# Überblick über die Rennspiel-Einheit

## Grober Überblick

Die Einheit wurde konzipiert für den Informatikunterricht in **Klasse 7** an allgemeinbildenden Gymnasien in Baden-Württemberg und deckt in **10 bis 12 Unterrichtsstunden** das Thema **Algorithmen** ab.

Programmiert wird mit Scratch 3.

Die SuS lernen Scratch und verschiedene Befehle erst mit einfachen Programmen kennen, in denen eine Figur bewegt wird bzw. Formen gezeichnet werden und werden dann zu einer Rennspiel-Grundversion geleitet, die abschließend durch eigene Ideen erweitert werden soll. Letzteres kann auch als Grundlage für eine Benotung dienen.

Gerade bei der Programmierung ist es schön, möglichst viel Raum für Kreativität zu lassen. Dies ist in Kombination mit Notengebung, Lernzielen und großen heterogenen Klassen nicht immer einfach. Diese Einheit versucht einen Mittelweg zu finden. Wenn kein benotetes Projekt am Ende der Einheit stehen soll oder mehr als 10 bis 12 Stunden dafür eingeplant werden können, kann das Material als Anregung in kreativen Freiräumen verwendet werden.

Gestaltet ist die Einheit hauptsächlich durch Videos, die die SuS leiten. Wenn den SuS alle Videos zur Verfügung stehen, kann die Einheit auch vollständig zu Hause bearbeitet werden. Gerade für SuS der Stufe 7 lohnt es sich aber auch, die Videos im Unterricht, kombiniert mit Plenumsphasen, einzusetzen. D.h. die SuS bearbeiten die Aufgaben auf den Arbeitsblättern in individuellem Tempo, erhalten Erklärungen, Tipps und Lösungen per Videos, aber auch durch gegenseitige Unterstützung sowie durch die Lehrkraft. Wichtige Inhalte (z.B. Schleifen oder Variablen) werden im Plenum erarbeitet bzw. vertieft, wenn ein Großteil der Klasse diese Inhalte auch für die anstehenden Aufgaben benötigt und gerade in einem Video kennen gelernt hat. Zusätzlich können Checklisten eingesetzt werden, mit denen die SuS ihre Fortschritte dokumentieren.

## Vorbereitung der Einheit

Programmiert wird online mit Scratch 3: `https://scratch.mit.edu/`

### Projekte speichern mit Scratch-Accounts

Damit die SuS ihre Projekte abspeichern und weiter bearbeiten können, gibt es verschiedene Möglichkeiten, wobei sich Variante 2b als besonders praktikabel herausgestellt hat.

1. Projekte können über *Datei – Hochladen auf deinem Computer* auf den Computer bzw. auf einen USB-Stick o.ä. exportiert und zur Fortsetzung des Projekts wieder importiert werden.
2. Wer einen Scratch-Account hat, kann die Projekte online speichern über *Datei – Jetzt speichern*.
   a.) Um einen Account anzulegen benötigen SuS entweder eine Mailadresse und alle SuS legen selbst einen Account an oder
   b.) die Lehrkraft legt einen Lehrkräfteaccount und eine Scratch-Klasse an. Wie diese schöne Möglichkeit funktioniert, wird von App Camps ausführlich beschrieben unter:
   `https://drive.google.com/file/d/0BzMVvLOySsXMLTVBbFhSM3ljNjQ/view`
   Diese Variante vereinfacht gegenseitiges Ausprobieren der Projekte innerhalb der Klasse und das Bearbeiten von Aufgaben zu Hause. Außerdem kann sich die Lehrkraft mit wenig Aufwand einen Einblick in Projekte verschaffen und relativ schnell einen Überblick zum Stand der Projekte gewinnen. Wichtig ist, dass die SuS sich mit Pseudonymen oder wenigstens nur mit Vornamen anmelden.

**Hausaufgaben:** Gerade, wenn die Projekte online gespeichert werden, kann man Teile aus der unten vorgeschlagenen Stoffverteilung auch als Hausaufgabe aufgeben. Dann kann manches im Unterricht kürzer behandelt werden, sodass am Schluss mehr Zeit für Erweiterungen des Rennspielprojektes und/oder das gegenseitige Testen der Spiele besteht.

