Jugendwettbewerb Informatik: Algorithmus und Kontrollstrukturen

Algorithmus

Auch in unserem Alltag finden wir fast überall verschiedene Folgen von Anweisungen, sei es beim Rezept zum Kuchen backen, beim Montieren von Schränken, beim Basteln oder beim Wäsche waschen. Diese Anweisungen beschreiben uns Schritt für Schritt den genauen Ablauf von bestimmten Tätigkeiten und geben uns eine Reihenfolge vor, in der wir einzelne Aufgaben erledigen müssen.
All diese Anleitungen führen wir aus, um am Ende ein gewünschtes Ergebnis zu erhalten. Solche Folgen von Anweisungen nennt man **Algorithmen**.

Merke (Algorithmus)

Ein **Algorithmus** ist eine **endliche** Folge von **eindeutigen** und **ausführbaren** Anweisungen zur Lösung eines Problems.
Unter gleichen Voraussetzungen liefert die Ausführung eines Algorithmus stets das gleiche Ergebnis.

Ein Teilgebiet der Informatik beschäftigt sich mit dem **Programmieren**. Letztendlich liegt jedem Programm, das ein Programmierer erstellt, ein **Algorithmus** zugrunde. Es gilt hierbei ein Problem mithilfe von Programmierbausteinen durch logisches Denken zu lösen. In diesem Abschnitt lernst du mithilfe einer bausteinorientierten Programmiersprache, selbstständig Programme zu entwerfen, die gegebene Problemstellungen lösen.

Anweisung, Sequenz und Programm

Merke (Anweisung, Sequenz und Programm)

Der Roboter wird durch **Anweisungen** gesteuert, die man als Methodenaufrufe betrachten kann.

Eine **Sequenz** ist eine Folge von Anweisungen, bei deren **Ablauf** diese in der angegebenen Reihenfolge abgearbeitet werden.

Ein **Programm** ist eine Folge von Anweisungen in einer **Programmiersprache**, die von einem Gerät ausgeführt werden können.

Merke (Struktogramme)

Zur graphischen Darstellung eines Programmablaufs verwendet man sogenannte **Struktogramme**. Damit lassen sich Programmabläufe unabhängig von konkreten Programmiersprachen übersichtlich darstellen.

Für eine **Sequenz** sieht das Struktogramm folgendermaßen aus.

Anweisung 1
Anweisung 2
Anweisung 3
...

Schleifen I: Wiederholung mit fester Anzahl

Im vorherigen Programm haben wir gesehen, dass unser Algorithmus mit so vielen Anweisungen sehr lang und unübersichtlich wird. Wir haben außerdem immer wieder die gleichen Anweisungen verwendet. Einfacher wäre es doch, wenn wir unserem Programm sagen könnten, dass es eine ganze Sequenz einfach z.B. 10 mal wiederholen soll. Dazu gibt es in der Algorithmik die Kontrollstruktur **Wiederholung mit fester Anzahl**. Wir können so ganze Sequenzen wiederholen, ohne sie jedes mal wieder neu schreiben zu müssen.

Merke (Wiederholung mit fester Anzahl)

Die Kontrollstruktur **Wiederholung mit fester Anzahl** dient dazu, mehrfache Wiederholungen von Anweisungen oder ganzen Sequenzen einfach zu realisieren. Die Anzahl der Wiederholungen steht dabei von Anfang an bereits fest.

Wiederholung mit fester Anzahl

Struktogramm:

wiederhole x-mal
Sequenz

Bedingte Anweisung I: einseitig bedingte Anweisung

Siehe nächste Seite.

Bisher sind wir immer davon ausgegangen, dass Sequenzen beliebig oft ausgeführt werden und erst bei Eintreten einer Bedingung beendet werden. Oftmals ist es aber gewünscht, dass Anweisungen oder Sequenzen nur dann ein einziges mal ausgeführt werden, wenn zuvor ein bestimmtes Ereignis eingetreten ist. Dies lässt sich mit der „`bedingten Anweisung`“ realisieren.

Merke (Bedingte Anweisung)

Die Kontrollstruktur bedingte Anweisung dient dazu, eine Sequenz abhängig von einer Bedingung auszuführen.

Bedingte Anweisung

Struktogramm:

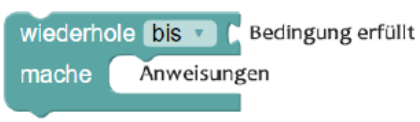
Bedingung erfüllt?	
WAHR	FALSCH
Sequenz	∅

Schleifen II: Wiederholung mit Bedingung

Bisher wussten wir bei Wiederholungen bereits deren genaue Anzahl. Meistens ist das aber nicht der Fall. Stattdessen sollen Wiederholungen solange ausgeführt werden, bis irgendein Ereignis eintritt. Dann soll die Wiederholung gestoppt werden. Dieses Konzept nennt man Wiederholung mit Bedingung.

Merke (Bedingte Wiederholung)

Die Kontrollstruktur bedingte Wiederholung wird verwendet, um eine Sequenz abhängig von einer Bedingung auszuführen. Sie wird dazu verwendet, Anweisungen oder ganze Sequenzen zu wiederholen, ohne dass zuvor die Anzahl der Wiederholungen bekannt sein muss.
Die Anweisungen werden dann solange ausgeführt, bis eine Abbruchbedingung eintritt (z.B. eine gedrückte Taste, ein Hindernis, usw.).



Bedingte Wiederholung

Struktogramm:

wiederhole bis *Bedingung*
Sequenz

Bedingte Anweisung II: zweiseitig bedingte Anweisung

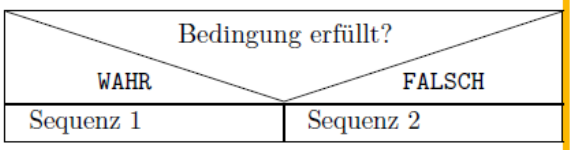
Bisher haben wir uns bei Bedingungen nur darauf beschränkt, Anweisungen auszuführen, wenn eine zu überprüfende Bedingung `wahr` ist. Falls die Bedingung nicht eingetreten ist, wurden alle Anweisungen innerhalb der Kontrollstruktur ignoriert und das restliche Programm abgearbeitet. Oftmals ist es aber notwendig, dass auch im Falle einer nicht eintretenden Bedingung bestimmte Anweisungen ausgeführt werden sollen.

Merke (Zweiseitig bedingte Anweisung)

Die Kontrollstruktur zweiseitig bedingte Anweisung dient dazu, um unterschiedliche Sequenzen abhängig von einer Bedingung auszuführen. Dabei werden sowohl im Fall `wahr` (engl. `true`) als auch im Fall `falsch` (engl. `false`) zusätzliche Anweisungen ausgeführt.

Zweiseitig bedingte Anweisung

Struktogramm:



Quelle: in Auszügen zusammengestellt aus Pfeffer, Wolfgang: Arbeitsheft Algorithmen, o.J.; online unter https://bwinf.de/jugendwettbewerb/ (19.2.2022).