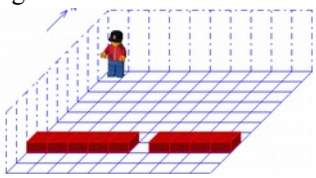
```
solange nichtIstWand tue
    Schritt
*solange
```

Er läuft durch das Loch in der Ziegelmauer bis zur gegenüberliegenden Wand.

Struktogramm und Befehlsliste 5

05wand.kdp

Welt beliebig
Lösung mit drei aufeinander folgenden solange-tue-Wiederholungen



Hauptprogramm

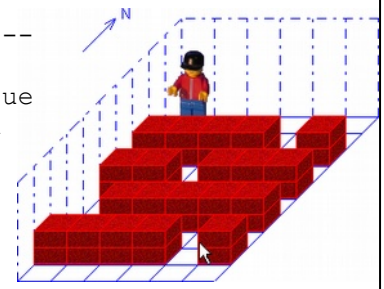
solange NichtIstZiegel
Schritt
solange IstZiegel
LinksDrehen
Schritt
RechtsDrehen
solange NichtIstWand
Schritt

```
// 1. Zur Ziegelmauer laufen
solange nichtIstZiegel tue
  schritt
  *solange
// 2. Ein Loch in der Mauer suchen
solange IstZiegel tue
 linksdrehen
 Schritt
 rechtsdrehen
 *solange
// 3. Weiter zur Südwand laufen
solange NichtIstWand tue
  schritt
*solange
```

Struktogramm und Befehlsliste 6

06wände.kdp

Welt beliebig
Lösung mit einer solange-tue-Wiederholung, die zwei Anweisungen aufruft



Hauptprogramm

solange NichtIstWand
ZURZIEGELMAUER
LOCHFINDEN

Anw.: ZURZIEGELMAUER

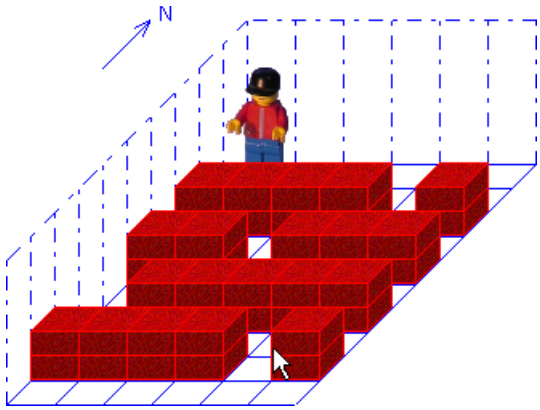
solange NichtIstZiegel	
NichtIstWand	
w	**f**
Schritt	Beenden

Anw.: LOCHFINDEN

RechtsDrehen
solange NichtIstWand
Schritt
LinksDrehen
solange IstZiegel
LinksDrehen
Schritt
RechtsDrehen

```
// --- Anweisungen ---------
Anweisung ZurZiegelmauer
  solange NichtIstZiegel tue
    wenn NichtIstWand dann
      Schritt
    sonst
      Beenden
    *wenn
  *solange
*Anweisung
// -------------------------
Anweisung LochFinden
  RechtsDrehen
  solange NichtIstWand  tue
    Schritt
  *solange
  LinksDrehen
  solange IstZiegel tue
    LinksDrehen
    Schritt
    RechtsDrehen
  *solange
*Anweisung

// === Hauptrprogramm =======
solange NichtIstWand tue
  ZurZiegelmauer
  Lochfinden
*solange
```

6: Karol soll durch die Löcher in den Mauern zur Südwand laufen

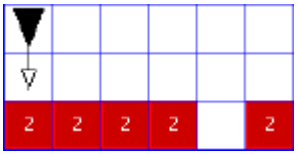


Aufgabe 1: Verschachtelte Strukturen

Quer zu Karols Weg von der Nord- zur Süd-Wand sind Ziegelmauern aufgebaut.Gott-Sei-Dank haben sie Löcher. Diese soll Karol finden und bis zur Südwand laufen.

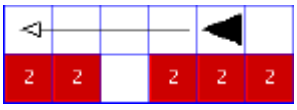
Welt: Größe beliebig
Weltdateien: 06wände1.kdw, 06wände2.kdw
Lösung: 06wände1.kdp, 06wände2.kdp

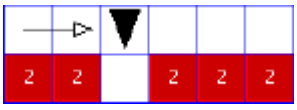
Befehle:	`Schritt, Linksdrehen, Rechtsdrehen, Beenden`
Bedingungen:	`IstZiegel, NichtIstZiegel, NichtIstWand`
Kontrollstrukturen	`Solang ... tue` / `...` / `*solange` — `wenn ... dann` / `...` / `*wenn` — `Anweisung ...` / `...` / `*Anweisung` — `Verschachtelt:` `solange ...` `wenn ...`



```
// --- Anweisungen -------
Anweisung ZurZiegelmauer
  solange NichtIstZiegel tue
    wenn NichtIstWand dann
      Schritt
    sonst
      Beenden
    *wenn
  *solange
*Anweisung
```

Öfter verwendete Anweisungen:
Diese Anweisung wird immer dann gebraucht, wenn Karol wieder ein Stück „freie Fahrt“ hat.
Er geht weiter nach Süden, bis er auf eine Mauer stößt.
Damit das Programm ohne Fehler endet, marschiert er nur nach Süden, wenn er nicht schon vor der Südwand steht


Zur Westwand


Loch suchen

```
Anweisung LochFinden
  RechtsDrehen
  solange NichtIstWand  tue
    Schritt
  *solange
  LinksDrehen
  solange IstZiegel tue
    LinksDrehen
    Schritt
    RechtsDrehen
  *solange
*Anweisung
```

Mit dieser Anweisung sucht Karol das Loch in der Mauer.
Doch zunächst geht er bis zur Außenwand im Westen.

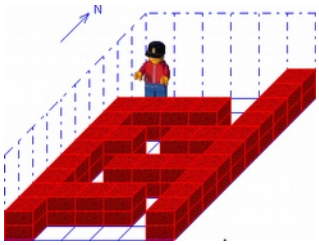
Dann dreht er zur Ziegelmauer.

Solange er jetzt vor einem Ziegel steht, macht er folgendes:
Linksum, Schritt, rechtsum

Steht er nicht mehr vor einem Ziegel, hat er das Loch gefunden.

Das eigentliche Programm:
Solange Karol nicht die Südwand erreicht hat, geht er zur nächsten Ziegelwand (ZurZiegelwand) und sucht dann das Loch in ihr (Loch-Finden)

```
// === Hauptprogramm =====
solange NichtIstWand tue
  ZurZiegelmauer
  Lochfinden
*solange
```

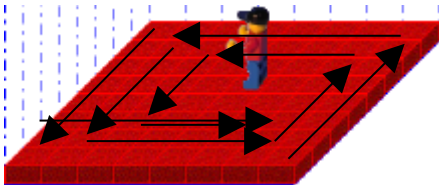


Aufgabe 2: Weitere Übung

Auch an der Ost- und Westwand sind Ziegelmauern.

7: Karol fliest den Boden

Aufgabe 1: Spiral-Lösung für 8x8-Welt



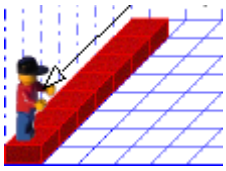
Karol soll in einer Welt von 8 x 8 Ziegeln den Boden mit Ziegeln fliesen. Er will dabei spiralförmig vorgehen.
Welt: 8 x 8
Weltdateien: 07fliesen1.kdw, ---
Lösung: 07fliesen1.kdp, 07fliesen2.kdp

Befehle: `Schritt, Linksdrehen, Hinlegen`

Bedingungen: `NichtIstWand`

Kontrollstrukturen:

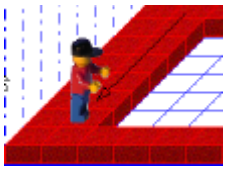
```
wiederhole x mal      solange ... tue      Anweisung xyz
    ...                   ...                 ...
* wiederhole          *solange             *Anweisung
```



// Verwendete Anweisung

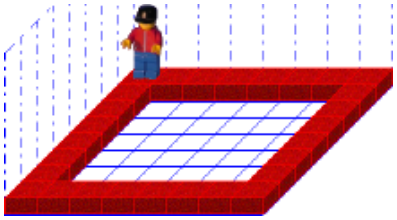
```
Anweisung FliesenAussen
  solange NichtIstWand tue
    Hinlegen Schritt
  *solange
  LinksDrehen
*Anweisung
```

Bei der Anweisung `FliesenAussen` bekommt Karol erklärt, wie er die Fliesen an einer Außenwand verlegt.



```
Anweisung FliesenInnen
  solange NichtIstZiegel tue
    Hinlegen Schritt
  *solange
  LinksDrehen
*Anweisung
```

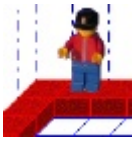
Bei der Anweisung `FliesenInnen` bekommt Karol erklärt, wie er die Fliesen auf einer Innenbahn verlegt.



// Hauptprogramm

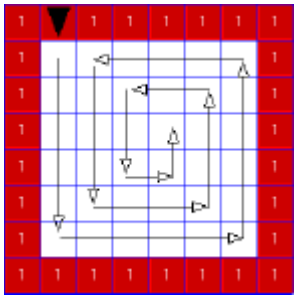
```
wiederhole 4 mal
  FliesenAussen
*wiederhole
```

Zuerst fliest Karol die Außenbahnen an den Wänden.



```
LinksDrehen
Schritt
RechtsDrehen
```

Dann bringt er sich in die richtige Position zum Weitermachen



```
wiederhole 11 mal
  FliesenInnen
*wiederhole
```

Er weiß, dass 11 Innenbahnen zu fließen sind. Deswegen befolgt er 11mal die Anweisung `FliesenInnen`.

Aufgabe 2: Spiral-Lösung für beliebige Welten

Das Fliesen des Bodens funktioniert nur in einer Welt mit 8 Ziegeln Breite. In einer anderen Welt sind es mehr oder weniger Fliesenbahnen innen.

// Hauptprogramm (alt)

```
...
...
wiederhole 11 mal
  FliesenInnen
*wiederhole
```

Beim Verlegen der Fliesenbahnen innen hört Karel nicht nach 11 Bahnen auf, sondern fliest solange, bis kein Feld mehr frei ist.

// Hauptprogramm (neu)

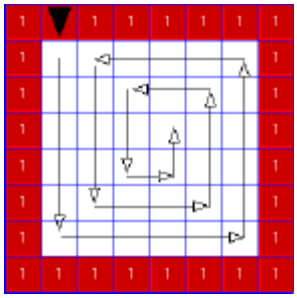
```
...
...
solange NichtIstZiegel tue
  FliesenInnen
*solange
```

Fach:	Name:	Datum:

Struktogramm und Befehlsliste 7

07Fliesen1.kdp

Welt 8 x 8 Ziegel
Lösung mit zwei Anweisungen und zwei aufeinander folgenden gezählten Wiederholungen



Hauptprogramm

wiederhole 4 **mal**
FLIESENAUSSEN
LinksDrehen
Schritt
RechtsDrehen
wiederhole 11 **mal**
FLIESENINNEN

Anw.: FLIESENAUSSE

solange NichtIstWand
Hinlegen
Schritt
LinksDrehen

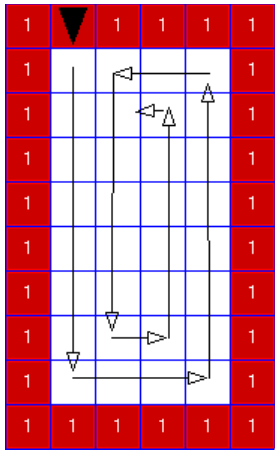
Anw.: FLIESENINNEN

solange NichtIstZiegel
Hinlegen
Schritt
LinksDrehen

```
// --- Anweisungen --------
Anweisung FliesenAussen
  solange NichtIstWand tue
    Hinlegen Schritt
  *solange
  LinksDrehen
*Anweisung
// ------------------------
Anweisung FliesenInnen
  solange NichtIstZiegel tue
    Hinlegen Schritt
  *solange
  LinksDrehen
*Anweisung

// === Hauptprogramm =======
wiederhole 4 mal
  FliesenAussen
*wiederhole
LinksDrehen
Schritt
RechtsDrehen
wiederhole 11 mal
  FliesenInnen
*wiederhole
```

