

Entwurf einer Unterrichtsstunde zur Einführung von Sensoren und bedingten Anweisungen in der Entwicklungsumgebung Scratch

Autor/in: *Sofia Miggelbrink*

Matrikelnummer: ██████████

**Thema der
Unterrichtseinheit:**

Erweiterung einer Aquariumsszene in Scratch um eine Farbveränderung der Figur abhängig von der Hintergrundfarbe zur Einführung von Sensoren und bedingten Anweisungen

Thema der Reihe:

Entwicklung eines Spiels in der Entwicklungsumgebung Scratch zur Einführung von Befehlssequenzen und der grundlegenden Kontrollstrukturen Verzweigung und Schleife

Klasse:

4. Klasse

Zeitumfang (Min.):

90

Inhaltsverzeichnis

Schriftliche Planung des Unterrichts

- | | |
|--------------------------------------|------|
| 1. Ziele und angestrebte Kompetenzen | S. 1 |
| 2. Didaktische Schwerpunkte | S. 2 |
| 3. Artikulationsschema | S. 8 |

Literaturverzeichnis S. I

Anhang S. II

Versicherung und Verwertungsrechte S. X

Schriftliche Planung des Unterrichts

1. Ziele und angestrebte Kompetenzen

Ein Stundenziel/Kernanliegen mit Indikator:

Die Schülerinnen und Schüler sollen bedingte Anweisungen mit Sensoren implementieren können. Sie zeigen dies, indem sie das Skript der Figur einer Aquariumsszene in Scratch so erweitern, dass diese fortlaufend abhängig vom Hintergrund die Farbe verändert.

Drei bis fünf Teilziele mit Indikator:

- Die SuS erkunden die „Fühlen“-Blöcke in Scratch, indem sie bestimmen und beschreiben, dass diese Blöcke den Zustand der Szene abfragen und ein vom Zustand abhängiges Ergebnis zurückgeben.
- Die SuS untersuchen vorgegebene Skripte mit sensorabhängigen Verzweigungen auf den Kontrollfluss des Programms, indem sie jeweils angeben, ob in Szenen bestimmte Befehle ausgeführt werden oder nicht.
- Die SuS ergänzen am PC die vorgegebene Szene mittels zweier Falls-Dann-Blöcke so, dass die Figur bei Ausführung des Skripts die Farbe (einmalig) an den Hintergrund anpasst.
- Die SuS ergänzen das Skript weiterhin mit einer Endlos-Schleife derart, dass die Bedingung zur Farbveränderung fortlaufend geprüft wird.

Hierdurch sollen folgende Kompetenzen laut Lehrplan/Empfehlungen gefördert werden (je eine inhaltsbezogene und prozessbezogene Kompetenz):

Die beschriebene Stunde, wie auch die Unterrichtsreihe, in die sie sich einfügt, fördert die in den Empfehlungen der Gesellschaft für Informatik beschriebenen Kompetenzen zur Modellierung und Implementierung einfacher Algorithmen.

Im Rahmen der Reihe erwerben die Schülerinnen und Schüler hierzu schrittweise Fähigkeiten in der Verwendung der „algorithmischen Grundbausteine Anweisung, Sequenz, Wiederholung, Verzweigung“ (GI17, S. 13). Sie erkennen, dass alle Programme aus diesen kleineren Bausteinen zusammengesetzt sind und analysieren, entwerfen und entwickeln eigene kleine Programme.

In dieser Unterrichtsstunde steht dabei die planungsvolle Erweiterung eines Programms um die neu eingeführten bedingten Anweisungen im Fokus. Dabei wird auch die prozessbezogene Kompetenz „Problemlösen/kreativ sein“ des Lehrplans Mathematik des Landes NRW für die Grundschule gefördert: die Schülerinnen und Schüler können die Aufgabe nicht ohne Weiteres lösen, sondern müssen systematisch vorgehen und Strategien entwickeln, um zu einer Lösung zu gelangen.

2. Didaktische Schwerpunkte

Die vorliegende Unterrichtsstunde richtet sich an eine vierte Klasse und soll innerhalb einer Projektwoche zum Thema Informatik stattfinden. Eine solche Projektwoche bietet den Schülerinnen und Schülern die Möglichkeit, sich entsprechend ihrer Interessen verschiedenen Arbeitsgruppen anzuschließen und sich über einen Zeitraum mehrerer Stunden intensiv mit einem Thema zu beschäftigen. Diese Ausgangslage kommt der Konzeption der Reihe entgegen, da die projekthafte Arbeit in der Informatik und Informatikdidaktik eine besonders prominente Rolle einnimmt (vgl. Hu06, S. 41f., S. 76ff.).

