

2 Karol lernt einfache Tätigkeiten

Bisher hast du Karol im Direktmodus durch seine Karol-Welt bewegt mit Hilfe der blauen Pfeiltasten und den entsprechenden Schaltflächen. Jede Anweisung an Karol wurde von ihm sofort befolgt.

Nun wirst du lernen, wie man zuerst eine Sammlung von Anweisungen erstellt und Karol diese erst auf Abruf abarbeitet. Man sagt auch, du erstellst ein Programm.

Bemerkung:

Du wirst nun viele Programme erstellen. Damit du den Überblick bewahrst, solltest du einen Ordner *ProgrammeKarol* anlegen, in den du die Dateien sammelst. Auch die Namen deiner Programme sollten ein bestimmtes Muster aufweisen. Das Programm aus Übung 2.1 nennst beispielsweise *201UForm*. So weißt, dass in der Übung 2.1 Karol ein U abschreiten musste.

2.1 Karol geht im Kreis

Übung 2.1:

Karol soll eine U-Form der Länge 4 und der Breite 3 abschreiten.

Implementiere zu zugehörigen Algorithmus. Teste dein Programm. Speichere anschließend das Programm und die Welt unter dem Namen *201UForm*.

Algorithmus umgangssprachlich	Algorithmus als Karolprogramm
4-mal vorwärts	Schritt Schritt Schritt Schritt
linksum	LinksDrehen
3-mal vorwärts	Schritt Schritt Schritt
linksum	LinksDrehen
4-mal vorwärts	Schritt Schritt Schritt Schritt

Anstatt Karol deine Befehle über die Steuerungstasten zu geben, sollst du nun links im Editorfenster Programmbefehle zur Steuerung eingeben.

Dazu hast du folgende Möglichkeiten:

- a) Direkte Eingabe der Anweisung (z. B. Schritt, LinksDrehen, RechtsDrehen, Hinlegen, usw.) über die Tastatur.
- b) Öffnen eines Kontextmenüs mit der rechten Maustaste und Auswahl der vordefinierten Anweisungen (z. B. Aufheben, MarkeSetzen....) in der Mitte des Menüs