07Fliesen2.kdp

Welt beliebig
Lösung mit zwei Anweisungen und einer gezählten Wiederholungen und einer solange-tue-Wiederholung



Hauptprogramm

wiederhole 4 **mal**
FLIESENAUSSEN
LinksDrehen
Schritt
RechtsDrehen
solange NichtIstZiegel
FLIESENINNEN

Anw.: FLIESENAUSSE

solange NichtIstWand
Hinlegen
Schritt
LinksDrehen

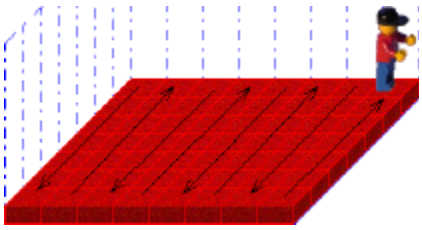
Anw.: FLIESENINNEN

solange NichtIstZiegel
Hinlegen
Schritt
LinksDrehen

```
// --- Anweisungen --------
Anweisung FliesenAussen
  solange NichtIstWand tue
    Hinlegen Schritt
  *solange
  LinksDrehen
*Anweisung
// ------------------------
Anweisung FliesenInnen
  solange NichtIstZiegel tue
    Hinlegen Schritt
  *solange
  LinksDrehen
*Anweisung

// === Hauptprogramm =======
wiederhole 4 mal
  FliesenAussen
*wiederhole
LinksDrehen
Schritt
RechtsDrehen
solange NichtIstZiegel tue
  FliesenInnen
*solange
```

Fach:	Name:	Datum:

8: Karol fliest den Boden mit einem Kompass



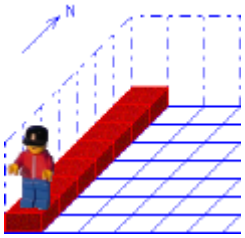
Aufgabe 1: Lösung mit Kompass

Karol soll in einer beliebigen Welt den Boden mit Ziegeln fliesen.
Er will dabei bahnenweise vorgehen.

Welt: beliebig
Weltdateien: ---
Lösung: 08fliesen1.kdp, 08fliesen2.kdp

Befehle:	`Schritt, Linksdrehen, Rechtsdrehen, Hinlegen`
Bedingungen:	`NichtIstWand, IstSüden`

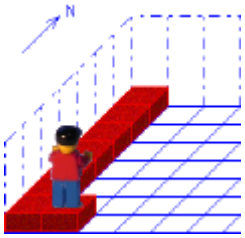
Kontrollstrukturen:	`wiederhole x mal  solange ... tue  Anweisung xyz  Verschatetl:`
	`...  ...  ...  wenn ...`
	`* wiederhole  *solange  *Anweisung  wenn ...`



// Verwendete Anweisung

```
Anweisung FliesenBahn
  solange NichtIstWand tue
    Hinlegen
    Schritt
  *solange
*Anweisung
```

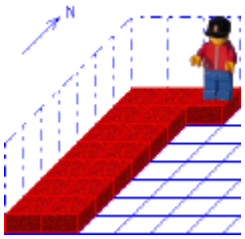
Solange keine Wand kommt, legt Karol einen Ziegel vor sich hin und macht einen Schritt auf die Fliese



```
Anweisung Umkehren
  wenn IstSüden dann
    LinksDrehen
    wenn NichtIstWand dann
      Hinlegen
      Schritt
      LinksDrehen
    *wenn
  sonst
    RechtsDrehen
    wenn NichtIstWand dann
      Hinlegen
      Schritt
      RechtsDrehen
    *wenn
  *wenn
*Anweisung
```

Wenn K. nach Süden schaut, dann dreht er nach links. Wenn keine Wand vor ihm ist, dann legt er einen Ziegel, macht einen Schritt und dreht noch einmal nach links.

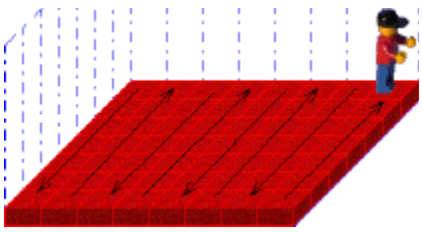
Sonst schaut er nach Norden um macht statt der Links- jetzt Rechtsdrehungen.



// Hauptprogramm

```
// Startfliese
Schritt
linksdrehen linksdrehen
hinlegen Schritt
Rechtsdrehen Rechtsdrehen
// Rest
solange NichtIstWand tue
  FliesenBahn
  Umkehren
*solange
```

Karol legt in die NW-Ecke links oben einen Ziegel.



Solange er nicht vor einer Wand steht, fliest er eine Bahn und kehrt um und macht mit der nächsten Bahn weiter.

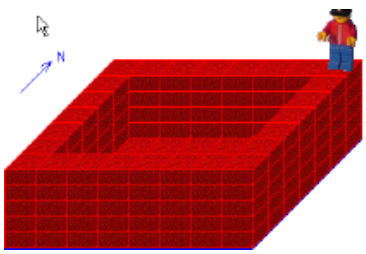
Aufgabe 2: Schwimmbadbau mit Kompass

Karol baut ein Schwimmbad. Dazu fliest er zuerst den Boden und baut dann die Wände.

// Hauptprogramm (ergänzt)

```
...
Rechtsdrehen
wiederhole 16 mal
  Fliesenbahn Rechtsdrehen
*wiederhole
```

Wenn der Boden fertig ist, dreht Karol nach rechts und baut vier Außenwände mit 4 Ziegeln Höhe.



Struktogramm und Befehlsliste 8

08Fliesen1.kdp

Welt beliebig
Lösung mit zwei Anweisungen und einer solange-tue-Wiederholung, nachdem in der NW-Ecke die Startfliese gelegt ist

Hauptprogramm

Schritt
LinksDrehen
LinksDrehen
Hinlegen
Schritt
RechtsDrehen
RechtsDrehen
solange NichtIstWand
FLIESENBAHN
UMKEHREN

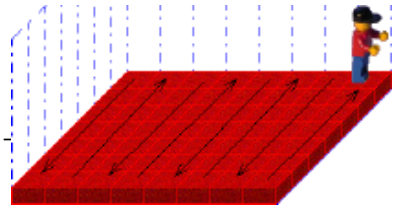
Anw.: FLIESENBAHN

solange NichtIstWand
Hinlegen
Schritt

Anw.: UMKEHREN

IstSüden	
w	**f**
LinksDrehen	RechtsDrehen
NichtIstWand	**NichtIstWand**
w / **f**	**w** / **f**
Hinlegen	Hinlegen
Schritt	Schritt
LinksDrehen	RechtsDrehen

```
// --- Anweisungen --------
Anweisung FliesenBahn
  solange NichtIstWand tue
    Hinlegen Schritt
  *solange
*Anweisung
// ------------------------
Anweisung Umkehren
  wenn IstSüden dann
    LinksDrehen
    wenn NichtIstWand dann
      Hinlegen
      Schritt
      LinksDrehen
    *wenn
  sonst
    RechtsDrehen
    wenn NichtIstWand dann
      Hinlegen
      Schritt
      RechtsDrehen
    *wenn
  *wenn
*Anweisung
// === Hauptprogramm =======
// Startfliese
Schritt
linksdrehen linksdrehen
hinlegen Schritt
Rechtsdrehen Rechtsdrehen
// Restlicher Boden
solange NichtIstWand tue
  FliesenBahn
  Umkehren
*solange
```

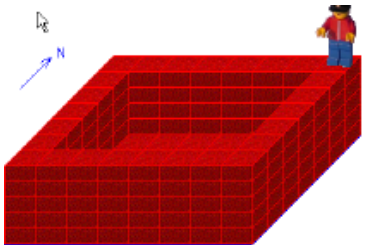


08Fliesen2.kdp

Welt beliebig
Befehlsliste von oben. Das Hauptprogramm wird um diese Anweisungen ergänzt

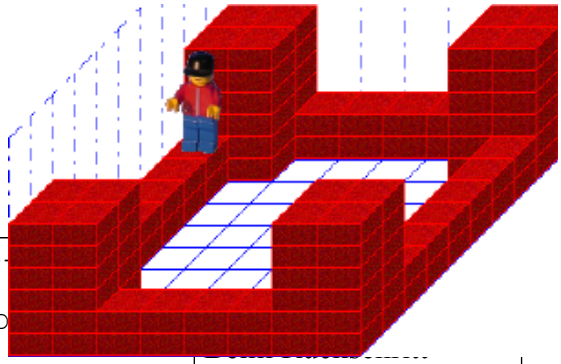
RechtsDrehen	
IstWand	
w	**f**
LinksDrehen	RechtsDrehen
LinksDrehen	
wiederhole 16 **mal**	
FLIESENBAHN	
LinksDrehen	

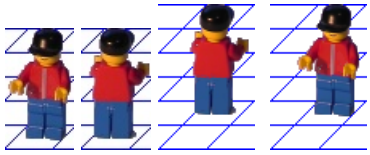
```
// Erst Himmelsrichtung feststellen und drehen
Rechtsdrehen
Wenn IstWand dann
    linksdrehen linksdrehen
sonst
  rechtsdrehen
*wenn
// Wände bauen
wiederhole 16 mal
  Fliesenbahn linksdrehen
*wiederhole
```



9: Karol baut eine Mauer mit 4 massiven Ecktürmen

Aufgabe: Karol kann **nur 1 Ziegel tief** hüpfen
Er soll eine zweilagige Mauer mit vier Ecktürmen bauen.
Welt: mindestens 5 Ziegel lang und breit
Weltdateien: ---
Lösung 09EcktürmeBauen.kdp

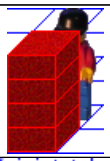




Umdrehen Schritt Umdrehen

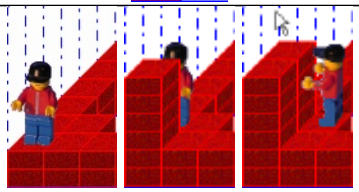
```
// --- Anweisungen ---
Anweisung Umdrehen
  LinksDrehen LinksD
*Anweisung
// -------------------------
Anweisung RückSchritt
  Umdrehen Schritt Umdrehen
*Anweisung
```

macht er einen Schritt rückwärts.

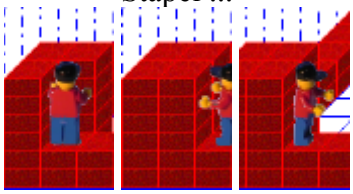


```
Anweisung Stapel
  wiederhole 4 mal
    Hinlegen
  *wiederhole
*Anweisung
```

Hier legt er vier Ziegel aufeinander.



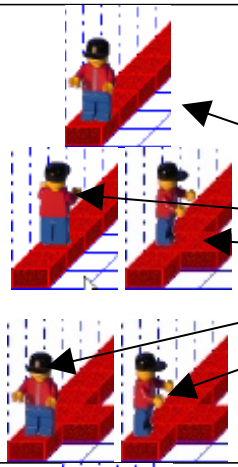
Zur Wand – Rückschritt - Stapel ...



```
Anweisung Turmbauen
  solange NichtIstWand tue
    Schritt
  *solange
  RückSchritt stapel
  RechtsDrehen
  RückSchritt stapel
  RechtsDrehen
  RückSchritt stapel
  LinksDrehen
  RückSchritt stapel
  umdrehen
*Anweisung
```

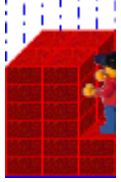
Hier geht er erst zur nächsten Wand.

Dann baut er im Rückwärtsgehen den Turm.



```
// === Hauptprogramm ========
wiederhole 2 mal
  wiederhole 4 mal
    solange NichtIstWand tue
      Hinlegen Schritt
    *solange
    umdrehen Schritt
    RechtsDrehen Hinlegen
    RechtsDrehen Schritt
    LinksDrehen
  *wiederhole
*wiederhole
```

Karol baut die zweilagige Mauer und sieht in jeder Ecke auch den Ziegel für den Turm schon vor.