Da die vorliegende Unterrichtsplanung sich nicht auf eine konkrete Lerngruppe bezieht, können hier nur einige allgemeine Annahmen hinsichtlich der Lerngruppe getroffen werden. Bei der realen unterrichtlichen Umsetzung können Anpassungen notwendig sein. Es wird davon ausgegangen, dass für die Lerngruppe keine speziellen sprachlichen Voraussetzungen zu beachten sind, d. h., alle Schülerinnen und Schüler sind sprachlich imstande, die Aufgabenstellung zu verstehen. Weiterhin wird davon ausgegangen, dass die Schülerinnen und Schüler über grundlegende Kenntnisse und Kompetenzen im Umgang mit dem Computer verfügen. Sie können beispielsweise Programme öffnen und schließen, Tastatur und Maus bedienen und haben Erfahrung mit der in der Schule verwendeten Lernsoftware. Die Schülerinnen und Schüler haben vor Beginn der Projektwoche allerdings noch keine Erfahrung mit Programmierung und (grafischen) Programmierumgebungen sowie Algorithmen und Kontrollstrukturen.

Da die Schülerinnen und Schüler im Rahmen der Projektwoche die Möglichkeit haben, sich entsprechend ihrer persönlichen Interessen und Neigungen für ein Thema zu entscheiden, ist davon auszugehen, dass die Lerngruppe überdurchschnittlich lernbereit und intrinsisch motiviert ist. Hierzu trägt auch das mutmaßlich attraktive Ziel der Projektarbeit bei, ein eigenes kleines Computerspiel zu entwickeln. Dennoch muss davon ausgegangen werden, dass die Schülerinnen und Schüler unterschiedliche Lernvoraussetzungen besitzen; unterschiedliche Vorerfahrungen der Kinder mit digitalen Medien vonseiten des Elternhauses bedeuten eine eher heterogene Lerngruppe. Hierauf wird bei der Konzeption von Hilfsangeboten zu achten sein. Zwar kommen die Kinder aus verschiedenen Klassen, kennen sich aber untereinander durch den Unterricht in klassenübergreifenden Fächern. Sie können gut miteinander in Partnerarbeit arbeiten.

Im Folgenden sind die Stundenthemen der Unterrichtsreihe zum Thema „Spielprogrammierung mit Scratch“ in der Projektwoche aufgeführt:

**Lehr- und
Lernausgangs-
lage der SuS**

**Aufbau der Un-
terrichtsreihe**

Std.	Thema der Stunde
1	Die ersten Schritte: Gestaltung einer Scratch-Szene und die Blöcke der Kategorien „Bewegung“, „Aussehen“ und „Klang“
2	Skripte als Befehlssequenzen: Entwurf kurzer Befehlsfolgen zur Bewegung und Gestaltung einer Figur
3	Noch mal von vorn: Die Zählschleife und die Endlosschleife als Beispiele für Kontrollstrukturen zur Wiederholung
4	Mit Gefühl: Bedingte Anweisungen mit Bedingungsblöcken der Kategorie „Fühlen“ zur Farbveränderung einer Figur
5	Kombination von „Fühlen“-Blöcken mithilfe der Operatoren Und, Oder, Nicht
6	Entwurf und Planung des „Aquarium-Spiels“

Zu Beginn der vierten Stunde können die Kinder somit bereits mit der Entwicklungsumgebung Scratch umgehen; sie haben eigene kleine Szenen entworfen, indem sie die entsprechenden Hintergründe ausgewählt und Figuren hinzugefügt haben. Diese Figuren haben sie bewegt und verändert durch Anwahl einzelner Befehlsblöcke aus den Kategorien „Bewegung“ und „Aussehen“. Die Schülerinnen und Schüler können diese Blöcke zu Skripten zusammenfügen und zwei grundlegende Schleifen benutzen, um die wiederholte Ausführung von Blöcken zu implementieren.

Die Perspektive der Unterrichtsreihe ist es, dass die Schülerinnen und Schüler am Ende der Projektwoche ein kleines Aquarium-Spiel mit drei bis vier Figuren (z. B. Fisch, Seestern, Tintenfisch, ...) entwerfen, die sich bei Berührung des Mauszeigers oder beim Drücken bestimmter Tasten bewegen, verändern oder Töne von sich geben.

Bedingte Anweisungen und Verzweigungen gehören neben den Sequenzen und Wiederholungen zu den algorithmischen Kontrollstrukturen (Hu07, S. 183) Diese sind von überragender Bedeutung in der Programmierung und bei der Entwicklung von Algorithmen, weswegen sie A. Schwill als fundamentale Idee der Informatik im Bereich der Masteridee Algorithmisierung klassifiziert (Sc93, S. 22). Folglich sind Kontrollstrukturen grundlegende Bausteine der allermeisten Programme und durch diese in Computern und anderen elektronischen Alltagsgegenständen allgegenwärtig.