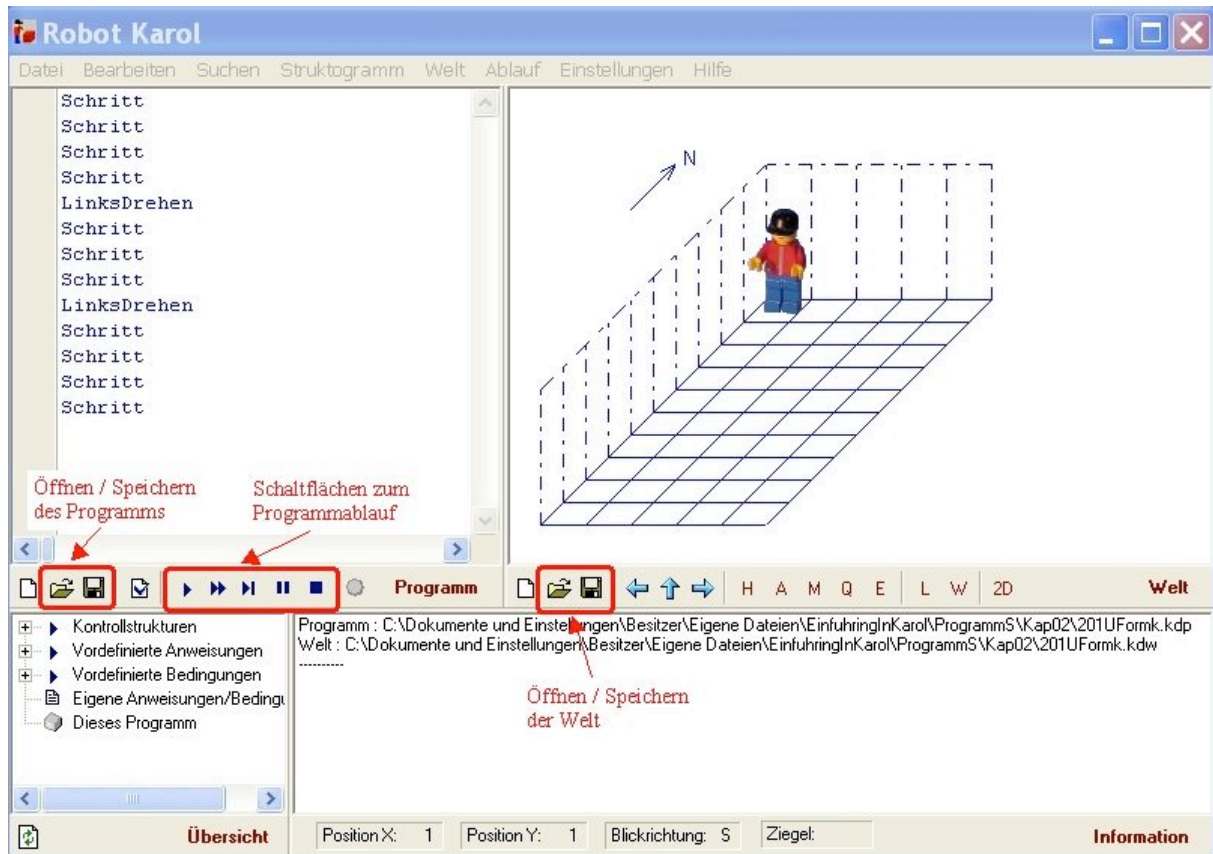


Abbildung 2.1: Karol läuft eine U-Form

In Abbildung 2.1 zeigt zusätzlich eingerahmt die Schaltflächen für

1. Start, Schnelldurchlauf, Einzelschritt. Pause und Abbruch
2. Speichern bzw. Öffnen des Programms, beispielsweise *201UForm.kdp*
3. Speichern bzw. Öffnen der Welt, beispielsweise *201UForm.kdw*

Bemerkung:

In dem gewählten Ordner befinden sich nun die beiden Dateien *201UForm.kdp* („Karol deutsch Programm“) und *201UForm.kdw* („Karol deutsch Welt“). Es ist sinnvoll, das Programm und die zugehörige Welt unter dem gleichen Namen zu speichern.

Übung 2.2

Erzeuge das in Abbildung 2.2 dargestellte Rechteck. Hierzu setzt Karol an jeder Ecke des Rechtecks eine Marke. Speichere anschließend dein Programm und auch die zugehörige Anfangswelt unter dem Namen *202Rechteck*.

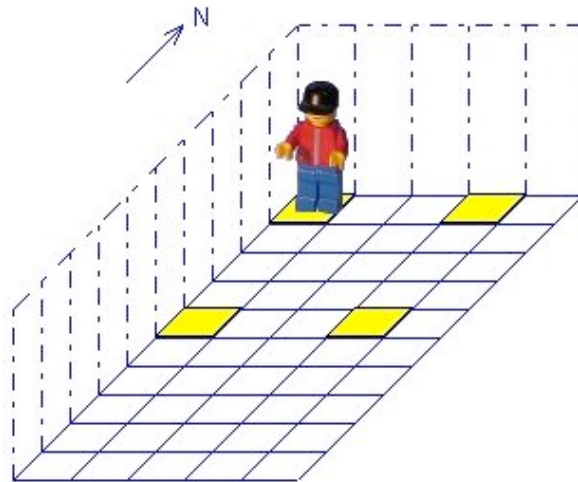


Abbildung 2.2: Karol markiert ein Rechteck

2.2 Karol baut Ziegelreihen

Übung 2.3:

- Karol baut eine gerade Reihe aus 6 Ziegelsteinen. (*203aReihe*)
- Karol baut eine Zickzackreihe aus 6 Ziegelsteinen. (*203bZickzack*)