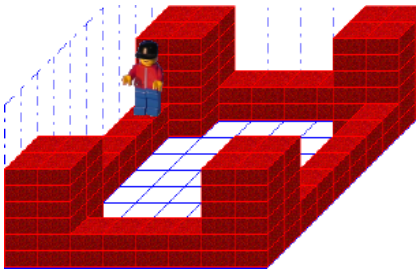
```
wiederhole 4 mal
  Turmbauen
*wiederhole
```

Karol baut auf die Mauer die vier Ecktürme

Struktogramm und Befehlsliste 9

09EcktürmeBauen.kdp

Welt beliebig
Lösung mit vier Anweisungen, gezeählte und solange-tue-Wiederholung innerhalb der Anweisungen, zwei verschachtelten gezählten Wiederholungen für die zweilagige Mauer und einer gezählten Wiederholung für die Ectürme



Hauptprogramm

wiederhole 2 **mal**	
	wiederhole 4 **mal**
	solange NichtIstWand
	Hinlegen
	Schritt
	UMDREHEN
	Schritt
	RechtsDrehen
	Hinlegen
	RechtsDrehen
	Schritt
	LinksDrehen
wiederhole 4 **mal**	
	TURMBAUEN

Anw.: UMDREHEN

LinksDrehen
LinksDrehen

Anw.: SCHRITTRÜCKWÄRTS

UMDREHEN
Schritt
UMDREHEN

Anw.: STAPEL

wiederhole 4 **mal**	
	Hinlegen

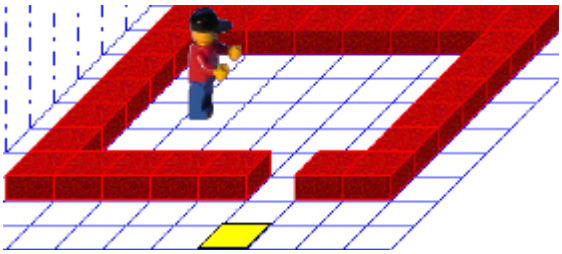
Anw.: TURMBAUEN

solange NichtIstWand	
	Schritt
SCHRITTRÜCKWÄRTS	
STAPEL	
RechtsDrehen	
SCHRITTRÜCKWÄRTS	
STAPEL	
RechtsDrehen	
SCHRITTRÜCKWÄRTS	
STAPEL	
LinksDrehen	
SCHRITTRÜCKWÄRTS	
STAPEL	
UMDREHEN	

```
// --- Anweisungen ------------
Anweisung Umdrehen
  LinksDrehen LinksDrehen
*Anweisung
// ----------------------------
Anweisung Schrittrückwärts
  Umdrehen Schritt Umdrehen
*Anweisung
// ----------------------------
Anweisung Stapel
  wiederhole 4 mal
    Hinlegen
  *wiederhole
*Anweisung
// ----------------------------
Anweisung Turmbauen
  solange NichtIstWand tue
    Schritt
  *solange
  Schrittrückwärts stapel
  RechtsDrehen Schrittrückwärts stapel
  RechtsDrehen Schrittrückwärts stapel
  LinksDrehen Schrittrückwärts stapel
  umdrehen
*Anweisung

// === Hauptprogramm ===========
wiederhole 2 mal
  wiederhole 4 mal
    solange NichtIstWand tue
      Hinlegen Schritt
    *solange
    umdrehen Schritt
    RechtsDrehen Hinlegen
    RechtsDrehen Schritt LinksDrehen
  *wiederhole
*wiederhole
wiederhole 4 mal
  Turmbauen
```

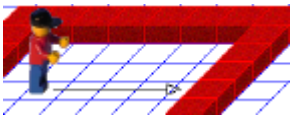
10: Karol sucht den Zimmerausgang

Aufgabe: Karol ist in einem Zimmer mit nur einem Ausgang; egal wo Karol steht er muss durch die Türe finden und auf dem Teppich vor dem Zimmer stehen bleiben

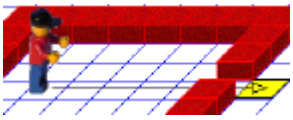


Welt: Größe der Welt und des Zimmers beliebig
Weltdateien: 10Zimmerausgang.kdw
Lösung: 10Zimmerausgang.kdp

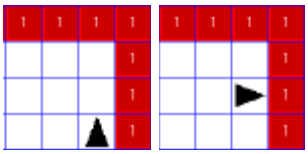
Befehle:	`Schritt, Linksdrehen, Rechtsdrehen`
Bedingungen:	`IstZiegel, NichtIstZiegel, NichtIstMarke`
Kontrollstrukturen:	`Solang ... tue` / `...` / `*solange` — `wenn ... dann` / `...` / `sonst` / `..` / `*wenn` — *Verschachtelt:* `solange ...` / `wenn ...`



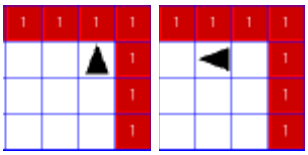
Karol geht zur Wand



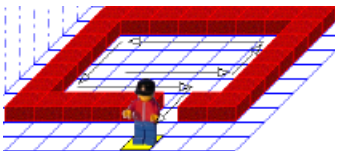
Er findet auf Anhieb die Tür



Karol schaut nach jedem Schritt nach rechts, ob die Tür erreicht ist.



In der Ecke dreht er nach links und setzt seine Suche fort.



```
// --- Anweisungen -----------
Anweisung Wandsuchen
  solange NichtIstZiegel tue
    Schritt
    wenn IstMarke dann
      Beenden
    *wenn
  *solange
*Anweisung
```

Karol läuft geradeaus, bis er die Wand erreicht oder – weil er auf Anhieb die Tür findet – auf dem Teppich steht. In diesem Fall beendet er das Programm.

```
// -------------------------
Anweisung Türsuchen
  solange IstZiegel tue
    linksdrehen
    wenn NichtIstZiegel dann
      Schritt
      rechtsdrehen
    sonst
      LinksDrehen
      Schritt
      rechtsdrehen
    *wenn
  *solange
// -------------------------
Anweisung ZuTeppichgehen
  solange NichtIstMarke tue
     schritt
  *solange
*Anweisung
```

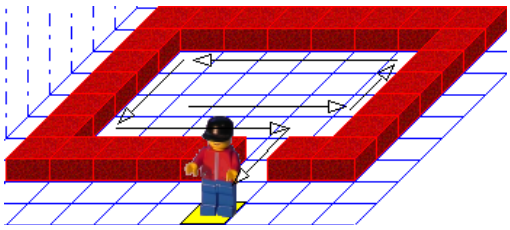
Solange Karol vor einem Ziegel steht,
- dreht er sich nach links
- Wenn kein Ziegel vor ihm ist
  + macht er einen Schritt
  + und dreht sich wieder zur Wand;
  sonst (er hat eine Ecke erreicht)
  + dreht er sich nochmals links
  + und macht einen Schritt.

Karol geht durch die Tür. Dabei macht er solange Schritte, bis er auf dem Teppich (=Marke) steht.

```
// === Hauptprogramm ========
Wandsuchen
Türsuchen
ZuTeppichgehen
```

Struktogramm und Befehlsliste 10

10Zimmerausgang.kdp



Größe der Welt, des Zimmers und die Position Karols im Zimmer beliebig

Lösung mit drei Anweisungen. Solange-Tue-Wiederholung innerhalb der Anweisungen, verschachtelt mit wenn bzw. wenn-dann-sonst

Hauptprogramm

WANDSUCHEN
TÜRSUCHEN
ZUTEPPICHGEHEN

Anw.: WANDSUCHEN

solange NichtIstZiegel		
	Schritt	
	IstMarke	
	w	**f**
	Beenden	

Anw.: TÜRSUCHEN

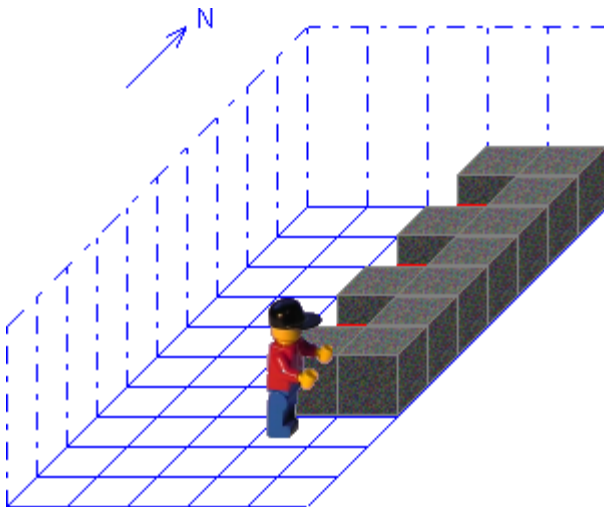
solange IstZiegel		
	LinksDrehen	
	NichtIstZiegel	
	w	**f**
	Schritt	LinksDrehen
	RechtsDrehen	Schritt
		RechtsDrehen

Anw.: ZUTEPPICHGEHEN

solange NichtIstMarke
Schritt

```
// --- Anweisungen ------------
Anweisung Wandsuchen
  solange NichtIstZiegel tue
    Schritt
    wenn IstMarke dann
      Beenden
    *wenn
  *solange
*Anweisung
// ----------------------------
Anweisung Türsuchen
  solange IstZiegel tue
    linksdrehen
    wenn NichtIstZiegel dann
      Schritt
      rechtsdrehen
    sonst
      LinksDrehen
      Schritt
      rechtsdrehen
    *wenn
  *solange
*Anweisung
// ----------------------------
Anweisung ZuTeppichgehen
  solange NichtIstMarke tue
     schritt
  *solange
*Anweisung

// === Hauptprogramm ==========
Wandsuchen
Türsuchen
ZuTeppichgehen
```

11: Karol füllt ein Regal an der Ostwand mit Ziegeln



Aufgabe:
An der Ostwand von Karols Werkstatt ist ein Regal aus grauen Quadern aufgebaut. Das Regal ist kürzer als die Wand.
In die Regalnischen soll Karol Ziegel legen.
Wenn er das Ende des Regals erreicht hat, soll er seine Arbeit beenden

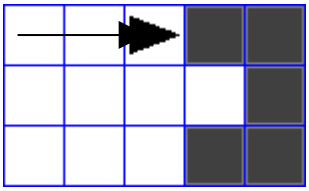
Welt: Größe beliebig
Weltdateien: 11regal1.kdw, 11regal2.kdw
Lösung: 11regal1.kdp

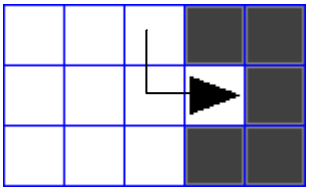
Befehle: `Schritt, Hinlegen, Linksdrehen, Rechtsdrehen`

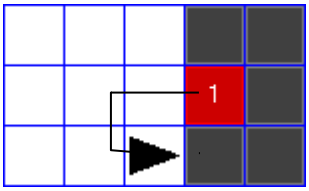
Bedingungen: `IstWand, NichtIstWand`

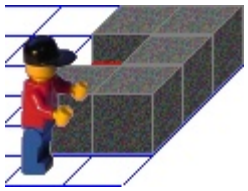
Kontrollstrukturen:

```
Solang ... tue        wenn ... dann        Verschachtelt:
   ...                   ...               solange ...
*solange              *wenn                    wenn ...
```









Karol am Regalende

```
// 1. Karol läuft zum Regal
LinksDrehen
solange NichtIstWand tue
  Schritt
*solange
// 2. Vor der Regalwand wieder-
holt er
solange IstWand tue
  RechtsDrehen
  Schritt
  LinksDrehen
  Schritt
  wenn IstWand dann
    LinksDrehen LinksDrehen
    Schritt
    RechtsDrehen RechtsDrehen
    Hinlegen
    RechtsDrehen
    Schritt
    Linksdrehen
  *wenn
*solange
```

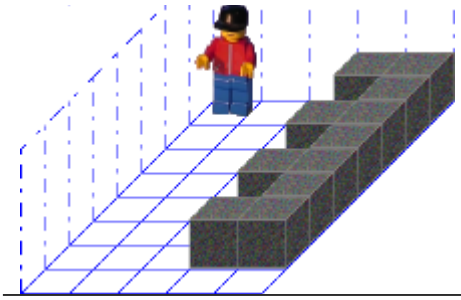
Karol dreht sich nach links und läuft zur hinteren Regalecke

Solange Karol vor einer Regalecke steht, läuft er um die Regalecke in die Regalnische

Wenn er in einer Regalnische steht (IstWand), dann dreht er um,
legt in die Nische einen Ziegel,
dreht nach rechts,
läuft einen Schritt und
schaut wieder Richtung Regal

Struktogramm und Befehlsliste 11

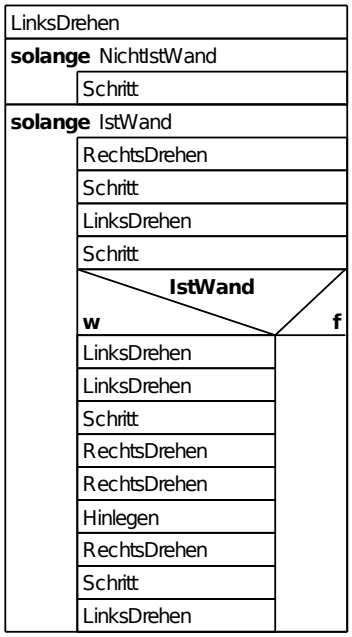
11regal1.kdp



An der Ostwand von Karols Werkstatt ist ein Regal aus grauen Quadern aufgebaut. Das Regal ist kürzer als die Wand.