## Die Videos

Die begleitenden Videos befinden sich in einer Youtube-Playliste unter
https://www.youtube.com/playlist?list=PLWF8TDS-FUzWFCmMD4TP_z_yBk0h2b16f

Wenn Sie den SuS nicht alle Videos zur Verfügung stellen wollen, können Sie die Videos zu einer eigenen Playlist zusammenfügen oder einzeln z.B. in Moodle einbinden. Falls Sie die Videos offline zur Verfügung stellen wollen, können Sie diese auch mit geeigneten Tools herunterladen. Bitte geben Sie die Videos in diesem Fall nicht an andere Personen als Ihre SuS weiter.

Die Videos zu dieser Einheit enthalten neben Erklärungen und Tipps auch kleine Aufgaben und Lösungen, damit sie auch für eine Fernlernphase geeignet sind. In dem Fall, dass am Ende der Einheit ein benotetes Rennspielprojekt stehen soll, ist es wichtig, die Videos, die den SuS zur Verfügung gestellt werden, zumindest bei den Rennspielerweiterungen einzuschränken.

## Ziele und Bezug zum Bildungsplan

Im Bildungsplan wird Programmieren als eine »Realisierung von Ideen in Software« beschrieben, die als »schöpferischer und produktiver Prozess ein wesentlicher Bestandteil des Informatikunterrichts« ist.[1] Mit Problemen umzugehen und sie strategisch zu lösen ist ein wichtiger Bestandteil der Allgemeinbildung, da sich die SuS hierdurch intelligentes Wissen erarbeiten, das sie auch außerhalb des schulischen Kontexts anwenden können.[2] Im Informatikunterricht »entwerfen [SuS] Problemlösungen, die auf grundlegenden Programmierbausteinen basieren, und erfahren so, dass die Lösung nicht in den Bausteinen selbst, sondern hauptsächlich in der Art und Weise ihrer Anordnung liegt«.[3]

Ein Ziel dieser Unterrichtseinheit soll es sein, dieses algorithmische Problemlösungsdenken zu fördern. Dies soll im Sinne einer »Fundamentalen Idee«[4] auch unabhängig von der eingesetzten Programmiersprache bzw. -umgebung erfolgen. Konkreter soll den SuS *Konzeptwissen* zu Algorithmen vermittelt werden, ohne dabei die Handlungsorientierung zu vernachlässigen.[5]

Im Bildungsplan[6] werden folgende Ziele genannt, die von dieser Einheit abgedeckt werden: Die SuS können

(a) die algorithmischen Grundbausteine Anweisung, Sequenz, Schleife/Wiederholung, Verzweigung und Bedingung erläutern;

(b) Algorithmen als Verknüpfung von Anweisungen und Kontrollstrukturen beschreiben;

(c) Variablen als änderbaren Wertespeicher (z. B. als Speicher für Punktestand, Rundenzähler in Spielen etc.) erläutern;

(d) Algorithmen zu gegebenen Problemstellungen entwerfen;

(e) Algorithmen in einer geeigneten (z. B. visuellen) Programmierumgebung implementieren und dabei Variablen und algorithmische Grundbausteine zielorientiert anwenden;

(f) in grafischer Form (z. B. als Flussdiagramm) dargestellte Algorithmen erklären;

(g) Codeabschnitte schrittweise untersuchen und deren Wirkung interpretieren;

Da die Zeit für die Einheit knapp bemessen ist, kann nicht alles in dem Maße vertieft werden, wie es wünschenswert wäre in Anbetracht dessen, dass die SuS in den nachfolgenden Schuljahren nur wenig oder gar kein Informatik mitbekommen. Bei diesen Rahmenbedingungen ist es um so wichtiger, den SuS »Lust auf mehr« zu machen und ihr Interesse zu wecken. Deshalb sollen den SuS von Anfang an schnelle Erfolgserlebnisse ermöglicht werde. Sie sollen aber auch ein Stück weit kurzzeitig verzweifeln dürfen, knobeln müssen und die Theorie und Vokabeln lernen müssen. All dies gehört auf dem Weg zum eigenständigen Programmieren dazu.

