



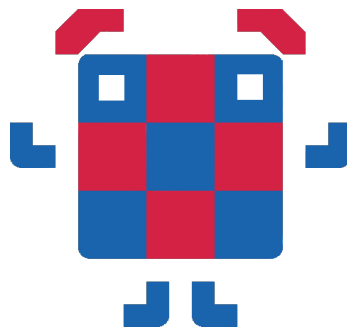
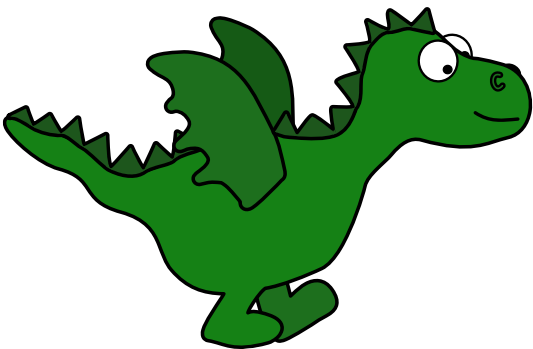
5. – 7. Klasse

Teil 2

Informatik erleben

– Fehlersuche –

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



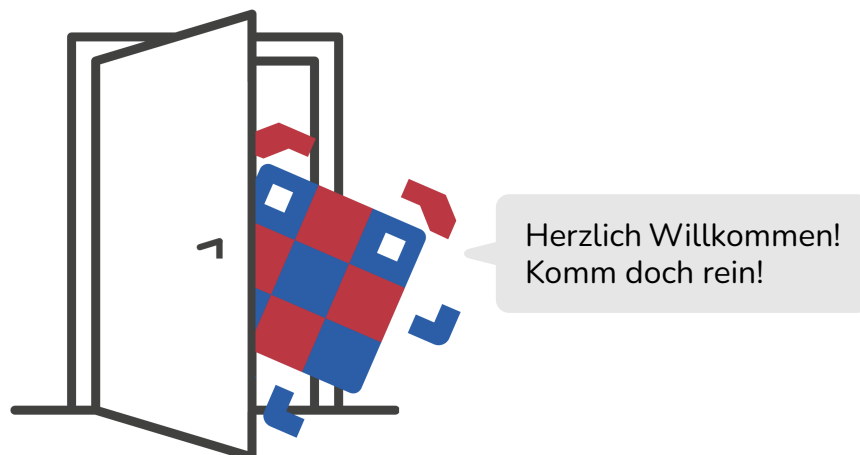
Herzlich Willkommen zu unserer Lernreihe

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.



Das vorliegende Lehrmaterial von IT4Kids und zugehörige Begleitmaterialien für Schüler*innen stehen, soweit nicht anders angegeben, unter der Creative Commons-Lizenz CC BY-NC-SA 4.0. Weitere Informationen zu der Lizenz findest Du hier: <http://creativecommons.org/licenses/by-nc-sa/4.0/>



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** aufge-

arbeitet 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 Frustrationstoleranz 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 allgegenwärtig.

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

EVA



45 Minuten
plugged

Standardalgorithmen



45 Minuten
(un)plugged

Variablen



90 Minuten
plugged

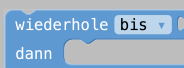
Fehlersuche & Testen



45 Minuten
plugged

Seite 6

Schleifen



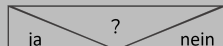
45 Minuten
plugged

Verzweigungen



90 Minuten
plugged

Struktogramm



45 Minuten
plugged

Funktionen



45 Minuten
plugged

Eigenes Spiel



90 Minuten
plugged



Fehlersuche und Testen

Fehlersuchen im Quellcode – oder auf Englisch **Debugging** – ist eine der Hauptherausforderungen von Programmierer*innen: Warum kompiliert mein Programm nicht? Was bedeutet diese Fehlermeldung? Denn auch Expert*innen schreiben keinen fehlerfreien Code auf Anhieb. Dafür gibt es zu viele Abhängigkeiten zwischen verschiedenen Programmteilen, die es zu beachten gilt. Daher ist es umso wichtiger – auch als Programmierneuling – eine positive Haltung zu Fehlern aufzubauen und sie als Ausgangspunkt für neues Wissen zu sehen, um sich nicht von ihnen demotivieren zu lassen. Mindestens genauso wichtig ist es, sein Programm zu testen, wenn es vermeintlich fehlerfrei läuft: Wie verhält es sich in Grenzfällen? Wurden wirklich alle möglichen Eingaben berücksichtigt?