In die Regalnischen soll Karol Ziegel legen. Wenn er das Ende des Regals erreicht hat, soll er seine Arbeit beenden

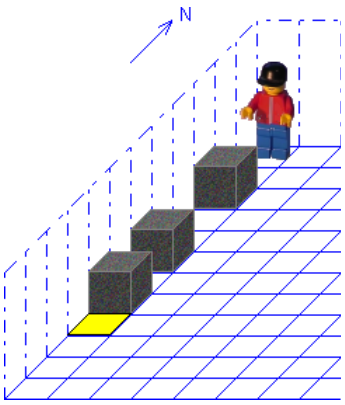
Hauptprogramm




```
// zur hinteren Regalecke laufen
Linksdrehen
Solange NichtIstWand tue
  Schritt
*solange
// solange vor Regal, Nischen belegen
Solange IstWand tue
  // um Regalecke laufen
  Rechtsdrehen
  Schritt
  Linksdrehen
  Schritt
  // Ist das noch eine Nische?
  wenn IstWand dann
     // aus der Nische laufen
     linksdrehen linksdrehen
     schritt
     rechtsdrehen rechtsdrehen
     // Stein hineinlegen
     Hinlegen
     // Schritt weiter gehen
     Rechtsdrehen
     Schritt
     Linksdrehen
  *wenn
*solange
```

Fach:	Name:	Datum:

12: Karol sucht ein Handy



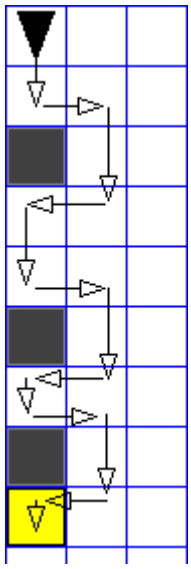
Aufgabe 1: Einfache Suche

Der Meister hat in der Maschinenhalle sein Handy an der Wand neben einer Maschine liegen lassen. Robot Karol soll es suchen und einstecken. Auf der Suche stehen ihm jedoch immer wieder die Maschinen an der Westwand (1 Maschine = 1 Quader) im Weg.
Welt: Größe beliebig

Weltdateien: 12handy1.kdw, 12handy2.kdw, 12handy3.kdw
Lösungen: 12handy1.kdp, 12handy2.kdp, 12handy3.kdp,

Befehle:	`Schritt, MarkeLöschen, Linksdrehen, Rechtsdrehen`
Bedingungen:	`NichtIstMarke, NichtIstWand, IstWand`

Kontrollstrukturen:	`Solang ... tue` `...` `*solange`
	`wenn ... dann` `...` `*wenn`
	`Anweisung ...` `...` `*Anweisung`
	`Verschachtelt: solange ... wenn ... sonst`



// Verwendete Anweisung

```
Anweisung UmMaschineLaufen
  LinksDrehen
  Schritt
  RechtsDrehen
  Schritt
  Schritt
  RechtsDrehen
  Schritt
  Linksdrehen
*Anweisung
```

Karol dreht nach links und läuft um die Maschine herum. Dann dreht er sich wieder in seine Ausgangsrichtung.

// Hauptprogramm

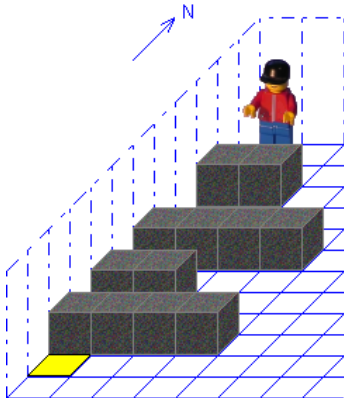
```
solange NichtIstMarke tue
  wenn IstWand dann
    UmMaschineLaufen
  sonst
    Schritt
  *wenn
*solange
MarkeLöschen
```

Solange Karol nicht das Handy gefunden hat, tut er folgendes:
- Wenn er auf eine Maschine trifft, läuft er um sie herum,
- sonst macht er einen Schritt.

Nun hebt er das gefundene Handy auf.

Aufgabe 2: Schwierigere Suche

Die Maschinen an der Westwand sind verschieden lang. Die Anweisung UmMaschineLaufen muss überarbeitet werden.

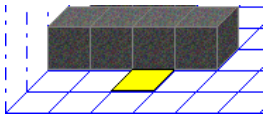



```
Anweisung UmMaschineLaufen
  solange IstWand tue
    LinksDrehen
    Schritt
    RechtsDrehen
  *solange
  Schritt
  Schritt
  RechtsDrehen
  solange NichtIstWand tue
    Schritt
  *solange
  linksdrehen
*Anweisung
```

Solange Karol vor der Maschine steht, tut er folgendes:
- Er dreht sich nach links, macht einen Schritt und dreht sich wieder nach rechts zur Maschine.

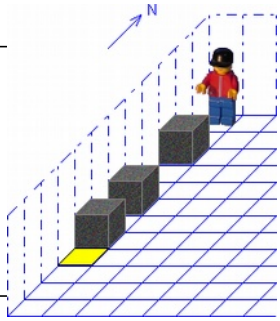
Hat er das Maschinenende erreicht,
- läuft er mit 2 Schritten an ihr vorbei,
- dreht sich nach rechts,
- läuft bis zur Wand und
- dreht sich wieder nach links.

Aufgabe 3: Komplizierte Suche



Die Maschinen an der Westwand sind verschieden lang und das Handy muss nicht an der Wand liegen.

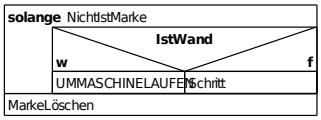
Struktogramm und Befehlsliste 12



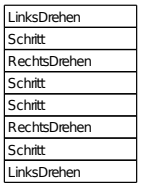
12handy1.kdp

Der Meister hat in der Maschinenhalle sein Handy an der Wand neben einer Maschine liegen lassen. Robot Karol soll es suchen und einstecken. Auf der Suche stehen ihm jedoch immer wieder die Maschinen an der Westwand (1 Maschine = 1 Quader) im Weg.

Hauptprogramm



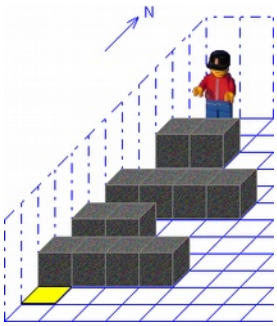
Anw.: UMMASCHINELAUFEN




```
// --- Anweisungen ---
Anweisung UmMaschineLaufen
  LinksDrehen  Schritt
  RechtsDrehen Schritt Schritt
  RechtsDrehen Schritt
  Linksdrehen
*Anweisung
// === Hauptprogram ===
solange NichtIstMarke tue
  wenn IstWand dann
    UmMaschineLaufen
  sonst
    Schritt
  *wenn
*solange
MarkeLöschen
```

12handy2.kdp

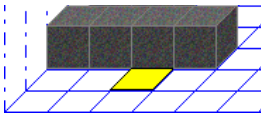
Die Maschinen an der Westwand sind verschieden lang. Die Anweisung UmMaschineLaufen muss überarbeitet werden.




```
// --- Anweisungen ---
Anweisung UmMaschineLaufen
  solange IstWand tue
    LinksDrehen Schritt RechtsDrehen
  *solange
  Schritt Schritt RechtsDrehen
  solange NichtIstWand tue
    Schritt
  *solange
  linksdrehen
*Anweisung
// === Hauptprogram ===
... (wie oben)
```

12handy3.kdp

Die Maschinen an der Westwand sind verschieden lang und das Handy muss nicht an der Wand liegen.

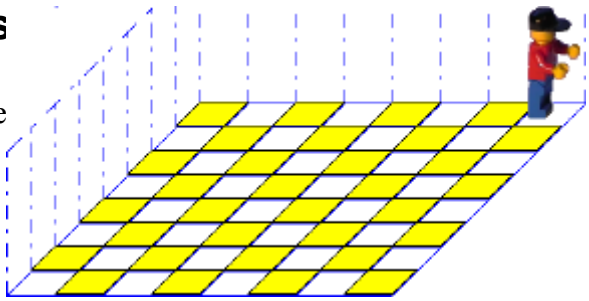


```
// --- Anweisungen ---
Anweisung AnMaschineEntlang
  LinksDrehen
  Schritt
  RechtsDrehen
  wenn NichtIstMarke dann
    wenn NichtIstWand dann
      Schritt
      wenn NichtIstMarke dann
        Schritt
      *wenn
      RechtsDrehen
    *wenn
  *wenn
*Anweisung
```

```
// === Hauptprogram ===
solange NichtIstMarke tue
  wenn IstWand dann
    wenn IstWesten dann
      LinksDrehen
    sonst
      AnMaschineEntlang
    *wenn
  sonst
    Schritt
  *wenn
*solange
MarkeLöschen
```

Fach:	Name:	Datum:

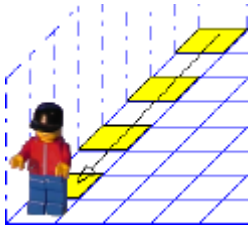
13: Karol erstellt ein Schachbrettmus



Aufgabe:
Karel soll seine Welt mit einem Schachbrettmuster aus laute
soll keine Rolle spielen.

Welt: Größe beliebig
Weltdateien: ---
Lösungen: 13Schachbrett.kdp

Befehle:	`Schritt, MarkeLöschen, Linksdrehen, Rechtsdrehen`
Bedingungen:	`NichtIstMarke, NichtIstWand, IstWand`
Kontrollstrukturen:	`Solang ... tue` `...` `*solange` — `wenn ... dann` `...` `*wenn` — `Anweisung ...` `...` `*Anweisung` — *`Verschachtelt: solange ... wenn ... sonst`*



Karol markiert eine Bahn.

// Verwendete Anweisung

```
Anweisung MarkiereBahn
  solange NichtIstWand tue
    MarkeSetzen
    Schritt
    wenn NichtIstWand dann
      Schritt
    *wenn
  *solange
*Anweisung
```

Karol markiert in einer Bahn jedes zweite Feld mit einer Marke.

Dazu setzt er, solange er nicht die Wand erreicht hat, eine Marke und macht zwei Schritte.

Beim zweiten Schritt muss er allerdings prüfen, ob er schon vor einer Wand steht.