[1] Vgl. Ministerium für Kultus, Jugend und Sport: Bildungsplan 2016, Informatik Klasse 7, S. 10.
[2] Vgl. Bovet: Wissenserwerb und Problemlösen, 221–236.
[3] Vgl. Ministerium für Kultus, Jugend und Sport: Bildungsplan 2016, Informatik Klasse 7, S. 10.
[4] Vgl. Hartmann u. a.: Informatikunterricht planen und durchführen, S. 31 ff.
[5] Zu Konzept- und Produktwissen vgl. ebd., S. 6 f.
[6] Ministerium für Kultus, Jugend und Sport: Bildungsplan 2016, Informatik Klasse 7, S. 19.

## Mögliche Stoffverteilung

| Std. | Thema | grober Verlauf und Ziele | Material |
|---|---|---|---|
| (0) | LightBot | ▪ `https://lightbot.com/flash.html` (benötigt Flash)<br>▪ Der LightBot kann als Einsteig in die Programmierung verwendet werden und macht in allen Altersstufen Spaß, ist für diese Einheit aber nicht notwendig. Er eignet sich übrigens auch hervorragend für Vertretungsstunden.<br>▪ Die SuS lernen, dass Befehle in angegebener Reihenfolge nacheinander abgearbeitet werden; algorithmisches Denken. | LightBot_Loesungen.pdf |
| 1 | Einstieg in Scratch | ▪ Kennenlernen der Scratch 3 Entwicklungsumgebung<br>▪ Einführung der Arbeitsform mit ABs, Videos und Checklisten<br>▪ Erste Anweisungen und Anweisungssequenzen<br>▪ Relative und absolute Bewegungen im Scratch-Koordinatensystem<br>▪ AB 1 und AB 2 werden gemeinsam besprochen. Die Lösungen werden nicht komplett in den Videos besprochen. Sie finden die Lösungen im Ordner *Scratch-ABs_Loesungen*. | Kopfhörer<br>AB 1<br>AB 2<br>Check 1<br>Videos |
| 2+3 | Malen mit Hilfe von Zähl- und Endlosschleifen | ▪ Die SuS fügen eine neue Figur und ein Bühnenbild ein.<br>▪ Zählschleifen werden verwendet, um Bewegungen besser verfolgbar zu machen: Viele kleine Schritte wiederholen statt einem großen.<br>▪ Der Code zum Malen von Quadraten und anderen gleichseitigen Formen wird mit Zählschleifen vereinfacht.<br>▪ Durch die Verwendung von Farben, Strichdicken, verschiedenen Winkeln und Endlosschleifen entstehen Kunstwerke. | Kopfhörer<br>AB 3<br>Check 1<br>Videos |
| 4 | Wiederhole-bis-Schleifen | ▪ Rückblick auf die Zähl- und Endlosschleifen<br>▪ Einführung der wiederhole-bis-Schleifen und zugehörigen Bedingungen in Scratch<br>▪ Formulierung der wiederhole-bis-Schleife als vollständige umgangssprachliche Sätze<br>▪ Sicherung zu Schleifen auf AB 4: *Wenn Anweisungen oder ganze Sequenzen wiederholt werden sollen, nutzt man Schleifen. In Scratch gibt es drei Schleifentypen: die Zählschleife, die Wiederhole-bis-Schleife und die Endlosschleife.*<br>▪ Einsatz der Wiederhole-bis-Schleife in kleinem Katz-Maus-Spiel | Kopfhörer<br>AB 4<br>Check 2<br>Videos |
| 5 | Programm-ablauf-pläne | ▪ Die SuS lernen mit Programmablaufplänen eine alternative Darstellung von Algorithmen kennen.<br>▪ Übertragung algorithmischen Denkens auf Alltagssituationen<br>▪ Ein Ablaufplan (Frühstück, mögliches Tafelbild: siehe Anhang) wird im Plenum und ein weiterer Ablaufplan wird in GA erarbeitet und dann auf Scratch (auf Basis von MausZuKaese.sb2) übertragen.<br>▪ Aufbau für AB5, Aufgabe 2: siehe unten | Kopfhörer<br>AB 5<br>Check 2<br>Videos<br>**Süßes**<br>MausZuKaese.sb2 |
| 6 | Rennspiel-Steuerung | ▪ Das Ziel »Rennspiel« wird formuliert.<br>▪ Einfügen einer Rennfigur und Malen einer Strecke mit nur zwei Farben. (Die Farben werden später zur Unterscheidung von Strecke und Streckenbegrenzung eingesetzt.)<br>▪ Programmieren einer ereignisbasierten Steuerung.<br>▪ Erweiterung der Steuerung durch eine Endlosschleife die für eine grundsätzliche Fortwärtsbewegung sorgt. (Diese Bewegung wird später durch eine Geschwindigkeitsvariable variiert.)<br>▪ Die SuS testen gegenseitig Strecke und Steuerung.<br>▪ Ausblick/Beginn mit Variablen (siehe nächste Stunde) | Kopfhörer<br>AB 6<br>AB 7<br>Check 2<br>Videos<br>Grafiken[7] |

