IT 4 KIDS

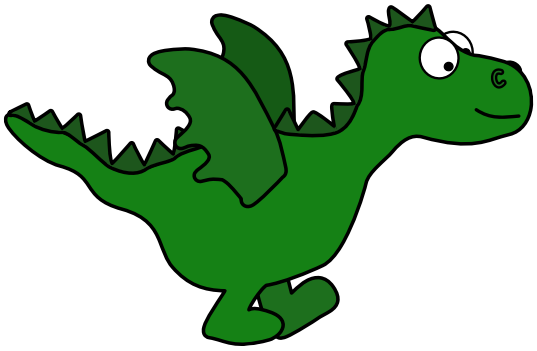
5. – 7. Klasse

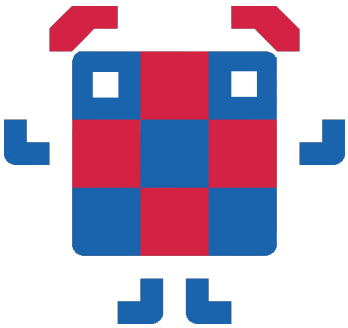
Teil 2

Informatik erleben

– Funktionen –

Hier gibt es Unterrichtsverlaufspläne, Arbeitsblätter, Kopiervorlagen und Programmieraufgaben für den Einstieg in die Welt der Algorithmen und der Informatik.





Herzlich Willkommen zu unserer Lernreihe

Wie schön, dass Du da bist! 🤗 Mit dieser Unterrichtsreihe bekommst Du alles, was Du brauchst, um den Einstieg in die Programmierung mit Deiner Lerngruppe sorglos zu gestalten an die Hand. Mithilfe der für den Unterricht konzipierten Lernsoftware **Cubi** kannst Du das Thema **Programmierung** kleinschrittig und ganzheitlich mit Deiner Klasse entdecken.

Das IT4Kids-Material zu **Schleifen**, **Verzweigungen**, **Variablen** und Co. vermittelt die grundlegenden Programmierkenntnisse, um das Informatik-Thema **Algorithmen** vollständig zu behandeln.

Keine Sorge: Es wird kein Vorwissen benötigt. Durch unsere Materialien kannst Du Dir die Welt der Programmierung Schritt für Schritt erschließen. Mithilfe vorgefertigter Programmieraufgaben für die Schüler*innen und ausgearbeiteter Unterrichtsverlaufspläne für Dich als Lehrkraft, wollen wir Dir so viel Unterrichtsvorbereitung abnehmen wie möglich. Dazu stellen wir Dir auch Arbeitsblätter, Kopiervorlagen und Musterlösungen zur Verfügung.

Du möchtest Dich erst einmal mit unserer Lernsoftware vertraut machen? Kein Problem! Du findest den Cubi-Editor unter `editor.i4k.org`. Das **Benutzerhandbuch für die Lernsoftware Cubi** verrät Dir alles, was Du bei der Nutzung der Lernsoftware wissen solltest. Du findest es im Begleitmaterial.

Informatik als Fachunterricht in der Sekundarstufe I

In immer mehr Bundesländern erhält das Fach Informatik einen festen Platz im Stundenplan der Jahrgangsstufen 5 bis 7 oder wird dort erprobt. Das Ziel der vorliegenden Lernreihe von IT4Kids ist, Schüler*innen einen ganzheitlichen Einstieg in den **Inhaltsbereich Algorithmen** zu bieten. Dabei vermitteln wir insbesondere die Prozessbereiche **Modellieren und Implementieren**, **Begründen und Bewerten** und **Kommunizieren und Kooperieren**. Wir folgen hiermit den *Grundsätze[n] und Standards für die Informatik in der Schule* der Gesellschaft für Informatik e.V.

Schon gewusst?

Die Inhalte von IT4Kids entsprechen dem Strategiepapier der KMK für *Bildung in der digitalen Welt* und den Zielen für nachhaltige Entwicklung.

Im Laufe der vorliegenden Unterrichtsstunden lernen die Schüler*innen verschiedene Anweisungen in der grafischen Programmierumgebung **Cubi** kennen. Mit diesen können sie **sequentielle Algorithmen** und Algorithmen mit **Schleifen** und **bedingten Anweisungen** modellieren und implementieren. Im zweiten Teil der Lernreihe kommen **Variablen** und **Funktionen** hinzu. Außerdem wird das große Thema **Fehlersuche und Testen** aufgearbeitet und Programme werden mit Stift und Papier geplant. Den Abschluss bildet ein kreatives Projekt, in dem eigene Spiele entwickelt werden.

Die Lernentwicklung der Schüler*innen wird über die gesamte Lernreihe hinweg auch durch **überfachliche Kompetenzen** gefördert. Dadurch, dass sie die Konsumperspektive verlassen und erfahren, wie sie die digitale Welt kreativ mitgestalten können, werden **personale Kompetenzen** gestärkt, die auf die Förderung der Selbstwirksamkeit, -behauptung und -reflexion abzielen.