// Hauptprogramm

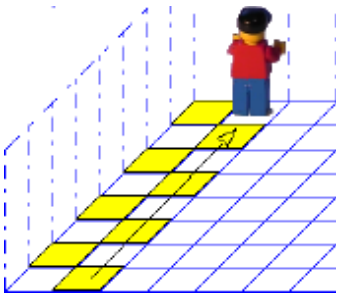
```
solange NichtIstWand tue
  MarkiereBahn
  LinksDrehen
```

Solange Karol nicht an die Ostwand stößt
- markiert er die Bahn nach S
- dreht nach links

Steht Karol noch nicht vor der Westwand, schwenkt nach Norden ...

```
  wenn NichtIstWand dann
    Schritt
    LinksDrehen
  *wenn
```

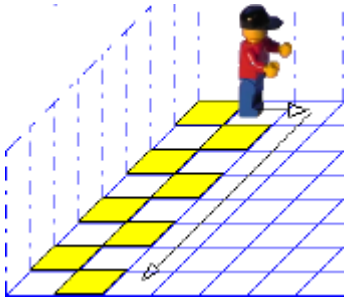
Steht er noch nicht vor einer Wand,
- macht er einen Schritt
- dreht er nochmals nach links,



und markiert die Bahn nach Norden.

```
  MarkiereBahn
```

- und markiert die Bahn nach N

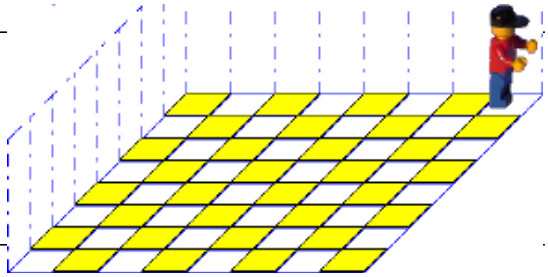


Steht er noch nicht vor der Westwand,
schwenkt er nach Süden
und markiert die nächste Bahn

```
  RechtsDrehen
  wenn NichtIstWand dann
    Schritt
    RechtsDrehen
  *wenn
*solange
```

Er dreht nach rechts.
Hat er noch nicht die Ostwand erreicht, dreht er sich wieder nach S und beginnt mit dem Markieren wieder von vorn.

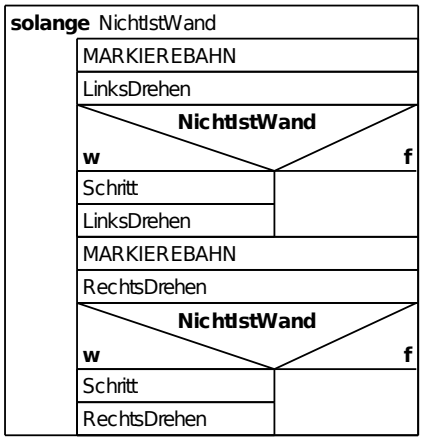
Struktogramm und Befehlsliste 13



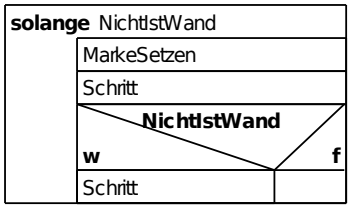
13Schachbrett.kdp

Karel soll seine Welt mit einem Schachbrettmuster aus lauter Marken auslegen. Die Größe der Welt soll keine Rolle spielen.

Hauptprogramm



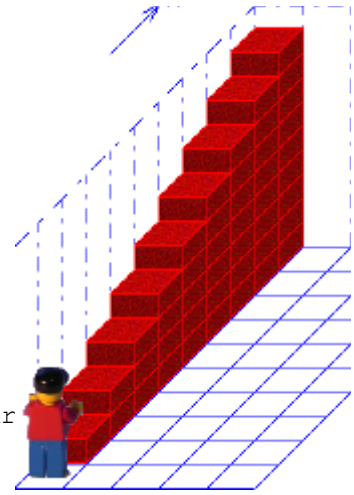
Anw.: MARKIEREBAHN




```
// --- Anweisung --------------
Anweisung MarkiereBahn
  solange NichtIstWand tue
    MarkeSetzen
    Schritt
    wenn NichtIstWand dann
      Schritt
    *wenn
  *solange
*Anweisung
// === Hauptprogramm ==========
solange NichtIstWand tue
  MarkiereBahn
  LinksDrehen
  wenn NichtIstWand dann
    Schritt
    LinksDrehen
  *wenn
  MarkiereBahn
  RechtsDrehen
  wenn NichtIstWand dann
    Schritt
    RechtsDrehen
  *wenn
*solange
```

Fach:	Name:	Datum:

14: Karol baut eine Treppe zur Südwand



Aufgabe:
Karel soll eine Treppe zur Nordwand bauen. Die Welt ist 10 Felder lang und 10 Ziegel hoch. Karel starte in der linken oberen Ecke.

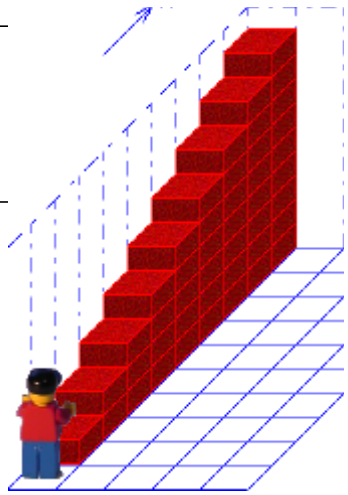
Welt: 11 x 5 x 10
Weltdateien: 14Treppe.kdw
Lösungen: 14Treppe.kdp

Befehle:	`Schritt, MarkeLöschen, Linksdrehen, Rechtsdr`			
Bedingungen:	`NichtIstMarke, NichtIstWand, IstWand`			

Kontrollstrukturen:	`Solang ... tue`
	`...`
	`*solange`
		`wenn ... dann`
		`...`
		`*wenn`
			`Anweisung ...`
			`...`
			`*Anweisung`
				Verschachtelt:
				solange ...
				wenn ... sonst

```
// --- Anweisungen -------
Anweisung Umkehren
  LinksDrehen
  LinksDrehen
*Anweisung
// ------------------------
Anweisung SchichtBisWand
  solange NichtIstWand tue
    Hinlegen
    Schritt
  *solange
*Anweisung
// ------------------------
Anweisung TreppAb
  solange IstZiegel tue
    Schritt
  *solange
  Schritt
  Schritt
*Anweisung
// === Hauptprogramm =======
Schritt
wiederhole 9 mal
  Umkehren
  SchichtBisWand
  umkehren
  Treppab
*wiederhole
```

Struktogramm und Befehlsliste 14



14Treppe.kdp

Karel soll eine Treppe zur Nordwand bauen. Die Welt ist 10 Felder lang und 10 Ziegel hoch. Karel starte in der linken oberen Ecke.

Hauptprogramm

Schritt	
wiederhole 9 **mal**	
	UMKEHREN
	SCHICHTBISWAND
	UMKEHREN
	TREPPAB

Anw.: UMKEHREN

LinksDrehen
LinksDrehen

Anw.: SCHICHTBISW

solange NichtIstWand	
	Hinlegen
	Schritt

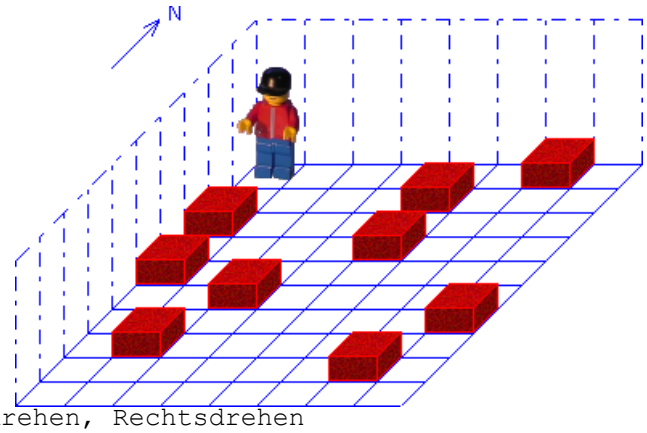
Anw.: TREPPAB

solange IstZiegel	
	Schritt
Schritt	
Schritt	

```
// --- Anweisungen -------
Anweisung Umkehren
  LinksDrehen
  LinksDrehen
*Anweisung
// ------------------------
Anweisung SchichtBisWand
  solange NichtIstWand tue
    Hinlegen
    Schritt
  *solange
*Anweisung
// ------------------------
Anweisung TreppAb
  solange IstZiegel tue
    Schritt
  *solange
  Schritt
  Schritt
*Anweisung
// === Hauptprogramm =======
Schritt
wiederhole 9 mal
  Umkehren
  SchichtBisWand
  umkehren
  Treppab
*wiederhole
```

Fach:	Name:	Datum:

15: Karol räumt auf

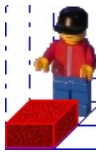


Aufgabe 1: Einzelne Ziegel

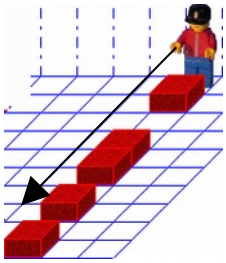
Es ist Feierabend und die Baustelle ist ein einziges Chaos. Überall liegen verstreute Ziegel herum. Karol muss die Reihen ablaufen und alle Ziegel einsammeln. Dann soll er wieder an seinen Ruheplatz zurückkehren.

Welt: 10 x 8
Weltdatei: 15RaeumAuf1.kdw, 15RaeumAuf2.kdw
Lösung: 15RaeumAuf1.kdp, 15RaeumAuf2.kdp,

Befehle:	`Schritt, Aufheben, Linksdrehen, Rechtsdrehen`
Bedingungen:	`NichtIstZiegel, IstZiegel, NichtIstWand, IstWand, IstSüden`
Kontrollstrukturen:	`Solang ... tue` / `...` / `*solange` — `wenn ... dann` / `sonst` / `*wenn` — `Anweisung ...` / `...` / `*Anweisung` — `wiederhole .. mal` / `...` / `*wiederhole`

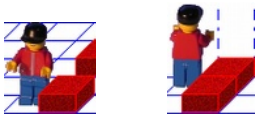


Ziegelweg und Schritt

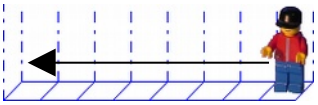


Bahnfrei

Wenden



nach N nach S



Zurück zum Start

// Verwendete Anweisungen

```
Anweisung Ziegelweg
  Wenn IstZiegel dann
    Aufheben
  *Wenn
  Wenn NichtIstWand dann
    Schritt
  *Wenn
*Anweisung
Anweisung BahnFrei
  solange NichtIstWand tue
    Ziegelweg
  *solange
*Anweisung
Anweisung Wenden
  wenn IstSüden dann
    LinksDrehen
    Ziegelweg
    LinksDrehen
  sonst
    RechtsDrehen
    Ziegelweg
    RechtsDrehen
  *wenn
*Anweisung
Anweisung ZurückZumStart
  RechtsDrehen
  solange NichtIstWand tue
    Schritt
  *solange
  LinksDrehen
*Anweisung
```

Karol schaut, ob vor ihm ein Ziegel liegt, und hebt diesen dann auf:

Wenn er nicht vor einer Wand steht dann macht er einen Schritt vorwärts.

Solange Karol nicht vor einer Wand steht, räumt er alle Ziegel auf dieser Bahn weg.

Am Ende einer Bahn muss er wenden. Wenn er nach Süden schaut wendet er links herum, sonst rechtsherum

Hat er alle 8 Bahnen geräumt, kehrt er zum Start zurück

// Hauptprogramm

```
wiederhole 8 mal
  BahnFrei
  Wenden
*wiederhole
ZurückZumStart
```

Karol muss 8 mal eine Bahn frei räumen und wenden.
Dann kehrt er zum Start zurück.

Aufgabe 2: Mehrere Ziegel übereinander gestapelt

Es können auch mehrere Ziegel aufeinander gestapelt sein (RaeumAuf2.kdw) Man muss nur die Anweisung ZiegelWeg anpassen! Statt WennIstZiegel ... Solange IstZiegel ...

Fach:	Name:	Datum:

Struktogramm und Befehlsliste 15

15RaeumAuf1.kdp

Größe der Welt: 10 x 8; Position Karols: links oben
Überall liegen verstreute Ziegel herum. Karol muss die Reihen ablaufen und alle Ziegel einsammeln. Dann soll er wieder an seinen Ruheplatz zurückkehren.