[7] Grafiken für weitere Rennfiguren, die Sie den SuS zur Verfügung stellen können, finden Sie im Material und als Übersicht unten.

| Std. | Thema | grober Verlauf und Ziele | Material |
|---|---|---|---|
| 7 | Variablen | ■ Möglicher Einstieg im Plenum: Variablen kann man sich als beschriftete Box (Platzhalter/Speicherplatz) vorstellen. Der Wert in dieser Box wird initialisiert und kann ausgelesen und verändert werden. Z.B. könnten zwei SuS für eine Minuten um jeweils einen Tisch rennen. Es wird für beide jeweils eine Variable (Box) zur Verfügung gestellt und beschriftet (Deklaration), die anfangs leer ist (Initialisierung mit 0). Zwei Helfer:innen werfen für jede Runde ein Bonbon in die entsprechende Box (Erhöhen des Wertes um 1). Nach einer Minute werden die Werte ausgelesen und verglichen. Ein:e 3. S versucht den Rekord zu knacken: Für jede Runde wird ein Bonbon aus der Box genommen, bis diese leer ist. Die Zeit wird gestoppt. Letzteres funktioniert auch, wenn wir nach dem ersten Wettbewerb nicht geschaut haben, was der Wert ist, und entspricht einer Wiederhole-bis-Schleife.<br>■ Sicherung zu Variablen: *Variablen sind Platzhalter für Werte und können in Scratch über »Neue Variable« eingeführt (deklariert) werden. Jede Variable benötigt einen sinnvollen Namen und sollte zu Beginn des Programms einen Startwert zugewiesen bekommen (Initialisierung). Der Wert einer Variable kann zur Laufzeit des Programms ausgelesen und verändert werden.*[8]<br>■ Optional: Einführung von Variablen durch Aufgabe, das Malen einer Spirale zu programmieren. (AB 7, Aufgabe 1)<br>■ Rennspiel: Beschleunigen und Bremsen mit Hilfe einer Variable. | Kopfhörer<br>AB 7<br>Check 2<br>Videos<br>(**Boxen**) |
| 8 | Verzweigungen | ■ Möglicher Einstieg im Plenum: Erweiterung des Frühstück-Ablaufplans durch eine Entscheidung »Willst du Müsli?«. Falls ja, folgt der bekannte Müsli-Algorithmus. Falls nein, »schmiere Brot«, »ist noch Brot da?«, »nimm einen Bissen« etc. analog zum Müsli-Algorithmus.<br>■ Eine Sicherung zu Verzweigungen ist auf AB 8 bereits abgedruckt.<br>■ Mit Hilfe von Verzweigungen wird beim Rennspiel das Verlassen der Strecke verhindert und die Geschwindigkeit begrenzt.<br>■ Abschließende Sicherung zu Algorithmen: *Ein Algorithmus ist eine exakte Vorschrift, die in einfachen Schritten beschriebt, wie (z.B. ein Computer) vorgehen soll. Ein Algorithmus ist zusammengesetzt aus Anweisungen/Anweisungssequenzen, Schleifen und Verzweigungen.* | Kopfhörer<br>AB 8<br>Check 2<br>Videos |
| 9–12 | Projekt erweitern | ■ Bis hierhin sollten alles SuS eine **Grundversion** des Rennspiels haben. Die Elemente der Grundversion werden auf dem **AB Projekt** nochmal aufgelistet.<br>■ Auf dem AB Projekt sind außerdem Ideen für Erweiterungen des Rennspiels aufgelistet. Je nachdem ob eine Unterstützung der SuS durch Videos gewünscht ist, gibt es das AB einmal mit und einmal ohne Hinweisen zu Videos.<br>■ Bevor das AB ausgegeben wird, können im Plenum Ideen für mögliche Erweiterungen gesammelt und Kriterien diskutiert werden, die eine gute Erweiterung ausmachen.<br>■ Nach Möglichkeit gegenseitiges Testen der Spiele mit anschließendem Feedback und Zeit, das Feedback einzuarbeiten. | Kopfhörer<br>ProjektMit<br>ProjektOhne<br>Videos |