Auch die **motivationale Einstellung** der Schüler*innen wird mit den Lehrinhalten gesteigert. Die Neugierde der Schüler*innen für den neuen Themenbereich der Informatik wird geweckt, sodass sie sich für diesen begeistern und neuen Problemstellungen ausdauernd begegnen können. Dabei wird eine positive Einstellung gegenüber experimentellem Lernen und die Frusttrationstoleranz der Schüler*innen ausgebaut.

Durch eine Varianz an Sozialformen und die Integration von Partner- und Gruppenarbeiten werden **soziale Kompetenzen** wie das Agieren in kooperativen Lernprozessen oder der konstruktive Umgang mit Konflikten und Vielfalt gefordert und gefördert.

Die Schüler*innen erweitern ihre **Methodenkompetenz**, indem sie beim Lernen strukturiert sowie systematisch vorgehen und eigene Arbeitsprozesse planen und organisieren. Das Lösen von Programmieraufgaben fordert ein hohes Maß an Problemlösefähigkeit, das im Verlauf der Lernreihe auf- und ausgebaut wird. Bei der Arbeit an ebendiesen Programmieraufgaben sowie den damit verbundenen Recherche- und Präsentationsaufträgen ist die Förderung der Medienkompetenz der Schüler*innen allgegenwertig.

Verankerungen von Inhalten zu Algorithmen in Bildungsplänen

Die Inhalte wurden für die verschiedenen Anforderungen der länderspezifischen Bildungspläne entwickelt. Um deren Varianz gerecht zu werden, wurden auch Unterrichtsstunden konzipiert, dessen Kernkompetenzen nur in einzelnen Bundesländern gefordert sind. In der folgenden Tabelle findest Du eine Übersicht über die Unterrichtsstunden. Aus ihr kannst Du entnehmen, welche Unterrichtsstunden im Bildungsplan Deines Bundeslandes verankert sind.

Zuordnung der Unterrichtseinheiten zu den landesspezifischen Bildungsplänen (Stand: Juli 2024)

Bundesland	Unterrichtseinheit								
	Eingabe-Verarbeitung-Ausgabe	Standardalgorithmen	Variablen	Fehlersuche & Testen	Schleifen mit Bedingungen	Verschachtelte Verzweigungen	Struktogramm	Funktionen	Eigenes Spiel
Baden-Württemberg			X	X			X		X
Bayern	X		X	X	X	X	X	X	X
Berlin/Brandenburg			X					X	X
Hamburg	X		X		X	X		X	X
Hessen			X	X			X		X
Mecklenburg-Vorpommern	X		X	X	X	X	X		X
Niedersachsen			X	X	X	X	X	X	X
Nordrhein-Westfalen							X	X	
Rheinland-Pfalz			X	X	X	X			
Saarland			X	X	X	X	X		X
Sachsen					X	X	X		
Schleswig-Holstein	X	X	X	X	X	X		X	X

Anmerkung: In Bremen gibt es keinen Informatikunterricht. In Sachsen-Anhalt gibt es Informatik im Wahlpflichtbereich nur in höheren Jahrgangsstufen. In Thüringen sind nur Themen des ersten Teils dieser Lernreihe im Bildungsplan verankert.

Inhaltsverzeichnis

Material für 1 x 45 Minuten

Funktionen

Die Königsdisziplin im Programmieren ist nicht nur, funktionierenden Code zu schreiben, sondern auch einen, der verständlich, wartbar und vor allem effizient ist. In dieser Stunde geht es um Funktionen. Durch sie wird es möglich, eine Abfolge von Bausteinen an verschiedenen Stellen im Programm wiederzuverwenden. Das erhöht die **Wartbarkeit** des Codes, weil Änderungen leichter eingefügt werden können. Außerdem kann so geschriebener Code wiederverwendet werden und der Schreibprozess wird dadurch kürzer.

Anknüpfung an Bildungspläne

Bayern (Informatik, Jgs. 7), **Berlin/Brandenburg** (Wahlpflichtfach Informatik, Jgs. 7), **Hamburg** (Informatik, Jgs. 7), **Niedersachsen** (Informatik, Jgs. 5 – 7), **Nordrhein-Westfalen** (Informatik, Jgs. 5/6), **Schleswig-Holstein** (Informatik, Jgs. 5 – 7)

Überfachliche Kompetenzen

Die Schülerinnen und Schüler...

- ... sind motiviert, Neues zu lernen und Dinge zu verstehen, strengen sich an, um sich zu verbessern.
- ... kennen und nutzen unterschiedliche Wege, um Probleme zu lösen.

Fachliche Kompetenzen

Die Schülerinnen und Schüler...

- ... strukturieren Handlungsabläufe in logische Teileinheiten.
- ... analysieren eine algorithmische Problemstellung, um Teilprobleme zu identifizieren.
- ... implementieren Algorithmen unter Berücksichtigung des Prinzips der Modularisierung.