**Begrenzte
Sachanalyse
des U.-
gegenstandes**

Schwill fasst die genannten Strukturen *Anweisung*, bei der bedingungsabhängig eine Befehlssequenz ausgeführt wird oder nicht, und *Verzweigung*, bei der bedingungsabhängig eine von zwei Befehlssequenzen ausgeführt wird, in der Algorithmik unter dem Begriff der *Alternative* zusammen [ebd.]. Die genaue Umsetzung dieser Kontrollstruktur(en) unterscheidet sich je nach Programmiersprache; zum Teil existieren auch eigene Anweisungen für mehrfache Verzweigungen, etwa durch die Anweisungen *if*, *if-else* und *switch-case* in der Programmiersprache Java (UI 10, S.164ff.).

Eine deutlich einfachere syntaktische Struktur haben die Kontrollstrukturen in der Programmiersprache Scratch. In dieser grafischen, nach didaktischen Maßgaben konzipierten Programmiersprache wird die Aneignung der Syntaktik deutlich entlastet, wodurch es bereits Schülerinnen und Schülern der Primarstufe möglich wird, sich mit Fragen der Programmierung auf semantischer und pragmatischer Ebene auseinanderzusetzen (Wa13, S. 2). In Scratch existieren die Befehlsblöcke *falls-dann* und *falls-dann-sonst*. Im Sinne einer didaktischen Reduktion wurde in der vorliegenden Stunde die Entscheidung getroffen, nur den ersten dieser beiden Blöcke mit den Schülerinnen und Schülern einzuführen, da dieser ausreichend ist, um das grundlegende Konzept einzuführen und die Funktion der Sensoren („Fühlen-Blöcke“) zu thematisieren. Außerdem genügt dieser Anweisungsblock für das Ziel der Unterrichtsreihe, das Aquarium-Spiel zu programmieren.

Weiterhin wurde die Entscheidung getroffen, die Begriffe „Bedingte Anweisung“ und „Verzweigung“ nicht einzuführen, da sie die Kinder nach Ansicht der Autorin verwirren könnten. Stattdessen wird vom Steuerungsblock „falls-dann“ (ebenso wie vom Steuerungsblock „wiederhole fortlaufend“) gesprochen, da die Blöcke ebenso beschriftet sind. Dabei wird ausgenutzt, dass die Bezeichnung der Blöcke in Scratch auch aufgrund der deutschen Beschriftung für die Kinder weitgehend selbsterklärend ist.

Die bedingten Anweisungen werden zusammen mit den Sensoren eingeführt, also den Blöcken der „Fühlen“-Kategorie, die links und rechts spitz zulaufen. Bei diesen handelt es sich um Funktionen mit einem Wahrheitswert, also solchen, die eine Boolesche Variable mit dem Wert „wahr“ oder „falsch“ zurückgeben. In vielen Programmiersprachen wird für diese ein eigener Datentyp zur Verfügung gestellt, der Wahrheitswerte speichert (vgl. etwa UI10) und es werden eine Anzahl an Operatoren zur Verknüpfung solcher Booleschen Variablen zur Verfügung gestellt, üblicherweise „und“, „oder“ und „nicht“. Die Anwendungsbereiche der Booleschen Algebra in der Informatik sind vielfältig und berühren die Inhaltsfelder Information und Daten, Algorithmen und Informatiksysteme, weswegen die GI im Unterricht der Sekundarstufe I eine explizite Behandlung empfiehlt, etwa im Kontext von Tabellenkalkulation (vgl. GI08, S. 29f.).

Zwar verfügt auch Scratch über die grundlegenden Booleschen Operatoren, erscheint aber nicht geeignet für die Einführung oder Thematisierung von Wahrheitswerten. Aus Gründen der didaktischen Reduktion werden diese Details mit den Schülerinnen und Schülern daher noch nicht thematisiert. Stattdessen erfahren die Kinder die „Fühlen“-Blöcke als Anweisungen, die eine „Antwort“ erfragen, nämlich entweder ja oder nein. Auf diese Weise werden die Kinder erstmals an die zur Verwendung von bedingten Anweisungen notwendigen Wahrheitswerte propädeutisch herangeführt, ohne diese bereits explizit zu behandeln.

Durch die beschriebenen Reduktionen kann das Thema der bedingten Anweisungen und Sensoren innerhalb einer Unterrichtsstunde behandelt werden. Der Fokus liegt auf der praktischen Umsetzung.

Der Inhalt der Stunde lässt sich mithilfe der Empfehlungen für Kompetenzen der GI legitimieren. Im Inhaltsbereich „Algorithmen“ heißt es dort als Zielvorgabe für das Ende der vierten Klasse, Schülerinnen und Schüler sollten „mit den algorithmischen Grundbausteinen *Anweisung*, *Sequenz*, *Wiederholung* und *Verzweigung*“ (GI17, S. 13) Algorithmen entwerfen, realisieren und testen. Weiterhin beschreibt die GI die algorithmischen Grundbausteine als „grundlegende Prinzipien einer Ablaufbeschreibung, die in verschiedenen Programmiersprachen unterschiedlich formuliert werden“ (ebd.). Das Entwerfen von Programmen auch mithilfe dieser Elemente sei „ingenieurartig konstruktiv und kreativ“ (ebd.). Ein allgemeinbildender Unterricht, der diese Elemente einführt, muss sich folglich an den übergeordneten semantischen und pragmatischen Prinzipien orientieren, die allen Programmiersprachen gemein sind, statt einen Fokus auf spezielle syntaktische Details zu legen. Dies gelingt mit der Einführung des Steuerungsblocks für die bedingte Anweisung in der syntaktisch einfachen Sprache Scratch.