Hauptprogramm

wiederhole 8 **mal**	
	BAHNFREI
	WENDEN
ZURÜCKZUMSTART	

Anw.: ZIEGELWEG

IstZiegel	
w	**f**
Aufheben	
NichtIstWand	
w	**f**
Schritt	

Anw.: BAHNFREI

solange NichtIstWand	
	ZIEGELWEG

Anw.: WENDEN

IstSüden	
w	**f**
LinksDrehen	RechtsDrehen
ZIEGELWEG	ZIEGELWEG
LinksDrehen	RechtsDrehen

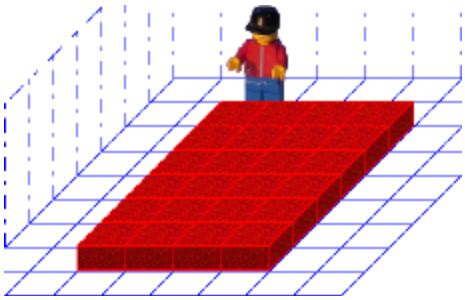
Anw.: ZURÜCKZUMSTART

RechtsDrehen	
solange NichtIstWand	
	Schritt
LinksDrehen	

```
Anweisung Ziegelweg
    wenn IstZiegel dann
      Aufheben
    *wenn
    wenn NichtIstWand dann
      Schritt
    *wenn
*Anweisung
// ------------------
Anweisung BahnFrei
  solange NichtIstWand tue
    Ziegelweg
  *solange
*Anweisung

// ------------------
Anweisung Wenden
  wenn IstSüden dann
    LinksDrehen
    Ziegelweg
    LinksDrehen
  sonst
    RechtsDrehen
    Ziegelweg
    RechtsDrehen
  *wenn
*Anweisung
// ------------------
Anweisung ZurückZumStart
  RechtsDrehen
  solange NichtIstWand tue
    Schritt
  *solange
  LinksDrehen
*Anweisung
// +++ Hauptprogram +++
wiederhole 8 mal
  BahnFrei
  Wenden
*wiederhole
ZurückZumStart
// ------------------
```

Fach:	Name:	Datum:

16: Karol räumt ein Beet leer



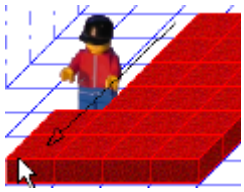
Aufgabe 1: Karol steht vor der Beetecke

Es ist Herbst und Karol steht vor einem Gartenbeet mit lauter Pflanzen (Ziegel) in jedem Feld. Er soll es mit möglichst wenig Schritten abräumen.

Welt: beliebig
Weltdatei: 16Gartenbeet1.kdw
Lösung: 16Gartenbeet1.kdp

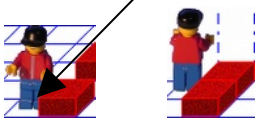
Befehle:	`Schritt, Aufheben, Linksdrehen, Rechtsdrehen`
Bedingungen:	`NichtIstZiegel, IstZiegel, NichtIstWand, IstWand, IstSüden`

Kontrollstrukturen:	`Solang ... tue` `...` `*solange`
	 `wenn ... dann` `sonst` `*wenn`
	 `Anweisung ...` `...` `*Anweisung`
	 `wiederhole .. mal` `...` `*wiederhole`



AufhebenReihe

Wenden



nach N nach S

// Verwendete Anweisungen

```
Anweisung AufhebenReihe
  solange IstZiegel tue
    Aufheben
    Schritt
  *solange
*Anweisung
Anweisung Wenden
  wenn IstSüden dann
    LinksDrehen
    Ziegelweg
    LinksDrehen
  sonst
    RechtsDrehen
    Ziegelweg
    RechtsDrehen
  *wenn
*Anweisung
```

Karol schaut, ob vor ihm ein Ziegel liegt, und hebt diesen dann auf und macht dann einen Schritt vorwärts.

Am Ende einer Bahn muss er wenden. Wenn er nach Süden schaut wendet er links herum, sonst rechtsherum

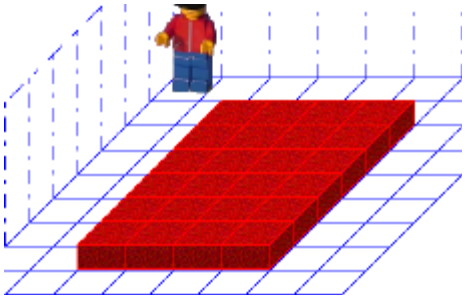
// Hauptprogramm

```
solange IstZiegel tue
  AufhebenReihe
  Wenden
*solange
```

Karol muss 8 mal eine Bahn frei räumen und wenden.
Dann kehrt er zum Start zurück.

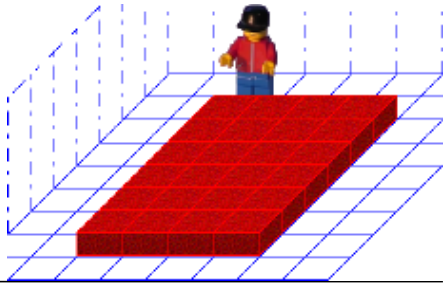
Aufgabe 2: Karol muss die Beetecke erst suchen

Karol muss das Feld erst suchen, weil er in der Ecke links oben startet.



Struktogramm und Befehlsliste 16

16Gartenbeet1.kdp



Größe der Welt und des Gartebeetes beliebig; Position Karols: An der NO-Ecke des Beetes
Es ist Herbst und Karol steht vor einem Gartenbeet mit lauter Pflanzen (Ziegel) in jedem Feld. Er soll es mit möglichst wenig Schritten abräumen.

Hauptprogramm

solange IstZiegel	
	AUFHEBENREIHE
	WENDEN

Anw.: AUFHEBENREIHE

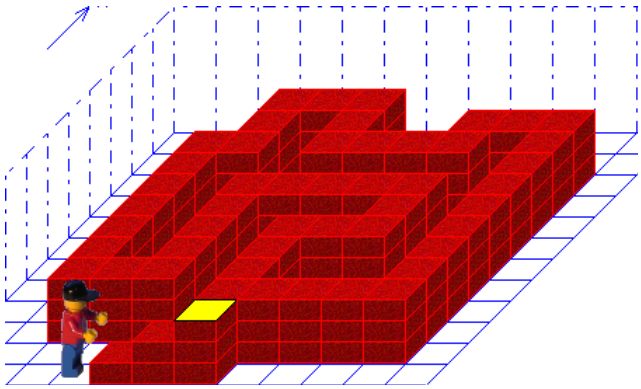
solange IstZiegel	
	Aufheben
	Schritt

Anw.: WENDEN

w \ **IstSüden**	**f**
Schritt	Schritt
LinksDrehen	RechtsDrehen
Schritt	Schritt
LinksDrehen	RechtsDrehen

```
// --- Anweisungen ---------------
Anweisung AufhebenReihe
  solange IstZiegel tue
    Aufheben
    Schritt
  *solange
*Anweisung
// -------------------------------
Anweisung Wenden
  wenn IstSüden dann
    Schritt
    LinksDrehen
    Schritt
    LinksDrehen
  sonst
    Schritt
    RechtsDrehen
    Schritt
    RechtsDrehen
  *wenn
*Anweisung
// === Hauptprogramm ============
solange IstZiegel tue
  AufhebenReihe
  Wenden
*solange
```

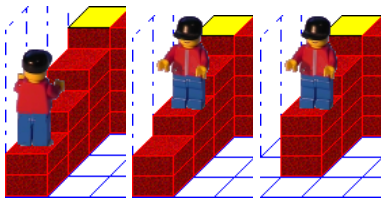
17: Karol auf Wacht



Aufgabe 1: Ein Kontrollgang auf der Mauer
Karol soll einen Burgwall ablaufen, ohne herunter zu fallen. Dazu muss er bei jedem Schritt prüfen, in welche Richtung es weiter geht. Zunächst steigt er die Treppe hoch, deren Ende markiert ist. Zur Sicherheit vor einem Überfall baut er hinter sich die Treppe ab.

Welt: Größe beliebig
Weltdateien: 17Wache1.kdw, 17Wache2.kdw
Lösungen: 17Wache1.kdp, 17Wache2.kdp

Befehle:	`Schritt, Aufheben, Linksdrehen, Rechtsdrehen`
Bedingungen:	`NichtIstMarke, IstZiegl, IstWand`
Kontrollstrukturen:	`Solang ... tue` / `...` / `*solange` — `wenn ... dann` / `...` / `*wenn` — `Anweisung ...` / `...` / `*Anweisung` — *`Verschachtelt:`* / *`wenn ...`* / *`wenn ...`* / *`wenn ...`*



Es geht geradeaus

Geht es links weiter? Ja!

Wenn nicht, geht es rechts weiter?

// Verwendete Anweisung

```
Anweisung TreppeSteigen
  Schritt
  LinksDrehen LinksDrehen
  solange IstZiegel tue
    Aufheben
  *solange
  RechtsDrehen RechtsDrehen
*Anweisung
Anweisung WacheGehen
  wenn IstZiegel dann
    Schritt
  sonst
    LinksDrehen
    wenn IstZiegel dann
      Schritt
    sonst
      RechtsDrehen
      RechtsDrehen
      wenn IstZiegel dann
        Schritt
      sonst
        WacheGehen
      *wenn
    *wenn
  *wenn
*Anweisung
```

Karol macht einen Schritt,
dreht sich um,
hebt so lange Ziegel auf, bis alle abgebaut sind,
dreht sich wieder Richtung Treppe

Wenn ein Ziegel vor Karol liegt
macht er einen Schritt
sonst dreht er sich nach links.

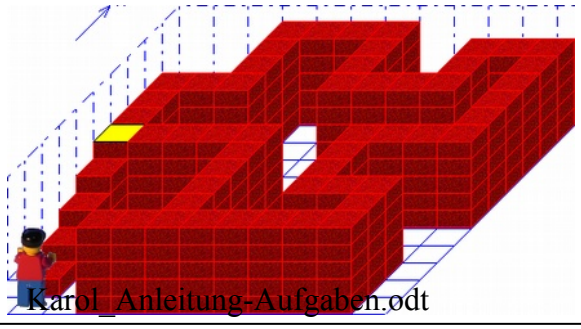
Liegt jetzt ein Ziegel vor ihm,
macht er einen Schritt
sonst dreht er sich um 180 °

Liegt jetzt ein Ziegel vor ihm,
macht er einen Schritt
sonst probiert er es noch einmal von vorne, weil er in einer Sackgasse steckt.

// Hauptprogramm

```
solange NichtIstMarke tue
  TreppeSteigen
*solange
WacheGehen
solange NichtIstMarke tue
  Wachegehen
*solange
MarkeLöschen
```

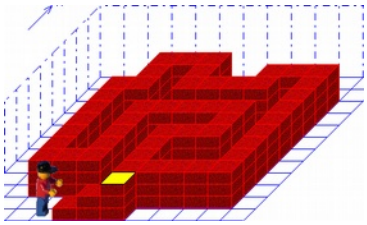
Solange Karol nicht die Marke am Ende der Treppe erreicht hat, steigt er eine Stufe hoch.
Dann geht er einen Schritt weiter, um die Marke zu verlassen.
Jetzt geht er so lange Wache, bis er zur Marke zurück kehrt.



Aufgabe 2: Mehrere Kontrollgänge, Sackgasse
Karol soll jetzt vier Runden Wache gehen.
Außerdem besitzt die Burgmauer eine „Sackgasse“.