Neue Bausteine

Funktionen:	Funktionen mit und ohne Parameter
Aussehen:	Setze Größe auf ... %, Nächstes Kostüm
Bewegung:	Springe zu x:... y:...

Weitere verwendete Bausteine

Schleifen: **Wiederhole ... mal**
Kontrolle: **Warte ... Sekunden**

Vorbereitung

Mache Dich mit den Definitionen von Funktionen vertraut und suche die Muster in dem Cubi-Level **Tanz**. Du findest die Level, indem Du oben links im **Menü** ☰ auf **Öffnen** 🗀 gehst und sie anschließend unter dem Tab **Entdeckerreihe** auswählst oder die QR-Codes weiter unten scannst.
Vollziehe den Unterschied zwischen Schleifen und Funktionen nach. **Weitergedacht:** Informiere Dich über die Bedeutung von Modularisierung in professionellen Softwareprojekten, z.B. mit dem Suchwort **Modulare Softwareentwicklung**.

Unterrichtsverlaufsplan

Zeit	Phase	Unterrichtsschritte	SF	Material
5	Einstieg	Definition von Modulen und Funktionen	P	
15	Erarbeitung	Mustersuche in Programmen, Funktionen in Cubi-Level **Tanz** programmieren	P, EA/ PA	☐ Präsentationstechnik ☐ Tablets/Laptops/PCs
15	Arbeitsphase	Cubi-Level **Vieleck**	EA/ PA	☐ Tablets/Laptops/PCs ☐ KV **Hilfekarten Vieleck** ☐ ggf. KV **QR-Codes Funktionen**
10	Präsentation und Reflexion	Vorstellen der Programmierergebnisse, Ausflug in professionelle Softwareprojekte	P	☐ Präsentationstechnik

EA = Einzelarbeit, GA = Gruppenarbeit, PA = Partnerarbeit, P = Plenum, S = Sitzkreis, SF = Sozialform

Einstieg

Eröffne die Stunde, indem Du mit den Schüler*innen das folgende Gedankenexperiment durchspielst: Stellt euch vor, ihr wollt zusammen eine Gemüsepfanne kochen. Jeder von euch bekommt eine bestimmte Aufgabe: Zwiebeln hacken, verschiedene Gemüsesorten klein schneiden, Reis kochen, Gemüse anbraten, würzen. Worauf muss geachtet werden?

- Das Gemüse sollte ungefähr gleich groß geschnitten werden, damit es gleichzeitig gar ist.
- Wenn jemand die Aufgabe **Zwiebeln hacken** übernimmt, sollte die Zwiebel am Ende der Aufgabe auch gehackt sein und nicht nur der Stängel abgeschnitten sein.
- Die Reihenfolge der Arbeitsschritte sollte eingehalten werden.
- Wenn jemand das Gemüse würzen soll, muss er/sie wissen, welche Gewürze schon verwendet wurden.

Stelle den Bezug zur Informatik her und erkläre die folgenden Sachverhalte. An großen Softwareprojekten, wie z.B. Google, WhatsApp oder TikTok, arbeiten hunderte Entwickler*innen an einem Projekt. Der Quellcode kann mehrere Millionen Zeilen haben. Um nicht den Überblick zu verlieren, müssen sich die Teams gut absprechen, wer welche Funktionalitäten implementiert und genau definieren, wie andere mit ihrem Stück Programm weiterarbeiten können – genau wie in eurem Kochprojekt.
Ein wichtiges Werkzeug hierfür sind **Funktionen**: eine Reihe von Anweisungen, die man einmal schreiben muss und dann an mehreren Stellen im Programm verwenden kann.

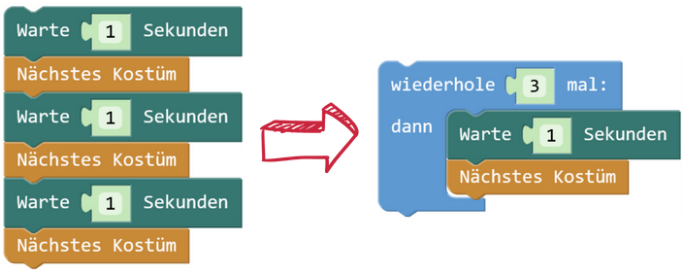
Erarbeitung

Leite dazu über, dass ihr euch nun **Funktionen** in Cubi anschaut. Öffne das Cubi-Level **Tanz** über eine digitale Tafel oder ähnliche Präsentationstechnik. Wie bewegt sich die Tänzerin? Welche Muster erkennen die Schüler*innen? Wie könnte man das Programm verkürzen? An welchen Stellen hilft eine **Schleife** und an welchen nicht? Warum? Verkürzt das Programm mit drei **Wiederhole 3 mal**-Schleifen.