**Legitimation
des Vorhabens
durch
curriculare
Vorgaben**

Die prozessbezogene Kompetenz „Modellieren und Implementieren“ wird in der vorgestellten Stunde besonders angesprochen, da die Schülerinnen und Schüler eine vorgegebene Idee in ein Skript übersetzen sollen. Die Kinder setzen also ein Modell „mit geeigneten Werkzeugen um“ (GI17, S. 9). Dafür müssen diese Werkzeuge, z. B. die Blöcke und die Programmierumgebung in Scratch, erst einmal eingeführt und verstanden werden. Durch die oben beschriebene Notwendigkeit der kreativen Lösungsfindung wird auch die prozessbezogene Kompetenz „Problemlösen/kreativ sein“ des Lehrplans Mathematik NRW für die Grundschule gefördert: Die Schülerinnen und Schüler „bearbeiten Problemstellungen. Dabei erschließen sie Zusammenhänge, stellen

Vermutungen an, probieren systematisch, reflektieren und prüfen, übertragen, variieren und erfinden“ (MSW08, S. 57). Zum Problemlösen zählt auch die Anwendung geeigneter Algorithmen (ebd., S. 59). Mit der inhaltlichen Orientierung in der Algorithmik ist die Stunde somit im Bereich der Grundschulmathematik gut aufgehoben, auch wenn ein direkter Anschluss an die Inhaltsbereiche des Grundschullehrplans nicht möglich ist.

Weiterhin wird auch in der inhaltsbezogenen Kompetenz „Sprache und Automaten“ der GI benannt, dass Schülerinnen und Schüler Sprache auch zum Programmieren verwenden (vgl. GI17, S. 14). Dies wird durch die Anwendung der Programmiersprache Scratch gefördert. Außerdem wird auch das Darstellen/Kommunizieren durch die Arbeit und Absprache mit einem Partner angesprochen (vgl. MSW08, S. 60). Allerdings haben diese Kompetenzen eine eher untergeordnete Rolle in der Unterrichtsstunde.

Die heutige Allgegenwart von Informationssystemen macht für die Schülerinnen und Schüler den Umgang mit PCs, Smartphones, Tablets und Co. zu einer Selbstverständlichkeit (Hu07, S. 59). Ein verantwortungsbewusster, planungsvoller Umgang mit diesen Medien erfordert allerdings ein tiefergehendes Verständnis der den Systemen zugrundeliegenden Mechanismen. Eine bloße Medienerziehung ist hier nicht ausreichend, konstatiert Hubwieser (ebd., S. 60ff.). Deswegen ist es sinnvoll, bereits in der Grundschule grundlegende informatische Kompetenzen zu vermitteln. In diesem Zusammenhang betont Hubwieser auch die überragende Bedeutung informatischer Kenntnisse auf einem zunehmend durch die Digitalisierung und Automatisierung geprägten Arbeitsmarkt sowie die Anforderungen, die den Kindern heute und in Zukunft von der Informationsflut abverlangt werden, wie sie etwa das Internet täglich produziert (ebd., S. 58f.).

Im Sinne eines allgemeinbildenden Informatikunterrichts (vgl. Hu07, S. 62) muss der Unterricht über die Vermittlung reiner Programmierkenntnisse hinausgehen. Mit der bedingten Anweisung lernen die Schülerinnen und Schüler in der entworfenen Stunde ein fundamentales logisches Konzept kennen, dass auch in vielen anderen Bereichen Anwendung findet: vom (mathematischen) Algorithmus über die Bauanleitung bis hin zum Kochrezept.

Durch die Einführung informatischer Konzepte schon in der Grundschule mittels Scratch wird den Jungen und Mädchen zudem ein leichter, kooperativer und unterhaltsamer Zugang zum Fach ermöglicht, was insbesondere den Mädchen zugutekommt. Denn dadurch wird einer Bildung von Geschlechterstereotypen vorgebeugt, die Computer und Programmierung als Männerdomäne ausweisen. Dies wird mit zunehmendem Alter der Kinder vor allem in der Sekundarstufe zum Problem, bemerkt Humbert (Hu06, S. 171ff.). Insofern leisten Veranstaltungen wie die hier skizzierte Projektwoche einen Beitrag zur Geschlechtergerechtigkeit.