Struktogramm und Befehlsliste 17

17Wache1.kdp



Karol soll einen Burgwall ablaufen, ohne herunter zu fallen. Dazu muss er bei jedem Schritt prüfen, in welche Richtung es weiter geht. Zunächst steigt er die Treppe hoch, deren Ende markiert ist. Zur Sicherheit vor einem Überfall baut er hinter sich die Treppe ab.
Größe der Welt und der Burg beliebig; Position Karols: Vor der Treppe

Hauptprogramm

solange NichtIstMarke
TREPPESTEIGEN
WACHEGEHEN
solange NichtIstMarke
WACHEGEHEN

Anw.: TREPPESTEIGEN

Schritt
LinksDrehen
LinksDrehen
solange IstZiegel
Aufheben
RechtsDrehen
RechtsDrehen

Anw.: WACHEGEHEN

IstZiegel	
w / f	
Schritt	LinksDrehen
	IstZiegel
	w / f
	Schritt: RechtsDrehen
	RechtsDrehen
	Schritt

```
// --- Anweisungen -------
Anweisung TreppeSteigen
  Schritt
  LinksDrehen LinksDrehen
  solange IstZiegel tue
    Aufheben
  *solange
  RechtsDrehen RechtsDrehen
*Anweisung
// ----------------------
Anweisung WacheGehen
  wenn IstZiegel dann
    Schritt
  sonst
    LinksDrehen
    wenn IstZiegel dann
      Schritt
    sonst
      RechtsDrehen
      RechtsDrehen
      Schritt
    *wenn
  *wenn
*Anweisung
// === Hauptprogramm =====
solange NichtIstMarke tue
  TreppeSteigen
*solange
WacheGehen
solange NichtIstMarke tue
  Wachegehen
*solange
```

17Wache2.kdp

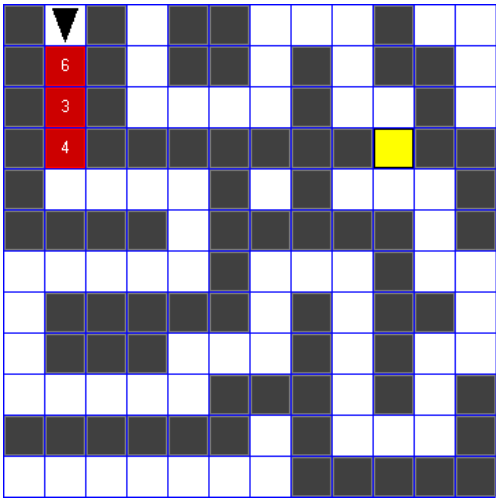
Hauptprogramm

solange NichtIstMarke
TREPPESTEIGEN
wiederhole 3 **mal**
WACHEGEHEN
solange NichtIstMarke
WACHEGEHEN

```
Anweisungen TreppeSteigen und Wache-
Gehen wie oben

Geändertes Hauptprogramm

// === Hauptprogramm ===
solange NichtIstMarke tue
  TreppeSteigen
*solange
wiederhole 3 mal
  WacheGehen
  solange NichtIstMarke tue
    Wachegehen
  *solange
*wiederhole
```

Fach:	Name:	Datum:

18: Karol bei den Pharaonen



Aufgabe 1: Nur der Eingang ist zugemauert

Karol soll im Labyrinth einer Pyramide einen Goldschatz suchen. Dazu muss er vor jedem Schritt prüfen, wie es weiter geht. Leider muss Karol erst den zugemauerten Eingang der Pyramide öffnen. Er weiß nicht, wie dick und wie hoch die Mauer ist. In der Pyramide kann man nicht im Kreis laufen.

Welt: Größe beliebig
Weltdateien: 18Pharao1.kdw, 18Pharao2.kdw
Lösungen: 18Pharao1.kdp, 18Pharao2.kdp

Befehle: `Schritt, Aufheben, Linksdrehen, Rechtsdrehe, Markelöschenn`

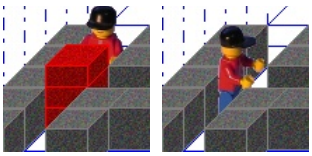
Bedingungen: `NichtIstMarke, NichtIstWand, IstWand`

Kontrollstrukturen:

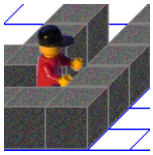
```
Solang ... tue        wenn ... dann      Anweisung ...
   ...                   ...                ...
*solange              *wenn              *Anweisung
```

Verschachtelt:

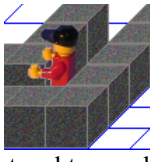
```
wenn ...
   wenn ...
solange ..
   wenn ...
```



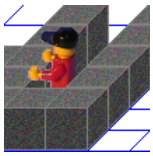
Ziegel abräumen, dann nach links schauen, ob Weg frei ist.; wenn nicht, wieder nach vor schauen



Karol steht vor einer Wand. Geht es links weiter? Nein!



Wenn nicht, geht es rechts weiter?



Oder ist es eine Sackgasse?

// Verwendete Anweisung

```
Anweisung MauerRäumen
  solange IstZiegel tue
    Aufheben
  *solange
  Schritt
  linksdrehen
  wenn IstWand dann
    rechtsdrehen
  *wenn
*Anweisung
```

Solagne Ziegel vor Karol liegen, räumt er sie weg
Dann macht er einen Schritt
Er schaut nach links.
Wenn dort eine Wand ist, schaut er wieder in die alte Richtung

```
Anweisung WegFinden
    linksdrehen
    wenn IstWand dann
      RechtsDrehen
      RechtsDrehen
      Wenn IstWand dann
        WegFinden
      *wenn
    *wenn
*Anweisung
```

Wenn ein Ziegel vor Karol liegt
macht er einen Schritt
sonst dreht er sich nach links.

Liegt jetzt ein Ziegel vor ihm,
macht er einen Schritt
sonst dreht er sich um 180 °

Liegt jetzt ein Ziegel vor ihm,
macht er einen Schritt
sonst probiert er es noch einmal von vorne, weil er in einer Sackgasse steckt.

// Hauptprogramm

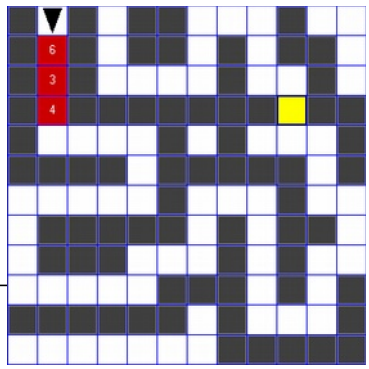
```
solange NichtIstMarke tue
  MauerRäumen
  wenn IstWand dann
    Wegfinden
  *wenn
*solange
MarkeLöschen
```

Solange Karol nicht die Marke am Ende der Treppe erreicht hat, steigt er eine Stufe hoch.
Dann geht er einen Schritt weiter, um die Marke zu verlassen.
Jetzt geht er so lange Wache, bis er zur Marke zurück kehrt.

Aufgabe 2: Gänge auch zum Teil vermauert; Rückweg finden

Ach die Gänge sind zum Teil zugemauert und Karol soll wieder zum Ausgang zurückfinden! Dazu setzt er zuerst am Eingang eine Marke, damit er wieder zurück findet.

Struktogramm und Befehlsliste 18

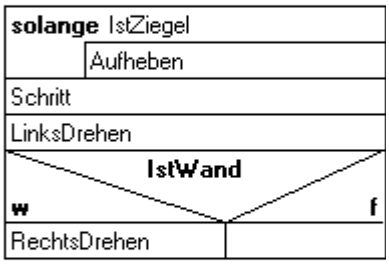


18Pharao1.kdp

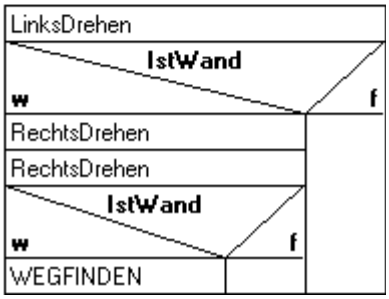
Karol soll im Labyrinth einen Goldschatz suchen. Leider muss Karol erst den zugemauerten Eingang der Pyramide öffnen. Er weiß nicht, wie dick und wie hoch die Mauer ist. In der Pyramide kann man nicht im Kreis laufen.
Größe der Welt, des Pyramiede: beliebig; Position Karols: Vor dem Eingang.

Hauptprogramm

Anw.: MAUERRÄUMEN

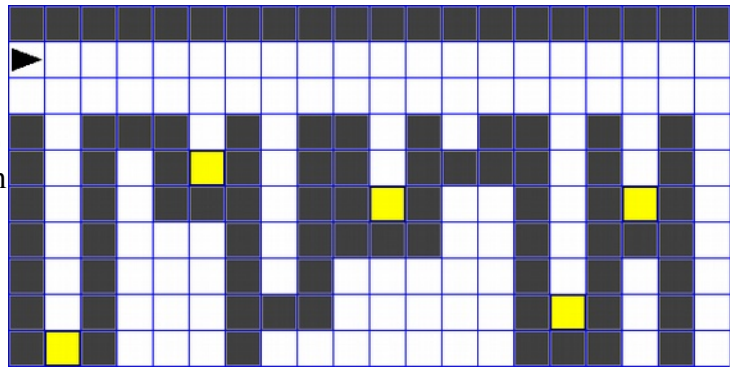


Anw.: WEGFINDEN




```
// --- Anweisungen -------
Anweisung MauerRäumen
  solange IstZiegel tue
    Aufheben
  *solange
  Schritt
  linksdrehen
  wenn IstWand dann
    rechtsdrehen
  *wenn
*Anweisung
// ------------------------
Anweisung WegFinden
    linksdrehen
    wenn IstWand dann
      RechtsDrehen
      RechtsDrehen
      Wenn IstWand dann
        WegFinden
      *wenn
    *wenn
*Anweisung
// === Hauptprogramm =====
solange NichtIstMarke tue
  MauerRäumen
  wenn IstWand dann
    Wegfinden
  *wenn
*solange
```

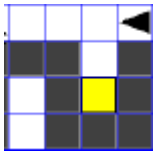
19: Karol in der Kanalisation



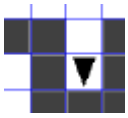
Karol befindet sich in einem Abwasserkanal mit zahlreichen Nebenarmen. Einige davon haben sich im Lauf der Zeit mit Schmutz (Marke) zugesetzt. Karol hat die Aufgabe sie zu reinigen Anschließend soll er sich wieder zu seinem Ausgangspunkt begeben.

Welt: Haupt- und Nebenkanäle
maximal 2 Ziegel breit
Weltdateien: 19Kanalisation.kdw
Lösungen: 19Kanalisation.kdp

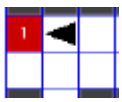
Befehle:	`Schritt, Hinlegen, Aufheben, Linksdrehen, Rechtsdrehen`
Bedingungen:	`NichtIstMarke, NichtIstWand, IstWand`
Kontrollstrukturen:	`Solang ... tue` `...` `*solange` — `wenn ... dann` `...` `*wenn` — `Anweisung ...` `...` `*Anweisung` — *Verschachtelt:* *wenn ...* *wenn ...*

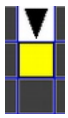


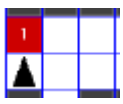
Noch kein Seitenkanal, aber beim nächsten Schritt!



Karol steht vor Wand und muss sich nach rechts drehen und anschließend gleich noch einmal, um den Seitenkanal wieder zu verlassen.

Vor dem Start: 

Klären: 

Am Ende des Klärgangs: 

// Verwendete Anweisung

```
Anweisung IstDaSeitenKanal
  LinksDrehen
  wenn IstWand dann
    RechtsDrehen
  *wenn
*Anweisung
```

Karol schaut nach links, ob hier ein Seitenkanal abzweigt.
Wenn das nicht der Fall ist (IstWand), dreht er sich wieder nach rechts.

```
Anweisung Klären
  Schnell
  wenn NichtIstWand dann
    Schritt
    wenn IstMarke dann
      MarkeLöschen
    *wenn
  sonst
    RechtsDrehen
  *wenn
  IstDaSeitenKanal
*Anweisung
```

Wenn keine Wand den Weg versperrt, macht Karol einen Schritt und löscht die Marke, falls er jetzt auf einer steht.
sonst weicht er der Wand mit einem Rechtsdreh aus

Jetzt schaut er nach, ob er in einen Seitenkanal abbiegen muss.