Levelvorlage:
`level.i4k.org/tanz_sek`

Levellösung:
`level.i4k.org/tanz_sek_lsg`

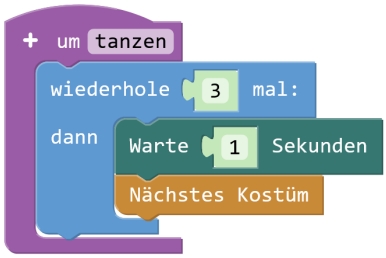
Lösung: *Die Bausteine **Warte … Sekunden** und **Nächstes Kostüm**, die direkt untereinanderstehen, können mit einer **Wiederhole … mal**-Schleife zusammengefasst werden.*



Weiter kann das Programm **Tanz** mit **Schleifen** nicht verkürzt werden, obwohl die Bausteine **Wiederhole 3 mal**, **Warte 1 Sekunde** und **Nächstes Kostüm** dreimal exakt gleich

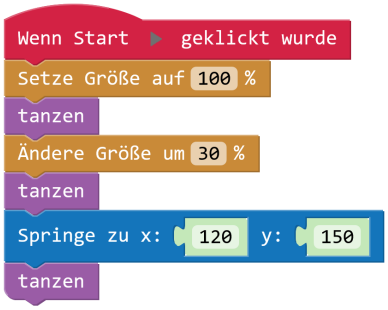
in dem Programm vorkommen. Das liegt daran, dass die Bausteine zwischen den Schleifen unterschiedlich sind und nicht in die Wiederhole ... mal-Schleife hineingezogen werden können.

Was eine Schleife hier nicht lösen kann, kann eine Funktion. Ziehe den Funktionsbaustein Um etwas tun in die Programmierfläche. Er steht für sich allein und wird nicht mit den bisherigen Bausteinen verbunden.

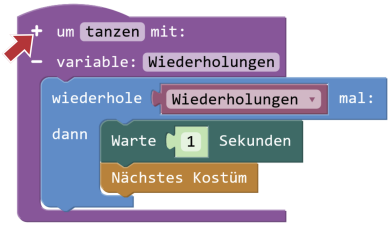


Ändere den Namen der Funktion von **etwas tun** zu **tanzen**, indem Du auf das helle Feld klickst. Löse eine der Schleifen aus dem großen Programm und ziehe sie in den Funktionsbaustein **tanzen**.

Öffne erneut die Kategorie Funktionen. Hier befindet sich nun ein neuer Befehlsbaustein mit dem Namen eurer Funktion tanzen. Ziehe ihn an die Stelle im Programm, wo vorher die Schleife war. Drücke auf **Start** ▶ und beobachtet, wie die Tänzerin genau das Gleiche macht wie vorher. Tausche nun auch die anderen Schleifen gegen den neuen Funktionsbaustein aus. Auch jetzt tanzt die Tänzerin so wie vorher, jedoch ist das Programm deutlich kürzer und einfacher zu lesen.



Baut gemeinsam die Geräte auf. Gib den Schüler*innen etwas Zeit, das Programm und das Erstellen der Funktionen für sich selbst zu wiederholen.



Richte anschließend die Aufmerksamkeit der Schüler*innen wieder nach vorne und stelle eine Besonderheit von Funktionen vor: **Parameter** oder auch **Übergabewerte** genannt. Sie werden wie eine Variable in der Funktion benutzt und bekommen ihren Wert beim Aufruf der Funktion. Füge einen Parameter zu der Tanz-Funktion hinzu, indem Du auf das **Plus** im Funktionskopf drückst. Ändere den Namen des Parameters von **x** zu **Wiederholungen**. Öffne die Kategorie Variablen und ziehe den Variablenbaustein Wiederholungen in das Zahlenfeld der Wiederhole ... mal-Schleife. Füge als letztes einen Zahlen-Baustein aus der Kategorie Mathe in den Funktionsaufruf und puzzle ihn an den Parameter **Wiederholungen**. Trage verschiedene Werte für die Wiederholungen ein und starte das Programm, z.B. **3** oder **5**.

Gib den Schüler*innen etwas Zeit, das Hinzufügen von Parametern für sich selbst zu üben. Schnelle Schüler*innen können die **Wartezeit** als zweiten Parameter hinzufügen und in den Warte-Baustein integrieren.

Was beobachten die Schüler*innen? Welche Vorteile sehen sie in der Verwendung von Parametern? Die Tänzerin kann bei jedem Funktionsaufruf unterschiedlich viele Schritte tanzen, ohne, dass der Wert in der Schleife verändert werden muss.

Deklaration und Aufruf

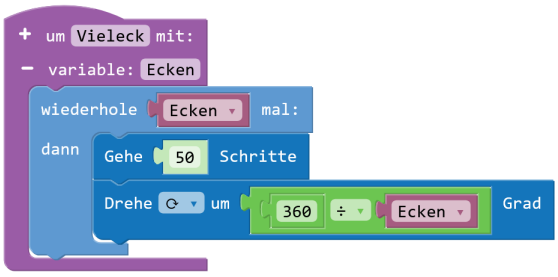
Eine Funktion zu definieren oder auch zu deklarieren macht man genau einmal in einem Programm pro Funktion. Danach kann man sie beliebig oft aufrufen und ausführen.