Anknüpfung an Bildungspläne

Baden-Württemberg (Aufbaukurs Informatik, Jgs. 7), **Bayern** (Informatik, Jgs. 7; Informationstechnologie, Anfangsunterricht), **Hessen** (Wahlfach Informatik, Jgs. 7), **Mecklenburg-Vorpommern** (Informatik und Medienbildung, Jgs. 6), **Niedersachsen** (Informatik, Jgs. 5 – 7), **Saarland** (IKT, S3), **Schleswig-Holstein** (Informatik, Jgs. 5 – 7), **optional: Rheinland-Pfalz** (IPS, Jgs. 5/6)

Überfachliche Kompetenzen

Die Schülerinnen und Schüler...

- ... gehen beim Lernen strukturiert und systematisch vor, planen und organisieren eigene Arbeitsprozesse.
- ... haben Vertrauen in die eigenen Fähigkeiten und glauben an die Wirksamkeit des eigenen Handelns.
- ... arbeiten ausdauernd und konzentriert, geben auch bei Schwierigkeiten nicht auf.
- ... können Informationen sammeln, aufbereiten, bewerten und präsentieren.

Fachliche Kompetenzen

Die Schülerinnen und Schüler...

- ... testen Algorithmen unter wechselnden Ausgangsbedingungen.

- ... testen die Korrektheit und die Wirkungsweise eines Algorithmus systematisch.
- ... beurteilen Programme im Hinblick auf Korrektheit.
- ... erkunden die Ursachen von Programmfehlern systematisch.

Neue Bausteine

Es werden keine neuen Bausteine eingeführt.

Vorwissen

Algorithmisches Verständnis, **Variablen**

Vorbereitung

Mach Dich mit den Beispielprogrammen aus der Arbeitsphase vertraut und löse die Quizfragen. Du findest das 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.

Unterrichtsverlaufsplan

Zeit	Phase	Unterrichtsschritte	SF	Material
5	Einstieg	Unterrichtsgespräch zu bisherigem Umgang mit Programmierfehlern und Strategien zur Fehlersuche in Cubi-Leveln	P	
15	Erarbeitung	Funktionsweise eines Algorithmus in Pseudocode-Form analysieren durch das Bestimmen von Variablenwerten	EA	<input type="checkbox"/> AB Variablen raten
15	Arbeitsphase	Spielregeln des Cubi-Leveln Wüstenjagd herausfinden	EA/ PA	<input type="checkbox"/> AB Wüstenjagd <input type="checkbox"/> Tablets/Laptops/PCs <input type="checkbox"/> ggf. KV QR-Codes Fehlersuche und Testen 2
10	Sicherung und Reflexion	Quiztime, Fehler als Teil des Programmierprozesses anerkennen	P	

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

Einstieg

Nenne zu Beginn der Stunde das Thema: **Fehlersuche**. Leite ein Unterrichtsgespräch an. Mögliche Gesprächseinstiege sind:

- Wie seid ihr bisher mit Fehlern in euren Programmen umgegangen?
- An welche Fehlersuche-Level erinnert ihr euch?
- Was passiert, wenn ein Programm einen Fehler hat?

Erarbeitung

Wiederhole zunächst mit den Schüler*innen die Besonderheiten eines Pseudocodes:

- Algorithmus/Programm auf dem Papier
- Unabhängig von der Programmiersprache
- Für Menschen verständlich, für die Maschine/den Computer jedoch nicht
- Pseudocode ist wie eine „Stichwortliste“ für ein Programm

Teile dann das Arbeitsblatt **Variablen raten** aus. Alternativ kannst Du es auch über eine digitale Tafel oder ähnliche Präsentationstechnik zeigen und ihr löst die folgende Aufgabe im Plenum.

Kläre Verständnisfragen zu der Notation des Pseudocodes auf dem Arbeitsblatt **Variablen raten**. Formuliere im Anschluss den Arbeitsauftrag: Die Schüler*innen sollen herausfinden, welche Werte die Variablen bei den gegebenen Anfangswerten am Ende des Programms haben.



Besonders schnelle Schüler*innen können durch systematisches Ausprobieren oder scharfes Hinschauen herausfinden, welche Aufgabe dieser Algorithmus löst.