Die Stunde beginnt mit einem kleinen Wiederholungsquiz. Dieses erfüllt die Funktion, den Schülerinnen und Schülern das bereits erworbene Wissen zu Scratch wieder in Erinnerung zu rufen und zu Stundenbeginn ein gemeinsames Fundament zu schaffen, auf dem aufgebaut werden kann. Die Kinder finden hierdurch wieder in das Thema zurück. Hieran schließt sich der Einstieg in die aktuelle Stunde an: Die Schülerinnen und Schüler sehen den Fisch Nepomuk in seinem blau-orangenem Aquarium herumschwimmen. Der

**Relevanz für
die SuS**

**Begründung
der wichtigsten
Entscheidungen des geplanten
Unterrichts**

schüchterne Fisch und sein Talent zur Farbänderung werden den Kindern vorgestellt. Die Kinder erkennen, dass der Fisch aufgrund seiner Farbe im orangefarbenen Teil des Aquariums deutlich besser getarnt ist als im blauen Teil. Dies ermöglicht den Kindern, Ideen zu äußern, wie der Fisch sich abhängig von seiner Position besser tarnen kann. Hieraus entwickeln die Kinder die Leitfrage der Stunde: Wie kann Nepomuk erkennen, wann er seine Farbe ändern soll – und wie kann dies gelingen? Durch die „Geschichte“ um den Fisch Nepomuk wird das Interesse der Schülerinnen und Schüler daran geweckt, dem kleinen Fisch zu helfen; die Leitfrage entwickelt sich auf natürliche Weise aus dem Unterrichtsgespräch.

An den Einstieg schließt sich eine zweistufige Arbeitsphase mit Zwischensicherung an. Der Ablauf dieser Arbeitsphase sowie der zeitliche Umfang der einzelnen Phasen wird den Schülerinnen und Schülern zu Beginn von der Lehrerin transparent gemacht. Hierzu zählt auch eine Wiederholung der Computerregeln: Eine Arbeit mit den Rechnern ist nur nach entsprechend deutlicher Aufforderung der Lehrerin gestattet und endet für alle Kinder mit Beginn der Präsentationsphase. Diese Vorgabe wird durch die Struktur der Stunde unterstützt, da eine Bearbeitung der Aufgabe 3, die die Computer erfordert, erst nach der Zwischensicherung erfolgt und für alle Kinder gleichzeitig beginnt und endet. Auf diese Weise sollen Ablenkungen vermieden werden, die die PCs möglicherweise für die Kinder bedeuten. Zudem ist den Kindern durch die Transparentmachung der einzelnen Phasen eine klare Struktur vorgegeben, an der die sich bei ihrer Arbeit orientieren können. Dies gilt als ein Merkmal eines guten und erfolgreichen Unterrichts (vgl. etwa Me10, S. 25f.).

In der ersten Phase werden die ersten zwei Aufgaben auf den Arbeitsblättern bearbeitet, in denen grundlegende Kenntnisse über die „Fühlen“-Blöcke und die bedingte Anweisung vermittelt werden. Die erste Aufgabe gestattet den Kindern, hinsichtlich der Funktion der blauen „Fühlen“-Blöcke Spekulationen anzustellen und sich hinsichtlich ihrer Vermutungen mit dem Partner oder der Partnerin auszutauschen. Hierdurch wird ein direkter „Wissensinput“ zugunsten einer stärkeren kognitiven Aktivierung der Kinder vermieden. Die zweite Aufgabe verlangt von den Kindern die Anwendung ihres neuerworbenen Wissens auf mehrere Fallbeispiele, wodurch geprüft wird, ob die neuen Inhalte verstanden wurden und ein Übergang für die spätere eigene Arbeit an den Rechnern geschaffen wird, in denen die Kinder in Teams eigene Skripte erstellen müssen.

Die Aufgaben sind also derart konzipiert, dass das Anforderungsniveau behutsam stufenweise ansteigt, um die Kinder nicht zu überfordern. Weiterhin wurde darauf geachtet, die Aufgaben kindgerecht und verständlich zu formulieren, um den Kindern zu ermöglichen, die Aufgaben weitgehend eigenständig zu bearbeiten. Dies entspricht den Anforderungen an die inhaltliche Klarheit der Aufgaben nach H. Meyer (Me10, S. 55). Um den Lernfortschritt der Schülerinnen und Schüler nach den ersten beiden Aufgaben sicherzustellen und etwaige Fragen und Probleme zu klären, werden die Aufgaben 1 und 2 im Rahmen einer Zwischensicherung mit der ganzen Lerngruppe besprochen. Hierdurch wird garantiert, dass kein Kind unvorbereitet in die Rechnerarbeit der Aufgabe 3 geht.