Arbeitsphase

Formuliere den Arbeitsauftrag für die Arbeitsphase: Die Schüler*innen öffnen das Level **Salto** und schreiben eine Funktion, welche als Parameter eine Zahl nimmt und dann ein Vieleck zeichnet, das so viele Ecken hat, wie durch den Parameter vorgegeben werden.

Levelvorlage:
`level.i4k.org/salto_sek`

Levellösung:
`level.i4k.org/salto_sek_lsg`



Die Schüler*innen erstellen eine neue Funktion und geben ihr den Namen Vieleck. Sie fügen über das **Plus** einen neuen Parameter hinzu und benennen ihn als **Ecken** oder **Kanten**. Die Funktion Vieleck besteht aus einer Wiederhole ... mal-Schleife und einem Gehe- und einem Drehe-Baustein. Durch das Hinzufügen des Parameters wurde eine neue Variable mit dem Namen **Ecken** erstellt. Diese Variable Ecken gibt die Anzahl der Wiederholungen für die Schleife an. Außerdem ergibt sich durch die Division des Vollwinkels 360° durch den Parameter Ecken die Gradzahl für die Drehung.

Als Hilfestellung kannst Du einzelnen Schüler*innen auf eigenen Wunsch Tippkarten austeilen. Als Knobelaufgabe können die Schüler*innen einen zweiten Parameter **Länge** hinzufügen und die zugehörige Variable Länge in den Gehe-Baustein ziehen. Wer möchte, kann auch die Länge abhängig von der Anzahl der Ecken berechnen.

Präsentation und Reflexion

Zum Abschluss der Stunde können die Schüler*innen ihre Arbeitsergebnisse vorstellen und darauf eingehen, was gut geklappt hat oder was ihnen schwer gefallen ist. Wenn eine digitale Tafel oder andere Präsentationstechnik vorhanden ist, können einzelne Lösungswege dort in groß gezeigt werden.
Leite ein Unterrichtsgespräch an, in dem die Schüler*innen Vermutungen anstellen können, wofür Funktionen in großen Softwareprojekten genutzt werden können. Eine Auswahl an Einsatzmöglichkeiten ist:

- Mehrere Programmierer*innen können an einem Programm gleichzeitig schreiben, weil jeder seine eigenen Funktionen schreibt und diese hinterher zusammengefügt werden.
- Es geht schneller, wenn man Programmcode nicht doppelt schreiben muss, sondern schon geschriebenen Programmcode wiederverwenden kann.

- Wenn später etwas an der Funktion geändert werden muss, muss man es nur an einer Stelle ändern und nicht an allen.

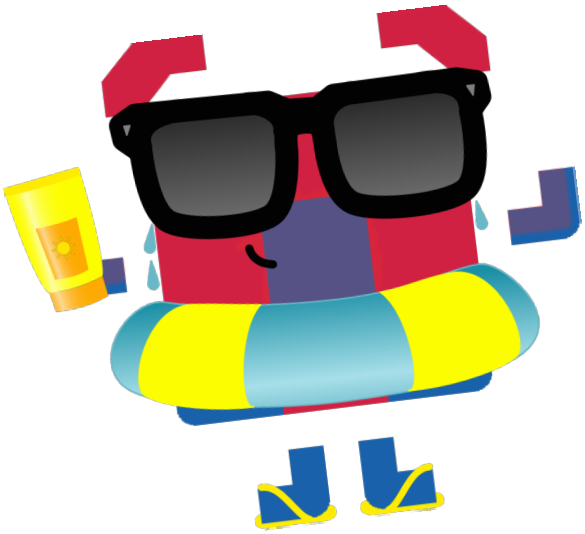
  Weitergedacht: Was passiert, wenn man etwas an den Parametern einer Funktion im Nachhinein ändert? Dann ist Vorsicht geboten, denn das ist eine häufige Fehlerquelle.

- Beim Programmieren kann man auch Funktionen nutzen, die man nicht selbst geschrieben hat, z.B. sind die Bausteine in Cubi alle Funktionen. Im Hintergrund wird Code in einer anderen Programmiersprach aufgerufen, welcher dann dazu führt, dass auf der Bühne etwas passiert.

Beende die Stunde mit einem Ausblick auf das nächste Thema.

Geschafft!

Großartig, Du hast es durch **Teil 2 der Lernreihe** geschafft! Was eine tolle Leistung!



Jetzt kannst Du Dich zurücklehnen, während Deine Klasse fleißig programmiert.