Vergleiche die Lösungen entweder in Kleingruppen oder im Plenum.

Arbeitsphase

Formuliere den Arbeitsauftrag für die kommende Arbeitsphase: Die Schüler*innen sollen durch systematisches Ausprobieren und eigene Analysemethoden herausfinden, welche Aufgabe das Cubi-Level **Wüstenjagd** erfüllt. Sie sollen auf dem Arbeitsblatt **Wüstenjagd** die Spielregeln und erkannte Abhängigkeiten eintragen. Haben sie das Grundprinzip erkannt, ermutige sie, Ausnahmen oder Grenzfälle zu identifizieren.

In dem Level hat sich außerdem ein Fehler eingeschlichen. Die Figur Cubi verhält sich nicht wie erwartet und läuft nach rechts, obwohl die Pfeiltaste nach links gedrückt wurde und andersherum. Was müssen die Schüler*innen in dem Programm ändern?

Teile das Arbeitsblatt **Wüstenjagd** aus. Baue außerdem mit den Schüler*innen gemeinsam die Tablets, Laptops oder Computer auf und stell sicher, dass jedes Gerät mit dem Internet verbunden ist. Lass sie den Cubi-Editor über editor.i4k.org öffnen und stell sicher, dass alle zu dem Level **Wüstenjagd** finden.



Levelvorlage:

level.i4k.org/wuestenjagd_sek



Levellösung:

level.i4k.org/wuestenjagd_sek_lsg

Sicherung und Reflexion

Öffne das Level **Wüstenjagd** über eine digitale Tafel oder ähnliche Präsentationsmöglichkeit. Löse zunächst den Programmierfehler von der Figur **Cubi** auf. Die Steuerung nach links und rechts ist vertauscht. Bitte ein Kind, den Fehler zu erklären. Woran hat es ihn erkannt? Wie kann der Fehler behoben werden?

Danach ist es Zeit für ein Quiz! Wer hat das Cubi-Level **Wüstenjagd** ganz genau unter die Lupe genommen und alle Regeln herausgefunden? Ziehe die Bühne so groß, damit die Programmierfläche verdeckt wird und die Schüler*innen nicht lücheln können. Bei Unsicherheiten könnt ihr die Situation der Frage in Cubi nachstellen.

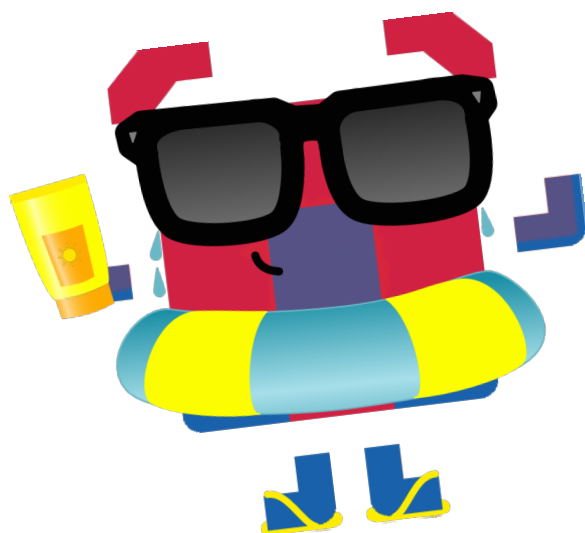
- Lässt sich Cubi in alle Richtung steuern? **Ja**

- Prallt Cubi immer vom Rand ab? **Nein, wenn man ganz häufig in eine Richtung klickt, verschwindet die Figur hinter dem Rand.**
- In welche Richtung dreht sich Cubi, wenn er den Rand berührt hat? **Zu der Wasserflasche**
- Wann hast du das Level gewonnen? **Wenn die Variable **Gefunden** größer als fünf ist, bzw. Cubi die Wasserflasche mindestens sechs Mal berührt hat.**
- Was passiert mit der Wasserflasche, wenn sie berührt wird? **Sie verschwindet und taucht fünf Sekunden später wieder auf.**
- Was passiert sonst noch im Level, wenn Cubi die Wasserflasche berührt? **Die Variable **Gefunden** wird um eins hochgezählt, die Variable **Zeit** um zehn und Cubi sagt: „Juhu!“**
- Wann ändert Cubi sein Kostüm? **Wenn die Wasserflasche angeklickt wurde oder, wenn er angeklickt wurde und eine zufällig generierte Zahl kleiner als drei ist. Die Wahrscheinlichkeit beträgt hierbei $\frac{1}{5} = 20\%$.**
- In welchem Muster bewegt sich die Wasserflasche? **Sie dreht sich in eine zufällige Richtung und geht dann bis zum Rand.**
- Wann erscheint ein Meteorit? **Wenn das Level verloren wurde.**
- Wie viele Sekunden hast du am Anfang Zeit, um die Wasserflasche zu fangen? **10 Sekunden**

