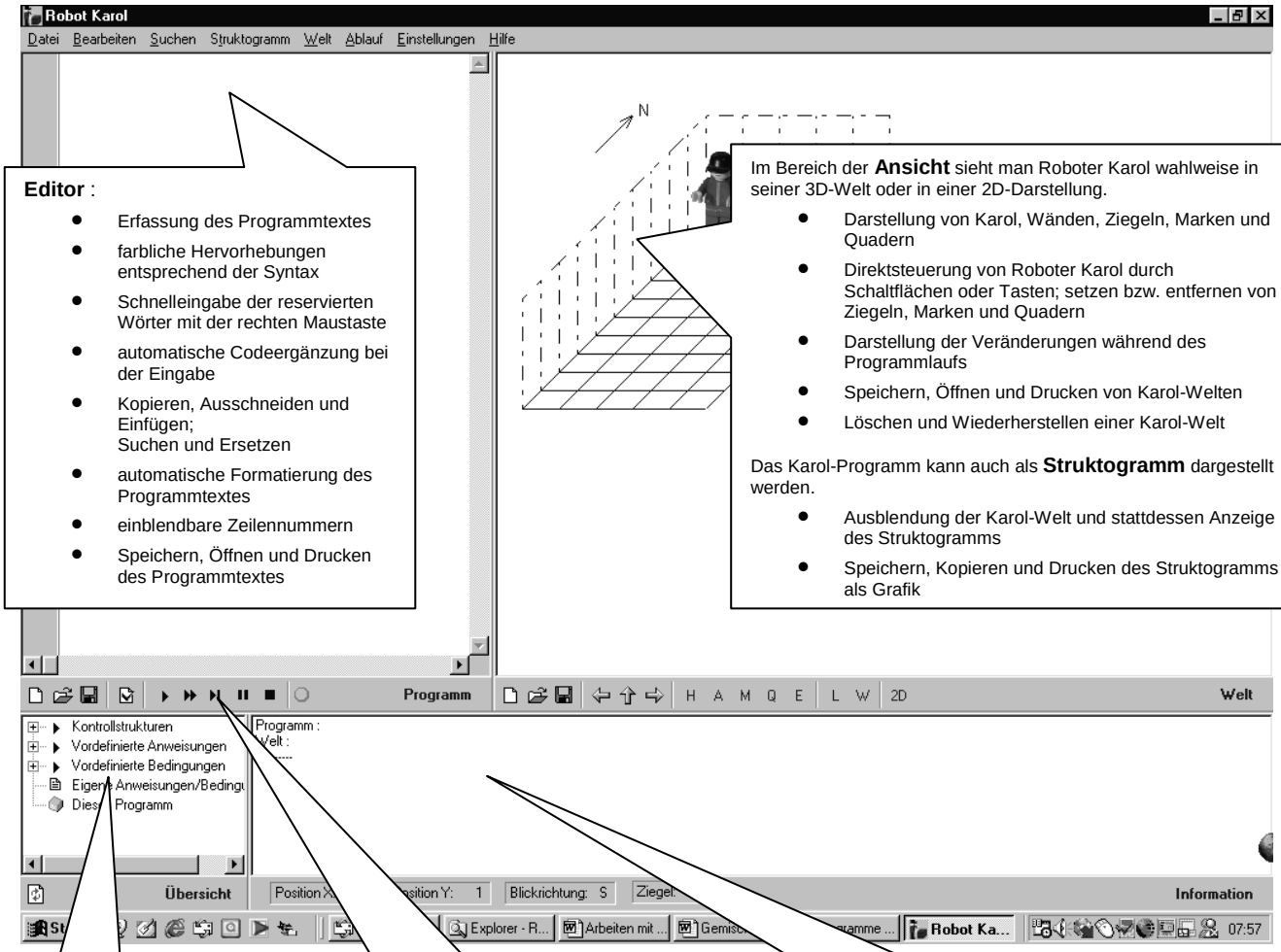
Robot Karol - Ein Programm zum Erlernen elementarer Programmierlogik



Editor :

- Erfassung des Programmtextes
- farbliche Hervorhebungen entsprechend der Syntax
- Schnelleingabe der reservierten Wörter mit der rechten Maustaste
- automatische Codeergänzung bei der Eingabe
- Kopieren, Ausschneiden und Einfügen;
  Suchen und Ersetzen
- automatische Formatierung des Programmtextes
- einblendbare Zeilennummern
- Speichern, Öffnen und Drucken des Programmtextes

Im Bereich der **Ansicht** sieht man Roboter Karol wahlweise in seiner 3D-Welt oder in einer 2D-Darstellung.

- Darstellung von Karol, Wänden, Ziegeln, Marken und Quadern
- Direktsteuerung von Roboter Karol durch Schaltflächen oder Tasten; setzen bzw. entfernen von Ziegeln, Marken und Quadern
- Darstellung der Veränderungen während des Programmlaufs
- Speichern, Öffnen und Drucken von Karol-Welten
- Löschen und Wiederherstellen einer Karol-Welt

Das Karol-Programm kann auch als **Struktogramm** dargestellt werden.