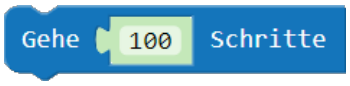
Baustein-Lexikon

Start

Der erste Baustein eines Blocks aus mehreren Bausteinen ist immer ein **Starbaustein**. Ein Programm einer Figur kann beliebig viele **Startbausteine** haben. **Startbausteine** zeichnen sich durch die Rundung am oberen Teil aus. Diese sagt aus, dass **Startbausteine** nicht an andere Bausteine angehängt werden können.

Der **Startbaustein Wenn Start geklickt wurde** ist der erste Baustein, den die Schüler*innen kennenlernen. Nachfolgende Bausteine werden nacheinander ausgeführt, unmittelbar nachdem das Level gestartet ▸ wurde.

Bewegung



Der Baustein **Gehe ... Schritte** bewegt die Figur die entsprechende Anzahl an Pixel in die aktuelle Richtung der Figur. Im Normalfall ist dies bei Programmstart nach rechts.

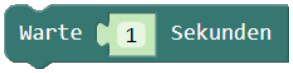
Mithilfe des **Springe zu x: ... y: ...** kann eine Koordinate festgelegt werden, zu der die Figur springen soll. Die Koordinaten können unterhalb der Bühne abgelesen werden.

Mit dem Baustein **Drehe rechts/links um ... Grad** dreht sich die Figur in die ausgewählte Richtung um die entsprechende Gradzahl. In den ersten Leveln brauchen die Schüler*innen nur den rechten Winkel.

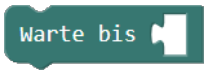
Kontrolle

Der Baustein **Wenn dann** leitet eine Verzweigung ein. Oben an das **wenn** wird eine Bedingung angepuzzelt. Diese ist entweder **wahr** oder **falsch**. Wenn die Bedingung wahr ist, werden die Bausteine, die neben dem **dann** stehen ausgeführt. Ist die Bedingung falsch, also nicht erfüllt, werden die Bausteine bei **dann** übersprungen und nicht ausgeführt.
Drückt man oben links auf dem Baustein auf das weiße Plus, wird ein neuer Verzweigungsarm hinzugefügt, an den eine weitere Bedingung angepuzzelt werden kann. Diese wird jedoch nur überprüft, wenn die erste Bedingung falsch war.

Der Baustein **Wenn dann sonst** leitet eine Verzweigung ein. Oben an das **wenn** wird eine Bedingung angepuzzelt. Diese ist entweder **wahr** oder **falsch**. Wenn die Bedingung wahr ist, werden die Bausteine, die neben dem **dann** stehen ausgeführt. Die Bausteine hinter **sonst** werden übersprungen.Ist die Bedingung falsch, also nicht erfüllt, ist es genau andersherum und die Bausteine bei **dann** werden übersprungen und an ihrer Stelle werden die Bausteine, die hinter **sonst** stehen, ausgeführt.
Drückt man oben links auf dem Baustein auf das weiße Plus, wird ein neuer Verzweigungsarm hinzugefügt, an den eine weitere Bedingung angepuzzelt werden kann. Diese wird jedoch nur überprüft, wenn alle vorherigen Bedingungen falsch waren.



Gelangt ein Programm zu einem **Warte**-Baustein, dann bleibt es hier für die Anzahl der eingegebenen Sekunden stehen. Andere Programmteile der Figur, die ihren eigenen **Startbaustein** haben, werden hierdurch nicht unterbrochen. Erst wenn die Zeit um ist, wird der nächste Baustein ausgeführt.



Gelangt ein Programm zu einem **Warte bis**-Baustein, dann bleibt es hier solange stehen, bis die Bedingungen erfüllt ist. Andere Programmteile der Figur, die ihren eigenen **Startbaustein** haben, werden hierdurch nicht unterbrochen. Erst wenn die Bedingung erfüllt ist, wird der nächste Baustein ausgeführt.

Fühlen

Bausteine der Kategorie **Fühlen** werden als Bedingungen in **Verzweigungen** oder **Schleifen mit Bedingungen** angepuzzelt. Das Programm prüft, ob die Bedingung **wahr** oder **falsch** ist. Ist die Bedingung wahr, werden die Bausteine in der **Verzweigung** oder **Schleife** ausgeführt.

Mit diesem Baustein wird geprüft, ob die Figur eine bestimmte Farbe berührt. Durch Klicken auf das Farbfeld kann die Farbe geändert werden, die geprüft wird.

Mit diesem Baustein kann die Figur reagieren, wenn sie etwas berührt. Durch Klicken auf den kleinen Pfeil kann ausgewählt werden, ob die Figur auf den Rand oder eine andere Figur reagieren soll, wenn es noch weitere Figuren in dem Level gibt.

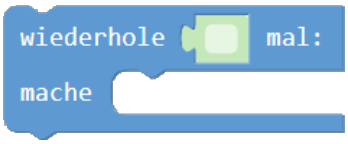
Mathe

Mit diesem Baustein kann ein Bereich definiert werden, aus dem eine zufällige Zahl gewürfelt wird. Die Zahlen können beliebig hoch sein, die zweite Zahl muss allerdings größer als die erste Zahl sein.