Betone am Ende der Stunde noch einmal, dass selbst die erfahrensten Programmierer*innen Fehler beim Schreiben ihres Codes machen und dass es wichtiger ist, einen Fehler identifizieren zu können, als sie gar nicht erst zu machen. Beende die Stunde mit einem Ausblick auf das Thema der nächsten Stunde.

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

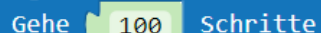
Der erste Baustein eines Blocks aus mehreren Bausteinen ist immer ein **Starbaustein**. Ein Programm einer Figur kann beliebig viele **Startbausteine** haben. **Startbausteine** zeichnen sich durch die Rundung am oberen Teil aus. Diese sagt aus, dass **Startbausteine** nicht an andere Bausteine angehängt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



Wenn Start ▶ geklickt

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



Gehe 100 Schritte

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.



Springe zu x: 0 y: 0

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.



Drehe rechts um 90 Grad

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



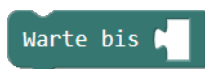
Der Baustein **Wenn dann** leitet eine Verzweigung ein. Oben an das **wenn** wird eine Bedingung angepuzzelt. Diese ist entweder **wahr** oder **falsch**. Wenn die Bedingung wahr ist, werden die Bausteine, die neben dem **dann** stehen ausgeführt. Ist die Bedingung falsch, also nicht erfüllt, werden 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.



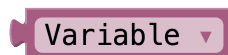
Mit diesem Baustein kann die Figur reagieren, wenn sie etwas berührt. Durch Klicken auf den kleinen Pfeil kann ausgewählt werden, ob die Figur auf den Rand oder eine andere Figur reagieren soll, wenn es noch weitere Figuren in dem Level gibt.

Mathe



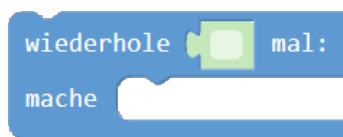
Mit diesem Baustein kann ein Bereich definiert werden, aus dem eine zufällige Zahl gewürfelt wird. Die Zahlen können beliebig hoch sein, die zweite Zahl muss allerdings größer als die erste Zahl sein.

Variablen

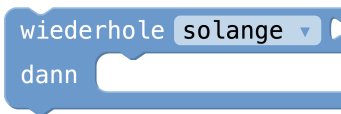


Über die Kategorie **Variablen** können Variablen erstellt, umbenannt und gelöscht werden. Variablen können oben links im Vorschauenfenster angezeigt oder wieder versteckt werden. **Setze Variable auf** ändert den Wert der Variable auf den angegebenen Wert. Mit **Erhöhe Variable um** und **Verringere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

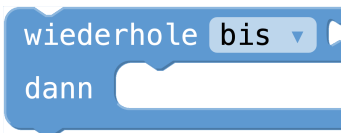
Schleifen



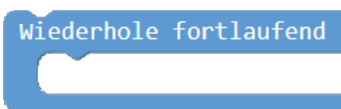
Mit der **Wiederhole ... mal**-Schleife können Bausteine, die in die Schleife eingefügt werden, wiederholt werden. Klicke auf die Zahl, um die Anzahl der Wiederholungen zu ändern.



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



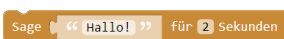
Mithilfe dieses Bausteins wechselt die entsprechende Figur das Kostüm und ändert so ihr Aussehen. Die Kostüme sind unterhalb der Bühne zu sehen. Hier wird zu dem Kostüm gewechselt, welches in dem Feld ausgewählt wurde.



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.

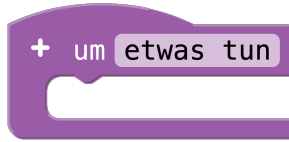


Mithilfe dieses Bausteins ändert die entsprechende Figur ihre Größe. Der Wert kann geändert werden. Je größer der Wert, desto größer wird die Figur.