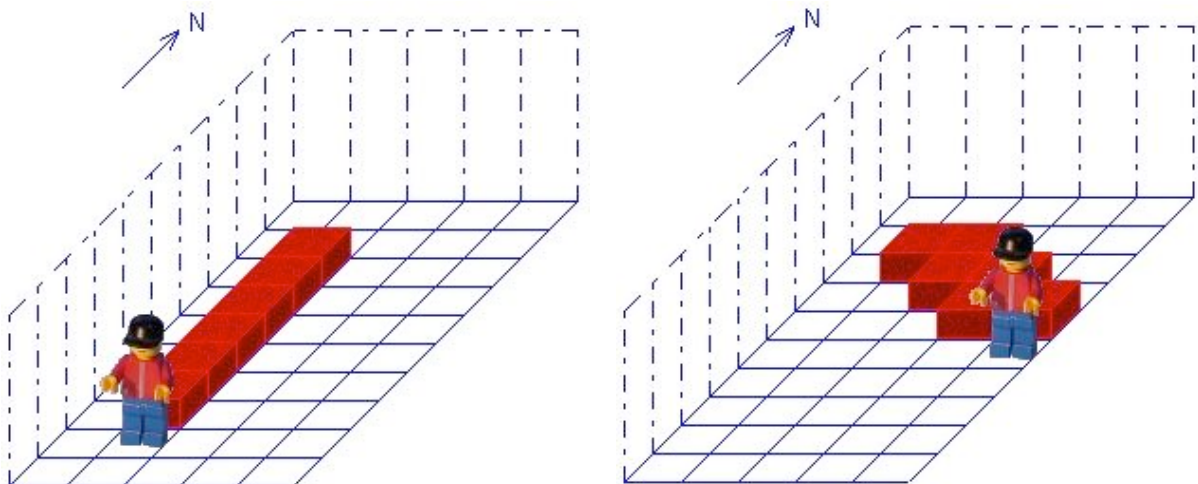


Abbildung 2.3: Karol baut gerade Reihe und Zickzackreihe

Übung 2.4:

Programmiere einen Algorithmus, so dass Karol das in Abbildung 2.4 dargestellte Siegerpodest baut. Baue das Podest auch in Ost-West-Richtung. (*204Podest*)

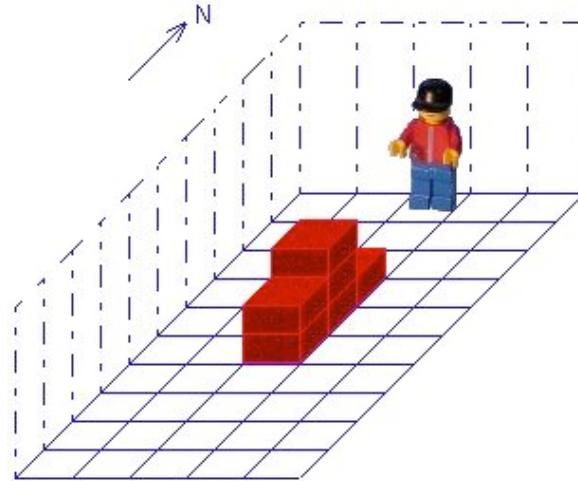


Abbildung 2.4: Karol baut ein Siegerpodest

Übung 2.5:

Karol soll aus Ziegelsteinen ein Quadrat der Kantenlänge 3 legen. (*205Quadrat*)

Übung 2.6:

Karol soll den ersten Buchstaben deines Vornamen mit Hilfe der Ziegelsteine schreiben. (*206Buchstabe*)

2.3 Karol kann lernen

Das Kontextmenü zeigt dir, welche Fähigkeiten Karol beherrscht. Aber Karol kann auch weitere Fähigkeiten lernen, indem du der Klasse *Roboter* weitere Methoden hinzufügst.

Übung 2.7:

Karol soll 3 Schritte gehen, eine Marke setzen, sich umdrehen, wieder 3 Schritte gehen und anschließend sich wieder umdrehen.

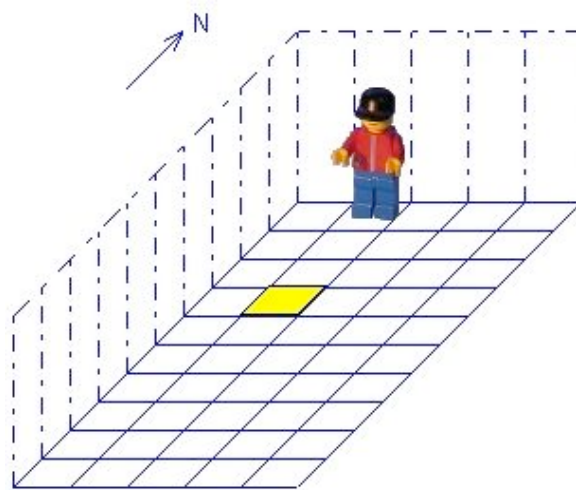


Abbildung 2.5: Karol lernt neue Fähigkeiten

Schritt	
Schritt	Riesenschritt
Schritt	
Markesetzen	MarkeSetzen
LinksDrehen	Umdrehen
LinksDrehen	
Schritt	
Schritt	Riesenschritt
Schritt	
LinksDrehen	Umdrehen
LinksDrehen	

Du könntest dir viel Schreibarbeit ersparen, wenn die drei Methoden *Schritt* zu einer Anweisung *Riesenschritt* zusammengefasst werden. Ebenso könntest du die beiden Methoden *LinksDrehen* zusammenlegen zur Anweisung *Umdrehen*. Karol muss also die beiden neuen Fähigkeiten *Riesenschritt* und *Umdrehen* lernen, du musst also zusätzlich neue Methode für die Klasse *Roboter* programmieren.