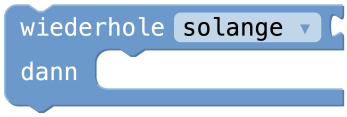
Variablen

Über die Kategorie **Variablen** können Variablen erstellt, umbenannt und gelöscht werden. Variablen können oben links im Vorschaufenster angezeigt oder wieder versteckt werden. **Setze *Variable* auf** ändert den Wert der Variable auf den angegebenen Wert. Mit **Erhöhe *Variable* um** und **Verringere *Variable* um** kann der Wert der Variable um ein festes Intervall positiv oder negativ geändert werden kann.

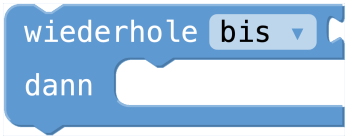
Schleifen



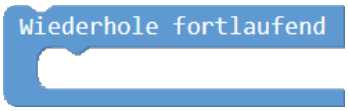
Mit der **Wiederhole ... mal**-Schleife können Bausteine, die in die Schleife eingefügt werden, wiederholt werden. Klicke auf die Zahl, um die Anzahl der Wiederholungen zu ändern.



Die **Wiederhole solange**-Schleife ist äquivalent zu **Wiederhole bis nicht**. Bei jedem Durchlauf wird geprüft, ob die Schleifenbedingung erfüllt ist. Solange die Bedingung erfüllt ist, werden die Bausteine in der Schleife ein weiteres Mal wiederholt. Ist die Bedingung nicht mehr erfüllt, wird die Schleife verlassen und die Bausteine nach der Schleife ausgeführt.



An die **Wiederhole bis**-Schleife kann ein Baustein aus den Kategorien **Fühlen** oder **Kontrolle** als Schleifenbedingung angepuzzelt werden. Bei jedem Durchlauf wird geprüft, ob diese Schleifenbedingung erfüllt ist. Bis die Bedingung erfüllt ist, werden die Bausteine in der Schleife ein weiteres Mal wiederholt. Sobald die Bedingung erfüllt ist, wird die Schleife verlassen und die Bausteine nach der Schleife ausgeführt.



Alle Bausteine, die in die **Wiederhole fortlaufend**-Schleife eingefügt werden, werden endlos lange wiederholt. Das Level stoppt erst, wenn man es über die Pause-Taste unterbricht oder über den Zurücksetzen-Button stoppt. Unter dieser Schleife können keine Bausteine angehängt werden, weil diese nie ausgeführt würden.

Aussehen

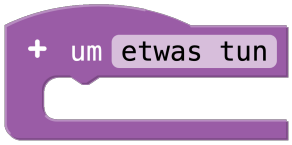
Mithilfe dieses Bausteins wechselt die entsprechende Figur das Kostüm und ändert so ihr Aussehen. Die Kostüme sind unterhalb der Bühne zu sehen. Hier wird zu dem Kostüm gewechselt, welches in dem Feld ausgewählt wurde.

Mithilfe dieses Bausteins wechselt die entsprechende Figur das Kostüm und ändert so ihr Aussehen. Die Kostüme sind unterhalb der Bühne zu sehen. Hier wird zu dem Kostüm gewechselt, welches unter der Bühne nach dem aktuellen Kostüm abgebildet ist.

Mithilfe dieses Bausteins ändert die entsprechende Figur ihre Größe. Der Wert kann geändert werden. Je größer der Wert, desto größer wird die Figur.

Wenn der **Sage**-Baustein ausgeführt wird, erscheint für die Figur eine Sprechblase. Dort wird der Text angezeigt, welcher in das Textfeld geschrieben wurde. Nach der angegebenen Zeit verschwindet die Sprechblase und der nächste Baustein wird ausgeführt.

Funktionen



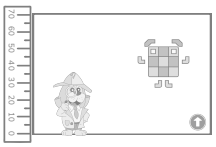
Mit Hilfe der Bausteine der Kategorie **Funktionen** kann man seine eigenen Bausteine erstellen. Dafür schreibt man ein Mini-Programm in den Baustein **Um etwas tun** und ändert den Namen in eine eigene Überschrift. Immer wenn der eigene Baustein ausgeführt wird, also in einem Programm unter einem **Startbaustein** steht, werden genau die Befehle ausgeführt, die man bei **Um etwas tun** eingegeben hat. Das ist ins Besondere dann hilfreich, wenn eine bestimmte Abfolge von Bausteinen mehrfach an unterschiedlichen Stellen eines Programms vorkommt.

Name: _____ Datum: _____

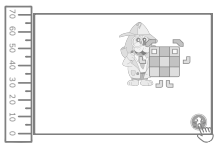
Detektivsuche

Aufgabe 1: Betrachte die beiden Bilder. Du bist der Detektivhund. Welche Aussagen sind wahr (✓) und welche sind falsch (✗)? Markiere sie.