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.

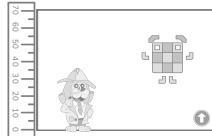


Name: _____ Datum: _____

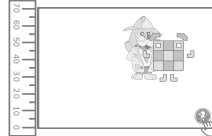
Detektivsuche

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

- Berühre ich Cubi?
- Berühre ich den Rand?
- Ist die Entfernung zum Rand oben kleiner als 20?
- Ist die Entfernung zum Rand oben größer als 30?
- Ist die Taste Pfeil nach oben gedrückt?



- Berühre ich Cubi?
- Berühre ich den Rand?
- Ist die Entfernung zum Rand oben kleiner als 20?
- Ist die Entfernung zum Rand oben größer als 70?
- Ist die Taste Pfeil nach oben gedrückt?



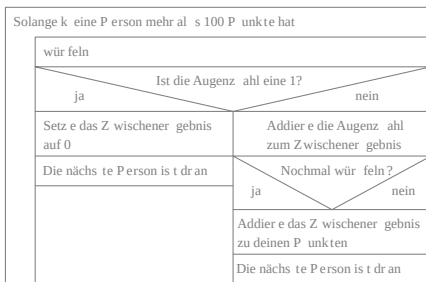
Aufgabe 2: Aussagen und Bedingungen können mit **und** oder **oder** verknüpft werden. Betrachte das obere Bild. Markiere auch die folgenden Aussagen als wahr (✓) oder falsch (x).

- Berühre ich den Rand und berühre ich Cubi?
- Ist die Entfernung zum Rand oben kleiner als 100 und größer als 10?

Alle Arbeitsblätter und Kopiervorlagen zu diesem Modul findest Du auf der Webseite von **IT 4 KIDS**: material.i4k.org/ab

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.

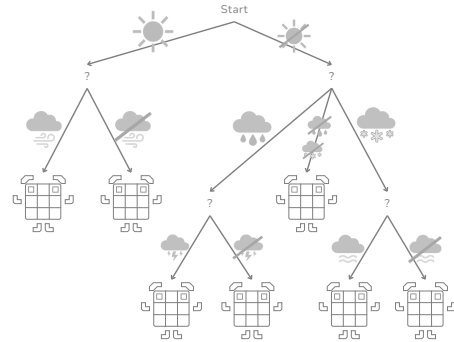
Dieses Material ist lizenziert unter CC BY-NC-SA 4.0. Weitere Informationen findest Du hier: <http://creativecommons.org/licenses/by-nc-sa/4.0/>



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?

CC BY-NC-SA 4.0. Weitere Informationen findest Du hier: <http://creativecommons.org/licenses/by-nc-sa/4.0/>

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: <input type="checkbox"/> ja <input type="checkbox"/> nein
	Fähigkeiten: _____
	Aufgabe: _____
[]	Name: _____
	Alter: _____ Steuerbar: <input type="checkbox"/> ja <input type="checkbox"/> nein
	Fähigkeiten: _____
	Aufgabe: _____

Dieses Material ist lizenziert unter CC BY-NC-SA 4.0. Weitere Informationen findest Du hier: <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Weitere Angebote von IT4Kids

Du willst weiter mit IT4Kids arbeiten? Neben einer Modulreihe zu den Themen **Sequenzen**, **Schleifen** und **Verzweigungen** mit der Zielgruppe **Orimiarstufe** 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 Fortbildungsangebot findest du auf unserer Webseite unter www.i4k.org/fortbildung. 😊

Du bist noch ungeschlüssig? 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. 💬

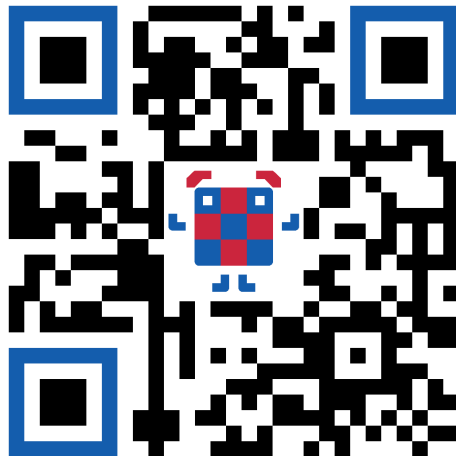
Wir freuen uns auf Dich! 😊



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