Die Aufgabe 3 wird von den Schülerinnen und Schülern in Partnerarbeit an den Computern erledigt. Auf diese Weise können sich die Kinder auch bei der Bearbeitung der anspruchsvollen dritten Aufgabe miteinander austauschen

und gemeinsame Ideen finden. Wesentlich bei dieser Form der „Paarprogrammierung“ ist selbstverständlich, dass die Kinder angehalten werden, sich in gleichen Teilen an der Programmierung zu beteiligen. Um den Einstieg zu erleichtern, wurde die Aufgabe entsprechend dem progressiven Kompetenzaufbau (vgl. Abschnitt 1) vorstrukturiert: Zunächst müssen die Kinder die entsprechenden „Fühlen“-Blöcke auswählen, diese im nächsten Schritt mit einem Block für eine bedingte Anweisung verknüpfen und erst im letzten Schritt die ihnen bereits bekannte Endlosschleife hinzufügen. Als visualisierte Hilfestellung sind neben den Computern Hilfsplakate der Scratch-Umgebung vorhanden, mithilfe derer sich die Kinder bei grundlegenden Schwierigkeiten orientieren können.

Um etwaigen weiteren Schwierigkeiten entgegenzuwirken, werden für die Aufgabe 3 im Sinne einer inneren Differenzierung Tippkarten bereitgestellt. Diese geben den Kindern in vier Stufen zunehmend konkretere Hinweise auf die Lösung der Teilaufgaben. Eine Differenzierung nach oben erfolgt durch eine zusätzliche Sternchenaufgabe, die über das geschilderte Stundenziel hinausgeht und dieses vertieft. Sie kann von besonders schnellen Gruppen bearbeitet werden. Die Konzepte der Partnerarbeit zur Unterstützung in heterogenen Lerngruppen sowie die Aufteilung der Aufgaben in „Fundamentum“ und „Additum“ folgt Vorschlägen der Binnendifferenzierung im Informatikunterricht nach Hubwieser (vgl. Hu07, S. 23; siehe auch Hu06, S. 82ff.). In der Gestaltung der drei verschiedenen Aufgaben wurde darauf geachtet, unterschiedliche Symbolebenen anzusprechen. Während die ersten beiden Aufgaben überwiegend sprachlich, teilweise ikonisch, konzipiert wurden, steht der praktische, enaktive Zugang im Fokus der dritten und letzten Aufgabe. Die wechselseitige Verknüpfung dieser drei Repräsentationsmodi nach dem sogenannten EIS-Prinzip (vgl. He11, S.185ff.) bietet den Schülerinnen und Schülern einen abwechslungsreichen Zugang und fördert das ganzheitliche Lernen.

An die zweite Erarbeitungsphase schließt sich die Sammlung und Sicherung der Resultate der Aufgabe 3 an. Laut H. Meyer gilt es dabei darauf zu achten, die Ergebnissicherung „klar und verbindlich“ zu gestalten (Me10, S. 55). In der Sammlungsphase werden zunächst Ansätze und Lösungen der Schülerinnen und Schüler von diesen mithilfe eines Beamers eigenständig vorgestellt. Dabei wird betont, dass auch noch unvollständige Lösungen richtige und gute Ideen enthalten und die Ausführbarkeit gegenüber gut durchdachten Ansätzen der Kinder zweitrangig sind. Gegebenenfalls präsentiert zunächst ein Team, das nicht fertig geworden ist, sodass dessen Ideen gewürdigt und von anderen Teams ergänzt werden können. Sollten mehrere unterschiedliche Lösungen produziert worden sein, können diese nebeneinandergestellt und von den Kindern verglichen und beurteilt werden. Hierdurch gewinnen die Kinder einen ersten Eindruck dafür, nach welchen Maßstäben die Qualität von Programmen bewertet werden kann. Zudem wird ein Klima des offenen Umgangs mit Fehlern und konstruktiver Kritik innerhalb eines Klassengesprächs kultiviert.

In Absprache mit den Schülerinnen und Schülern wird im Konsens eine „Musterlösung“ ausgewählt, die anschließend im Rahmen einer schriftlichen Sicherung von allen Kindern in ihre Hefte übertragen wird. Hierzu wird den Kindern ein zusätzliches Sicherungsblatt zur Verfügung gestellt. Die Kinder übertragen die Lösung und heften dieses und die anderen ausgeteilten Blätter anschließend in der für diese Projektwoche angelegten Mappe ab.