// === Hauptprogramm =====

```
Klären
rechtsdrehen
rechtsdrehen
hinlegen
linksdrehen
linksdrehen

solange NichtIstZiegel tue
  Klären
*solange

Aufheben
```

Karol klärt das Feld vor ihm, dreht sich um, legt einen Ziegel ab, damit er weiß, wann er wieder am Ausgang ist, und dreht sich wieder in die alte Rchtung

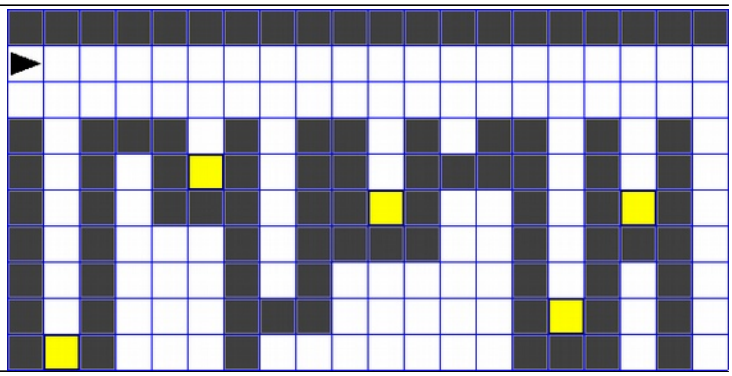
Solange er den Ziegel am Ausgang nicht wieder erreicht hat, klärt er ein Feld nach dem anderen.

Am Schluss hebt er seinen Ziegel wieder auf.

Struktogramm und Befehlsliste 19

19Kanalisation.kdp

Karol befindet sich in einem Abwasserkanal mit zahlreichen Nebenarmen. Einige davon haben sich im Lauf der Zeit mit Schmutz (Marke) zugesetzt. Karol hat die Aufgabe sie zu reinigen Anschließend soll er sich wieder zu seinem Ausgangspunkt begeben.



Hauptprogramm

KLÄREN
RechtsDrehen
RechtsDrehen
Hinlegen
LinksDrehen
LinksDrehen
solange NichtIstZiegel
KLÄREN
Aufheben
KLÄREN

Anw.: ISTDASEITENKANAL

LinksDrehen	
IstWand	
w	**f**
RechtsDrehen	

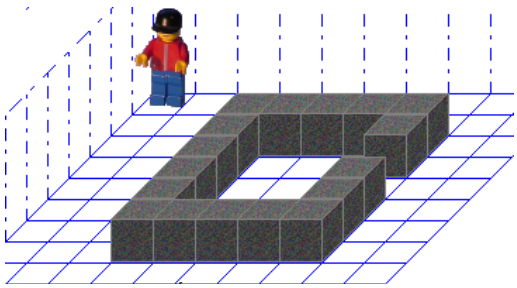
Anw.: KLÄREN

NichtIstWand	
w	**f**
Schritt	RechtsDrehen
IstMarke	
w	**f**
MarkeLöschen	
ISTDASEITENKANAL	

```
// --- Anweisungen -------
Anweisung IstDaSeitenKanal
  LinksDrehen
  wenn IstWand dann
    RechtsDrehen
  *wenn
*Anweisung
// ------------------------
Anweisung Klären
  wenn NichtIstWand dann
    Schritt
    wenn IstMarke dann
      MarkeLöschen
    *wenn
  sonst
    RechtsDrehen
  *wenn
  IstDaSeitenKanal
*Anweisung

// === Hauptprogramm =====
Klären
RechtsDrehen Rechtsdrehen
Hinlegen
LinksDrehen  Linksdrehen

solange NichtIstZiegel tue
  Klären
*solange

Aufheben
Klären
```

Fach:	Name:	Datum:

20: Karol sucht das Scheunentor



Der Bauer das Stadeltor nicht geschlossen. Karol soll es mit einem Ziegel verschließen. Das Stadeltor ist einen Ziegel breit und um den Stadel kann man herumlaufen.

Welt: Größe beliebig
Weltdateien:20stadeltor1.kdw, 20stadeltor2.kdw
Lösungen: 20stadeltor.kdp

// Verwendete Anweisung

```
Bedingung KeineStadelEcke
  wahr
  wenn IstWand dann
    wenn IstSüden dann
      RechtsDrehen Schritt LinksDrehen
      wenn NichtIstWand dann
        falsch
      sonst
        LinksDrehen Schritt Schritt
        LinksDrehen
      *wenn
    sonst
      LinksDrehen Schritt RechtsDrehen
      wenn NichtIstWand dann
        falsch
      sonst
        RechtsDrehen Schritt Schritt
        RechtsDrehen
      *wenn
    *wenn
  *wenn
*Bedingung
Bedingung KeinStadeltor
  wahr
  Linksdrehen
  wenn IstWand dann
    rechtsdrehen
  sonst
     schritt
     rechtsdrehen
     wenn IstWand dann
       falsch
     sonst
       linksdrehen
     *wenn
  *wenn
*Bedingung
```

Karol sagt: Keine Stadelecke (wahr)

Falls er vor einer Wand steht , prüft er, ob er nach Sünden schaut.

Er schaut, ob rechts daneben keine Wand ist

?

// Hauptprogramm

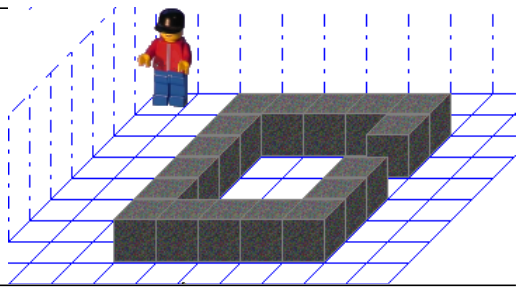
```
LinksDrehen Schritt RechtsDrehen
solange keineStadelEcke tue
  Schritt
*solange
Wenn IstNorden dann
  Rechtsdrehen
*wenn
Schritt
solange keinStadelTor tue
  Schritt
*solange
```

Fach:	Name:	Datum:

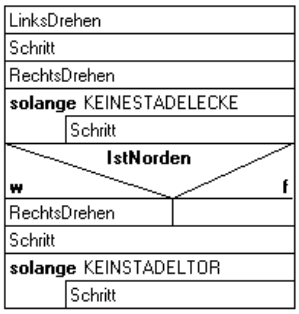
Struktogramm und Befehlsliste 20

10Zimmerausgang.kdp

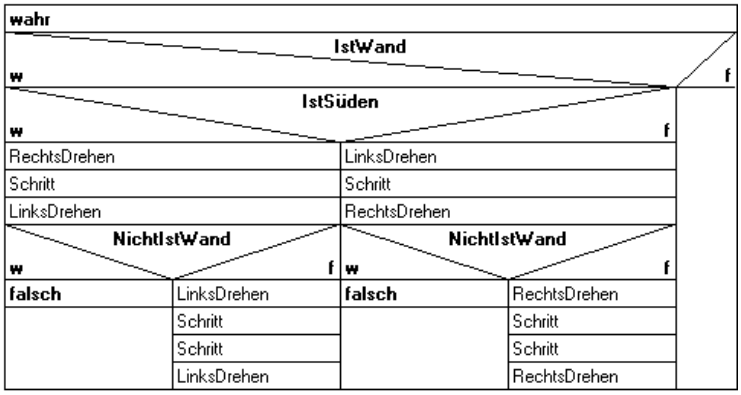
Der Bauer das Stadeltor nicht geschlossen. Karol soll es mit einem Ziegel verschließen. Das Stadeltor ist einen Ziegel breit und um den Stadel kann man herumlaufen.



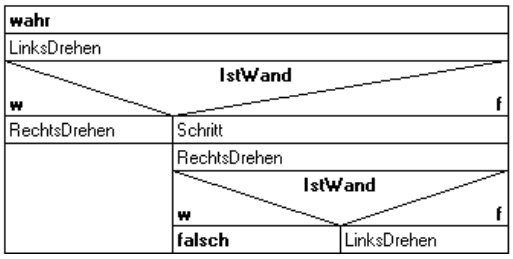
Hauptprogramm



Bed.: KEINESTADELECKE



Bed.: KEINSTADELTOR

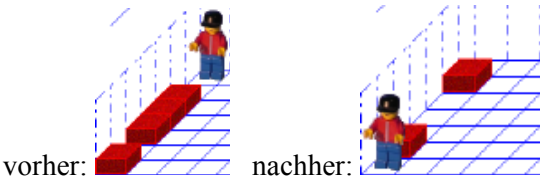



```
Bedingung KeineStadelEcke
  wahr
  wenn IstWand dann
    wenn IstSüden dann
      RechtsDrehen Schritt
LinksDrehen
      wenn NichtIstWand dann
        falsch
      sonst
        LinksDrehen Schritt
Schritt LinksDrehen
      *wenn
    sonst
      LinksDrehen Schritt
RechtsDrehen
      wenn NichtIstWand dann
        falsch
      sonst
        RechtsDrehen Schritt
Schritt RechtsDrehen
      *wenn
    *wenn
  *wenn
*Bedingung
// ------------------------------
Bedingung KeinStadeltor
  wahr
  Linksdrehen
  wenn IstWand dann
    rechtsdrehen
  sonst
     schritt
     rechtsdrehen
     wenn IstWand dann
       falsch
     sonst
       linksdrehen
     *wenn
  *wenn
*Bedingung
// === Hauptprogramm ==========
LinksDrehen Schritt RechtsDrehen
solange keineStadelEcke tue
  Schritt
*solange
Wenn IstNorden dann
  Rechtsdrehen
*wenn
Schritt
solange keinStadelTor tue
  Schritt
*solange
```

21: Weitere Aufgaben

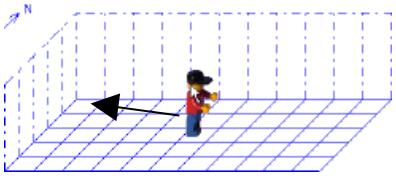
1 Invertieren

Karol steht vor einer Ziegelreihe. Wo vorher kein Ziegel war soll jetzt einer liegen, Felder, die Vorher Ziegel enthielten, sollen abgeräumt werden. Das Startfeld nicht vergessen!



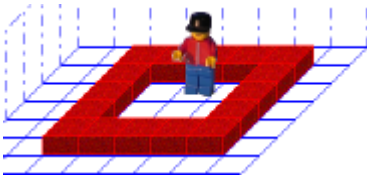
2 In die Nordwest-Ecke zurückkehren

Karol findet aus jeder beliebigen Position in seiner Welt in die Nordwest-Ecke zurück, wenn kein Hindernis mehr (Quader, zu hohe Ziegelstapel) seinen Weg verspert



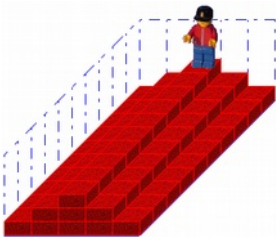
3 Rechteck füllen

Karol steht in einem Rechteck, dessen Rand durch Ziegel markiert ist. Er soll auf jedes Feld innerhalb des Rechtecks einen Ziegel legen. Dies soll er von einem beliebigen Startplatz in der Mitte des Rechtecks erledigen!



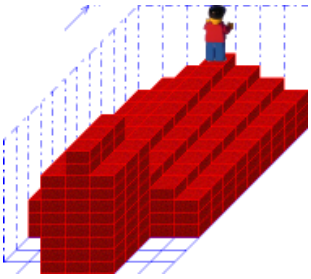
4 Dach

Der Roboter Karol soll das unten abgebildete Dach bauen (Welt 10 x 5 Ziegel). Erstelle für ihn dazu eine Liste von Anweisungen. Gehe dabei möglichst geschickt vor.



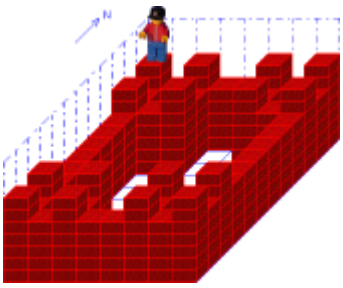
5 Kirche

Karol soll die Kirche rechts bauen. Die Länge soll 12 und die Breite der Welt soll 7 Ziegel betragen, die Höhe 10 Ziegel.



6 Burg

Karol soll die Burg rechts bauen. Die Länge und Breite der Welt soll mindestesn 7 Ziegel betragen, die Höhe 10 Ziegel.



7 Labyrinth

Karol soll eigenständig den Weg aus einem Labyrinth finden. Im Labyrinth kann man nicht im Kreis laufen. Das Ziel ist mit einer Marke gekennzeichnet. Wenn Karol immer links an der Wand lang marschiert, findet er trotz einiger Umwege zum markierten Ausgang.