Die Methode *Riesenschritt* zeigt die Abbildung 2.6:

```
Anweisung Riesenschritt
  Schritt
  Schritt
  Schritt
*Anweisung

Programm
  Riesenschritt
  MarkeSetzen
  LinksDrehen
  LinksDrehen
  Riesenschritt
  LinksDrehen
  LinksDrehen
*Programm
```

Abbildung 2.6: Die Methode *Riesenschritt*

Die Methode *Riesenschritt* beginnt mit dem Wort *Anweisung* und dann mit dem Bezeichner der Methode. Anschließend folgt die Sequenz von Anweisungen, die beim Aufruf der Methode abgearbeitet werden. Die Festlegung der Methode endet mit dem Schlüsselwort **Anweisung*.

Um den Programmhauptteil klar von den selbstdefinierten Methoden abzugrenzen, empfehle ich dir die Schlüsselwörter *Programm* und **Programm*.

Übersichtlich wird der Programmtext gestaltet, wenn du die Methoden etwas einrückst. Die Programmierumgebung hilft dir dabei, wenn du den Menüpunkt *Bearbeiten > Formatieren* verwendest.

Du hast zwar jetzt die Methode *Riesenschritt* programmiert, aber leider macht Karol in seiner Welt immer noch drei Schritte hintereinander und nicht einen Riesenschritt. Deswegen könntest du, wie in Abbildung 2.7 dargestellt, noch das Schlüsselwort *Schnell* in die Methode *Riesenschritt* einfügen. Nun springt Karol wirklich einen Riesenschritt vorwärts.

```
Anweisung Riesenschritt
  Schnell
  Schritt
  Schritt
  Schritt
*Anweisung
```

Abbildung 2.7: Karol springt nun einen Riesenschritt

Übung 2.8:

Programmiere nun auch die Methode *Umdrehen*. Teste dein Programm.
(208Umdrehen)

Übung 2.9:

Bringe Karol den Rösselsprung bei, wie ihn die Springerfigur beim Schach ausführt. Erstelle ein Programm, so dass Karol mehrere Rösselsprünge ausführt und nach jedem Sprung eine Marke setzt. (*209Roesselsprung*)

Übung 2.10:

Bringe Karol das Rückwärtsgehen bei. (*210Rueckwaerts*)

Übung 2.11:

Karol legt hinter sich eine Reihe von Ziegelsteinen. Verwende möglichst viele selbstdefinierte Methoden. Vergleiche dazu auch Abbildung 2.8! (*211ZiegelreiheHinten*)

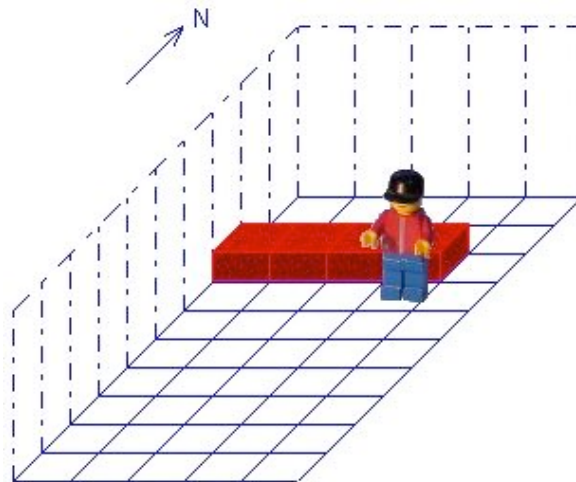


Abbildung 2.8: Karol legt hinter sich eine Ziegelreihe

Übung 2.12:

Schreibe zunächst eine Anweisung *LegeRechts*, die rechts neben Karol einen Ziegel hinlegt. Karol soll diese Anweisung dazu verwenden, beim Vorwärtsgehen rechts neben sich eine Mauer zu bauen. (*212ZiegelreiheRechts*)

Übung 2.13

Schreibe eine Anweisung *MauerRechts*, welche die Mauer aus Übung 2.12 errichtet. Die selbstdefinierte Anweisung *LegeRechts* kann darin verwendet werden. Karol soll die Anweisung *MauerRechts* verwenden, sich selbst in einem Mauerviereck einzusperren. (*213Gefaengnis*)