3. Artikulationsschema¹

Dauer (Min.)	Unterrichtsphase	Unterrichtsinhalt	Sozial-/Aktionsform	Materialien/Medien/Werkzeuge	didaktisch-methodischer Kommentar
10min	Einstieg	<ul style="list-style-type: none"> - Wiederholung der letzten Stunden durch ein kleines Quiz (Was bedeuten die bereits gelernten Blöcke?) - Vorstellung von Nepomuk und seinem Aquarium. SuS stellen Vermutungen an, wie sich Nepomuk besser verstecken kann -> Die Leitfrage wird erarbeitet: Wie kann Nepomuk erkennen, wann er seine Farbe ändern soll – und wie kann dies gelingen? - Der Ablauf der Stunde wird erklärt und die Computerregeln kurz wiederholt 	Angeleitetes Unterrichtsgespräch	Große Abbildungen der bekannten Blöcke, Computer mit dem Programm Scratch und der vorbereiteten Szene, Beamer	<ul style="list-style-type: none"> - Aktivierung des Vorwissens - Festigung des vorhandenen Wissens - Motivation durch das selbstständige Finden der Leitfrage und die Geschichte mit dem Fisch Nepomuk - Transparenz hinsichtlich des Unterrichtsziels und des Ablaufs der Stunde
15min	Erarbeitung und Zwischensicherung	- Erarbeitung der Fühlen-Blöcke und ihrer Funktion	Einzel- und Partnerarbeit	Arbeitsblätter	- Eigenständiges Erarbeiten des neuen Wissens
10min		Zwischensicherung der Ergebnisse <ul style="list-style-type: none"> - Ergebnissicherung der erarbeiteten Erkenntnisse - Die gesamte Klasse geht geschlossen in den Computerraum 	Unterrichtsgespräch im Plenum	Arbeitsblätter	<ul style="list-style-type: none"> - Anwenden des Wissens - Zusammenarbeit und Austausch durch Partnerarbeit (kooperatives Lernen) - Gemeinsames Sichern und Vergleichen der Ergebnisse
35min		<ul style="list-style-type: none"> - Anwendung der Fühlen-Blöcke in der Szene in Scratch mithilfe des Aufgabenblattes - Die Kinder können bei Schwierigkeiten auf die vorgefertigten Tippkarten zurückgreifen oder die Lehrkraft fragen 	Partnerarbeit	Arbeitsblätter, Computer mit dem Programm Scratch	- Unterstützung im Lernprozess durch die Lehrkraft und Tippkärtchen

¹ Unter der Artikulation wird im didaktischen Kontext die (zeitliche) Abfolge der Unterrichtsphasen verstanden.

Dauer (Min.)	Unterrichtsphase	Unterrichtsinhalt	Sozial-/Aktionsform	Materialien/Medien/Werkzeuge	didaktisch-methodischer Kommentar
10min	Auswertung/ Präsentation/ Reflexion	<ul style="list-style-type: none"> - Vorstellung verschiedener Gruppenergebnisse - Vergleichen verschiedener Lösungen - Einigung auf eine Musterlösung - Fragen werden von den Gruppen beantwortet 	Unterrichtsgespräch im Plenum	Computer und Beamer	<ul style="list-style-type: none"> - Wertschätzung aller Lösungen - Fehler werden als Lernanlass gesehen - Verschiedene Lösungen werden diskutiert - Das Verständnis wird gefestigt - Erklärungen werden von Schülern gegeben
10min	Sicherung	<ul style="list-style-type: none"> - Die Musterlösung wird auf das Sicherungsblatt übertragen - Die Arbeitsblätter werden von allen Kindern eingehftet 	Einzelarbeit	Sicherungsblätter	<ul style="list-style-type: none"> - Sicherung des Stundenziels durch das gemeinsame Dokumentieren - Festigung des Wissens

Literaturverzeichnis

- [GI17] Gesellschaft für Informatik (GI) e. V. (Hrsg.): Kompetenzen für informatische Bildung im Primarbereich, <https://metager.to/gibspdf>, Version: 993, Stand: 4.1.2018.
- [GI08] Gesellschaft für Informatik (GI) e. V. (Hrsg.): Grundsätze und Standards für die Informatik in der Schule – Bildungsstandards Informatik für die Sekundarstufe I, 2008.
- [He11] Heske, Henning: Ganzheitliches Lernen – Praxishandbuch für die Sekundarstufe I und II. In: Mathematik Didaktik, Hrsg. von T. Leuders, 6. Auflage, Cornelsen Verlag, Berlin, 2011, S.185-197.
- [Hu06] Humbert, Ludger: Didaktik der Informatik mit praxiserprobtem Unterrichtsmaterial, 2. überarbeitete und erweiterte Auflage, Teubner Verlag, Wiesbaden, 2006.
- [Hu07] Hubwieser, Peter: Didaktik der Informatik – Grundlagen, Konzepte, Beispiele; 3. überarbeitete und erweiterte Auflage, Springer Verlag, Berlin Heidelberg, 2007.
- [Me10] Meyer, Hilbert: Was ist guter Unterricht?, 7. Auflage, Cornelsen Verlag, Berlin, 2010.
- [MSW08] Ministerium für Schule und Weiterbildung des Landes NRW (Hrsg.): Richtlinien und Lehrpläne für die Grundschule in NRW, Ritterbach Verlag, Frechen, 2008, http://www.schulentwicklung.nrw.de/lehrplaene/upload/lehrplaene_download/grundschule/grs_faecher.pdf, Stand: 07.07.2016.
- [Sc93] Schwill, Andreas: Fundamentale Ideen der Informatik, Zentralblatt für Didaktik der Mathematik 1/1993, S. 1-30.
- [UI10] Ullenboom, Christian: Java ist auch eine Insel, 9. aktualisierte Auflage, Rheinwerk Verlag, Bonn, 2010.
- [Wa13] Walter, Daniel: Über die Eignung der Programmiersprache Scratch zur Aneignung von Programmierungskompetenzen, Beitrag auf der Herbsttagung des Arbeitskreises ‚Mathematikunterricht und Informatik‘ der GDM, 2013.