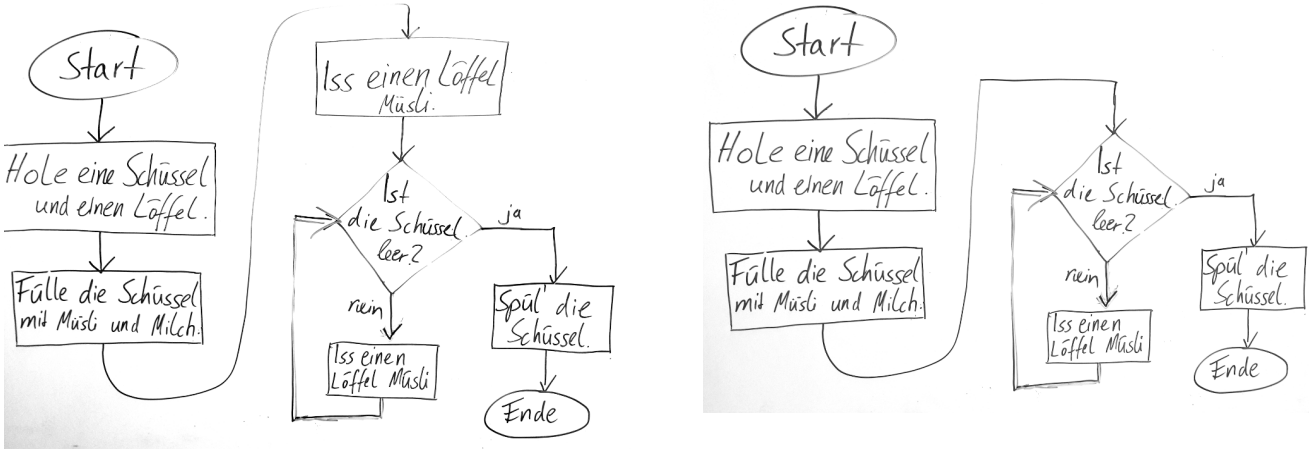
[8] In Scratch 3 können Variablen nur für eine Figur oder global sichtbar sein. Dies wird bei Einführung weiterer Rennfiguren für einen Zwei-Spieler-Modus interessant, kann aber dann bzw. auf Nachfrage geklärt werden.

# Anhang

## Weitere Anregungen

Zur Wiederholung zu Beginn der Stunden oder nach Einführung neuer Elemente lassen sich Werkzeuge wie Plickers, Kahoot o.ä. mit Fragen der Art »Was tut die Katze bei Ausführung von folgendem Code?« einsetzen.

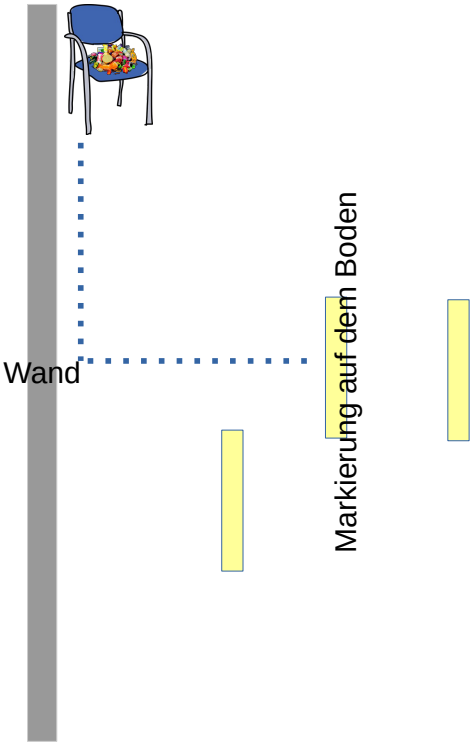
## Mögliches Tafelbild zu AB5




*Das linke Bild entsteht gemeinsam mit den SuS, die man etwas bremsen muss, damit der Ablaufplan nicht zu groß wird. Das rechte Bild entsteht aus dem linken nach Diskussion der Frage, wie man den Ablauf mit weniger Befehlen erreichen kann.*