- Ausblendung der Karol-Welt und stattdessen Anzeige des Struktogramms
- Speichern, Kopieren und Drucken des Struktogramms als Grafik

Übersicht

- Auflistung aller Kontrollstrukturen
- Auflistung aller vordefinierten Anweisungen und Bedingung
- Auflistung der selbstdefinierten Anweisungen und Bedingungen
- Darstellung des Programms in Form einer Baumstruktur

Programmablauf :

- Syntaxprüfung mit Fehlerhinweis und Markierung der betroffenen Zeile; Aufruf vor jedem Programmablauf oder gesondert
- Ablauf mit Verzögerung; nach jeder Anweisung wartet Karol eine einstellbare Zeit
- Schnellablauf ohne Verzögerung
- manueller Einzelschritt
- Pause und Programmstopp jederzeit möglich
- Setzen von Stopppunkten im Programmtext möglich

Informationsfläche

- Einblendung der aktuellen Karol-Position und der Karol-Blickrichtung
- auf Wunsch Einblendung der Ziegelanzahl die Karol momentan trägt
- Anzeige der Dateinamen von Programm und Welt
- Ausgabe von Fehlermeldungen, sowohl Syntaxfehler als auch Laufzeitfehler

Vordefinierte Methoden: (in der Klammer kann die Anzahl der Wiederholungen eingegeben werden)

Schritt( )	Karol macht, wenn erlaubt, einen Schritt in die Richtung, in die er schaut; auch mehrere Schritte sind mit "Schritt(Anzahl)" möglich
LinksDrehen( )	Karol dreht sich nach links (um 90°)
RechtsDrehen( )	Karol dreht sich nach rechts (um 90°)
HinLegen( )	Karol legt einen Ziegel vor sich hin
AufHeben( )	Karol hebt einen Ziegel auf, der vor ihm liegt
MarkeSetzen( )	Karol setzt eine Marke an die Stelle, auf der er steht
MarkeLöschen( )	Karol entfernt eine Marke von der Stelle, auf der er steht
Warten( )	Karol wartet eine Sekunde; auch "Warten(Millisekunden)" ist möglich
Ton( )	Karol gibt einen Ton von sich

Vordefinierte Methoden mit logischen Rückgabewerten:

IstWand	WAHR, wenn Karol vor einer Wand oder einem Quader steht
NichtIstWand	WAHR, wenn IstWand nicht zutrifft
IstZiegel	WAHR, wenn Karol vor einem (oder mehreren) Ziegel steht und zu ihm schaut; die Abfrage nach einer genauen Anzahl an Ziegeln ist auch möglich mit "IstZiegel(Anzahl)"
NichtIstZiegel	WAHR, wenn IstZiegel nicht zutrifft
IstMarke	WAHR, wenn Karol auf einer Marke steht
NichtIstMarke	WAHR, wenn IstMarke nicht zutrifft
IstSüden, IstNorden, IstWesten, IstOsten	WAHR, wenn Karol in diese Richtung schaut
IstVoll	WAHR, wenn die Überwachung der Tragfähigkeit von Karol eingeschalten ist und der maximale Tragumfang erreicht ist. (siehe Programmierumgebung-Einstellungen)
NichtIstVoll	WAHR, wenn IstVoll nicht zutrifft
IstLeer	WAHR, wenn die Überwachung der Tragfähigkeit von Karol eingeschalten ist und Karol gerade keinen Ziegel transportiert. (siehe Programmierumgebung-Einstellungen)
NichtIstLeer	WAHR, wenn IstLeer nicht zutrifft

Kontrollstrukturen

Einseitig bedingte Anweisung:

```
wenn [nicht] <Bedingung> dann
        <Anweisung>
            ...
        <Anweisung>
*wenn
```

Zweiseitig bedingte Anweisung:

```
wenn [nicht] <Bedingung> dann
        <Anweisung>
            ...
        <Anweisung>
sonst
        <Anweisung>
            ...
        <Anweisung>
*wenn
```

Wiederholung mit fester Anzahl:

```
wiederhole <x> mal
        <Anweisung>
            ...
        <Anweisung>
*wiederhole
```

Wiederholung mit Anfangsbedingung:

```
wiederhole solange [nicht] <Bedingung>
        <Anweisung>
            ...
        <Anweisung>
*wiederhole
```

Wiederholung mit Endbedingung:

```
wiederhole
        <Anweisung>
            ...
        <Anweisung>
*wiederhole solange [nicht] <Bedingung>
```

Endlose Wiederholung:

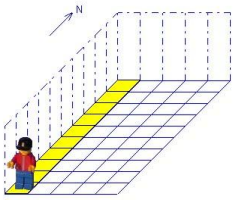
```
wiederhole immer
        <Anweisung>
            ...
        <Anweisung>
*wiederhole
```

Programmierregeln:

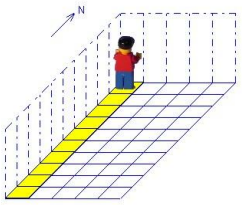
- Plane Dein Programm mit „Stift und Zettel“ (wichtig bei komplexen Aufgaben)
- Programmiere übersichtlich.
- Schleifen müssen geschlossen werden.
- Anweisungen, ... innerhalb von Schliefen werden eingerückt geschrieben (Übersichtlichkeit)
- Ein Programm darf nicht mit einer Fehlermeldung enden, auch wenn die Aufgabe erfüllt wurde
- Die Endversion eines Programms muss in jeder „Welt“ ohne Programmänderung lauffähig sein.
- Speichere jede Programmversion; z.B.: Aufgabe_1_v1.kdp (*v1.* läuft nur in Standardwelt)
- Aufgabe_1_v2.kdp (*v2.* läuft in jeder Welt)

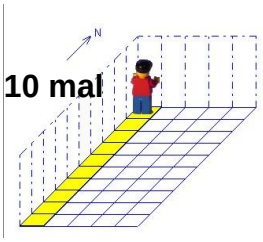
Jede Aufgabe muss so gelöst werden, dass sie in der „Standard“-Welt ohne Fehlermeldung erfüllt wird (*v1.*). Danach allgemeine Lösung suchen, testen und speichern (*v2.*)

Aufgabe 1: Der Roboter Karol soll bis an die Wand gehen und auf jedes Feld eine Marke setzen.



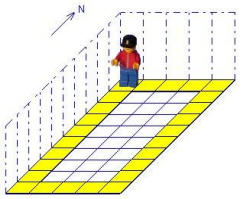
Aufgabe 2: Der Roboter Karol soll bis an die Wand und wieder zurück gehen (beim Hinweg Marken setzen).

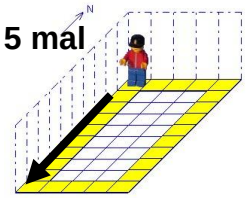


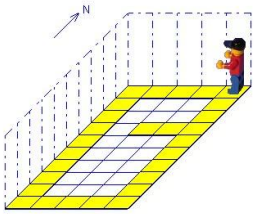


Aufgabe 3: Der Roboter Karol soll 10 mal bis an die Wand und zurück gehen.

Aufgabe 4: Der Roboter Karol soll einmal außen herum gehen und dabei Marken setzen.



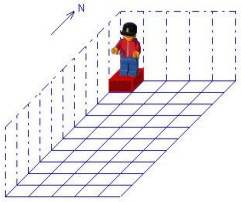




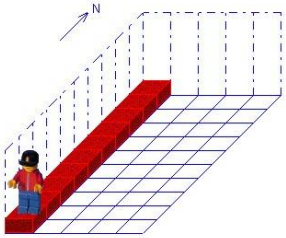
Aufgabe 5: Der Roboter Karol soll 5 mal außen herum gehen.

Aufgabe 6: Der Roboter Karol soll einmal außen herum gehen, egal von wo er startet. Laufweg mit Marken markieren.

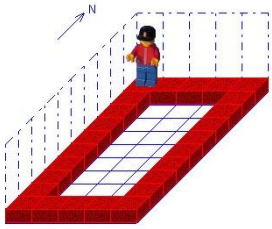
Aufgabe 7: Der Roboter Karol soll einen Ziegel unter sich legen.

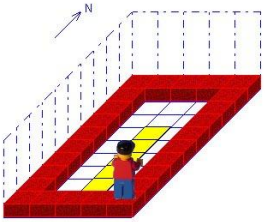


Aufgabe 8: Der Roboter Karol soll an der Wand eine Reihe Ziegel legen.



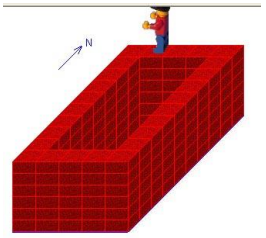
Aufgabe 9: Der Roboter Karol soll eine Reihe Ziegel außen herum legen.



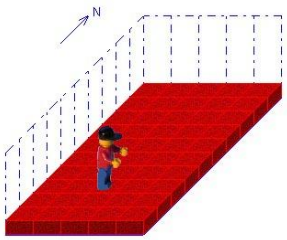


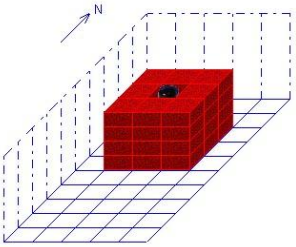
Aufgabe 10: Der Roboter Karol soll eine Reihe Ziegel außen herum legen, egal von wo er startet und den Laufweg ohne Ziegel mit Markierungen versehen.

Aufgabe 11: Der Roboter Karol soll außen herum eine Mauer bauen (bis zur Höhe seiner Welt).



Aufgabe 12: Der Roboter Karol soll eine ganze Lage Ziegel legen.





Aufgabe 13: Der Roboter Karol soll sich einmauern. Er darf sich dabei nicht auf Ziegel stellen.