1. Berühre ich Cubi?
2. Berühre ich den Rand?
3. Ist die Entfernung zum Rand oben kleiner als 20?
4. Ist die Entfernung zum Rand oben größer als 30?
5. Ist die Taste *Pfeil nach oben* gedrückt?



1. Berühre ich Cubi?
2. Berühre ich den Rand?
3. Ist die Entfernung zum Rand oben kleiner als 20?
4. Ist die Entfernung zum Rand oben größer als 70?
5. Ist die Taste *Pfeil nach oben* gedrückt?



Aufgabe 2: Aussagen und Bedingungen können mit **und** und **oder** verknüpft werden. Betrachte das **obere** Bild. Markiere auch die folgenden Aussagen als wahr (✓) oder falsch (✗).

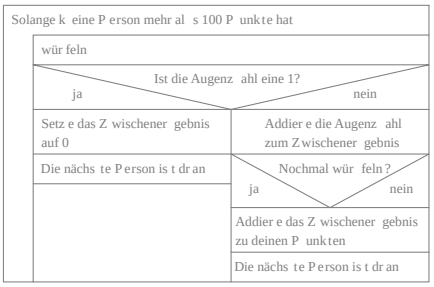
1. Berühre ich den Rand und berühre ich Cubi?
2. Ist die Entfernung zum Rand oben kleiner als 100 und größer als 10?
3. B
4. B

Auf

Nam

Die Gemeine 1

Aufgabe 1: Schaut euch das Struktogramm für einen Zug des Spiels **Die Gemeine 1** an. Welche Spielregeln könnt ihr daraus ableiten?

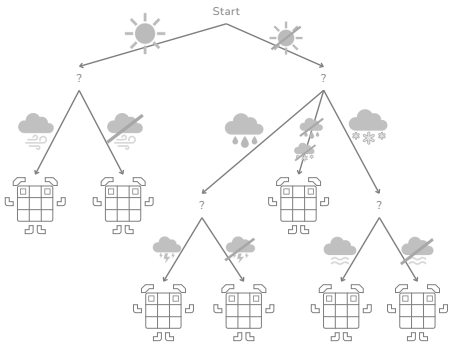


Aufgabe 2: Es spielen zwei Personen gegeneinander. Erweitere das Struktogramm um die Überprüfung, wer von den beiden gewonnen hat.

Name: _____ Datum: _____

Wettervorhersage

Aufgabe 1: Ziehe Cubis passende Kleidung an. Schau in der Kostümbibliothek nach, welche Varianten es von Cubi gibt.



Aufgabe 2: Welche Wetterkombinationen haben keinen Pfad im Entscheidungsbaum? Wie verhält sich Cubi in deinem Level?

Namen: _____ Datum: _____

Unsere Dokumentation (1)

Level: _____

Aufgabe 1: Beschreibt die Geschichte eures Levels.

Jahreszahl: _____

Ort: _____

Handlung: _____

Szene aus dem Level

Aufgabe 2: Charakterisiert die Figuren eures Levels

Name: _____

Alter: _____ Steuerbar: ☐ ja ☐ nein

Fähigkeiten: _____

Aufgabe: _____

Name: _____

Alter: _____ Steuerbar: ☐ ja ☐ nein

Fähigkeiten: _____

Aufgabe: _____

Weitere Angebote von IT4Kids

Du willst weiter mit IT4Kids arbeiten? Neben einer Modulreihe zu den Themen **Sequenzen**, **Schleifen** und **Verzweigungen** mit der Zielgruppe **Orimarstufe** bieten wir auch Fortbildungen für Dich und Dein Kollegium an. Hier lernen wir gemeinsam den Cubi-Editor kennen, sammeln grundlegende Programmiererfahrungen und planen eine erste Unterrichtsstunde mit Cubi speziell für Deine Klasse. Nach der Fortbildung kannst Du direkt am nächsten Tag eine Stunde Programmierung mit Deiner Klasse ausprobieren, weil wir alles gemeinsam in der Fortbildung vorbereitet haben. Alle aktuellen Informationen zu unserem Fortbildungsangbot findest du auf unserer Webseite unter `www.i4k.org/fortbildung`. 🤓
Du bist noch unschlüssig? Dann schau Dich gerne auf unserer Webseite `www.i4k.org` um, stöbere durch unser Material und lerne uns ein bisschen besser kennen. Wenn auf dem Weg Fragen aufkommen oder Du mit uns ins Gespräch über die Materialien kommen möchtest, dann melde Dich gerne per E-Mail unter info@it-for-kids.org bei uns oder nimm über unsere Webseite `www.i4k.org/kontakt` mit uns Kontakt auf. 💬
Wir freuen uns auf Dich! 🤗

Hilf uns, besser zu werden!

Dir sind Fehler in dem Material aufgefallen?
Du hast Verbesserungsvorschläge?
Du möchtest mehr zum Einstieg in die Programmierung?

Wir freuen uns über Dein Feedback:

feedback.i4k.org/lk/sek1/teil2