## Aufbau zu AB5, Aufgabe 2

**Aufgabe:** *Ein Roboter soll die Süßigkeiten aufheben, die an der Projektions-Wand auf einem Stuhl liegen. Der Roboter kennt nur die Befehle »Gehe einen Schritt vor«, »Drehe dich um 90 Grad nach rechts« und »Hebe die Süßigkeiten auf«. Au- ßerdem kann er mit seinen Sensoren prüfen (entscheiden), ob er die Wand berührt und ob er beim Stuhl mit den Süßigkeiten angelangt ist. Entwickle einen Programmablaufplan für ein Programm, das den Roboter von jeder der markierten Positionen im Klassenraum zu den Süßigkeiten führt. Teste es mit deinem*r Partner*in, indem ihr den Roboter spielt und euch streng an euren Ablaufplan haltet.*



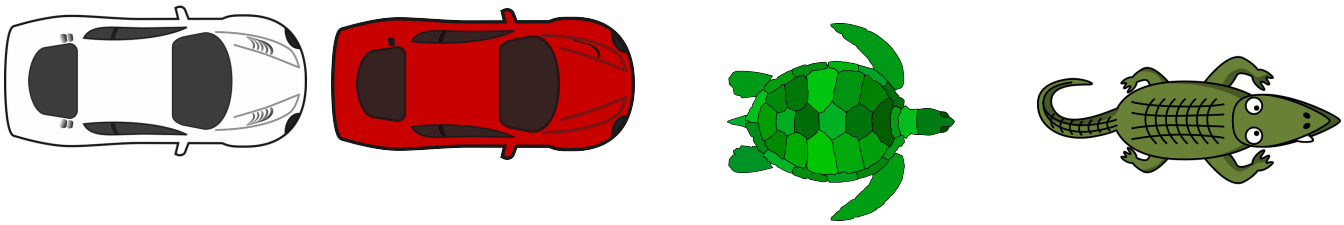

**Aufbau**

- Mindestens zwei Positionen im Klassenzimmer werden auf dem Boden markiert.
- An einer Wand steht ein Stuhl mit Süßigkeiten.
- Die Markierungen und der Stuhl müssen so positioniert sein, dass man von jeder Markierung zum Stuhl kommt, indem man geradeaus bis zur Wand läuft, sich nach rechts und dann bis zum Stuhl läuft.

Der Grund für mehr als eine Markierung: Durch die unterschiedlich weit entfernten Markierung wird der Vorteil der wiederhole-bis-Schleife gegenüber von Zählschleifen deutlich.

Einen möglichen Ablaufplan finden Sie im Ordner *Scratch-ABs_loesungen*.

## Zusätzliche Rennfiguren



*Quellen, jeweils frei nutzbar (CC0): Alligator – https://pixabay.com/p-47725/; Schildkröte – https://pixabay.com/p-294522/; Rennwagen – https://pixabay.com/p-296772*

## Rechtliches



## Literaturverzeichnis

Bovet, Gislinde: *Wissenserwerb und Problemlösen*, in: Volker Huwendiek (Hrsg.): Leitfaden Schulpraxis: Pädagogik und Psychologie für den Lehrberuf, Cornelsen Vlg Scriptor, 2014, S. 205 –237.

Hartmann, Werner, Michael Näf und Raimond Reichert: *Informatikunterricht planen und durchführen*, Berlin Heidelberg New York: Springer-Verlag, 2007.

Ministerium für Kultus, Jugend und Sport: *Bildungsplan 2016 – Sekundarstufe I – Informatik (Aufbaukurs Informatik Klasse 7)*, Stuttgart, 2017, URL: http://www.bildungsplaene-bw.de/,Lde/LS/BP2016BW/ALLG/SEK1/INF7.