Anhang

Name: _____

Datum: _____

Aufgabe 1:



Hallo Superprogrammierer!

Ich bin Enno und lerne mit dir heute ein paar neue Blöcke aus Scratch kennen.

Die Blöcke haben die Überschrift **Fühlen**.

Kannst du mir sagen, was die Blöcke machen?

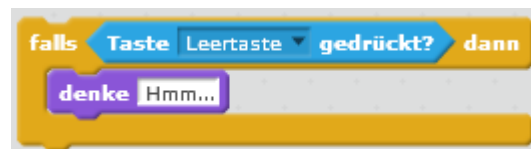


Was denkst du?

	Ja	Nein
Verändern die Blöcke das Aussehen (der Figur)?	<input type="radio"/>	<input type="radio"/>
Fragen die Blöcke nach Informationen (über die Figur)?	<input type="radio"/>	<input type="radio"/>
Bewegen die Blöcke die Figur?	<input type="radio"/>	<input type="radio"/>
Werden mit den Blöcken Töne abgespielt?	<input type="radio"/>	<input type="radio"/>
Können die Blöcke eine Aktion wiederholen lassen?	<input type="radio"/>	<input type="radio"/>

Selbstkontrolle: Klappe erst um, wenn du dich mit deinem Partner ausgetauscht hast!

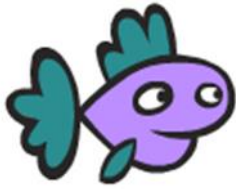
Die hellblauen Fühlen-Blöcke fragen nach Informationen über die Figur. Sie können nicht alleine stehen und brauchen einen orangenen Steuerungsblock:



Das Programm beantwortet die Frage mit „Ja“ oder „Nein“. Falls die Antwort „Ja“ ist, wird der zusätzlich eingefügte Bewegung-, Aussehen- oder Klangblock ausgeführt.

Hast du das verstanden? Tausche dich mit deinem Partner aus.

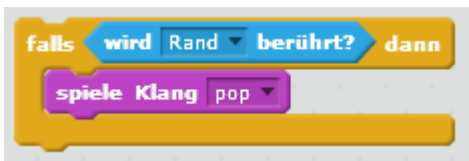
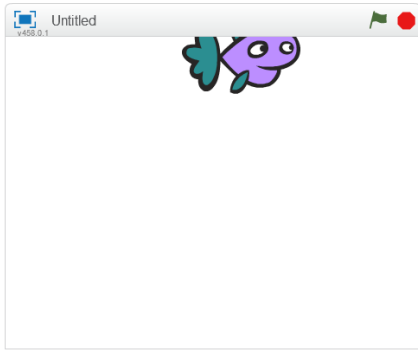
Aufgabe 2:



Die hellblauen Fühlen-Blöcke können nicht alleine stehen, sie müssen in einen anderen Block geschoben werden.

Hier siehst du vier Skripte mit ihren Szenen. Werden ihre Befehle ausgeführt?

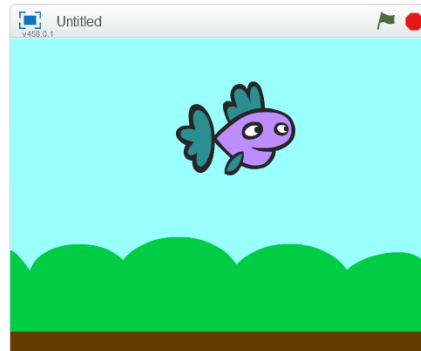
a.



Wird ausgeführt

Wird nicht ausgeführt

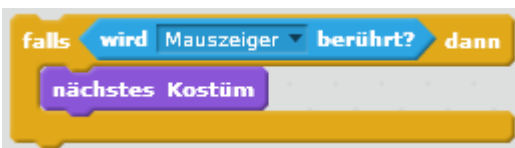
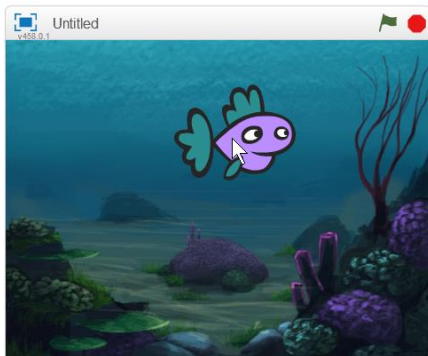
b.



Wird ausgeführt

Wird nicht ausgeführt

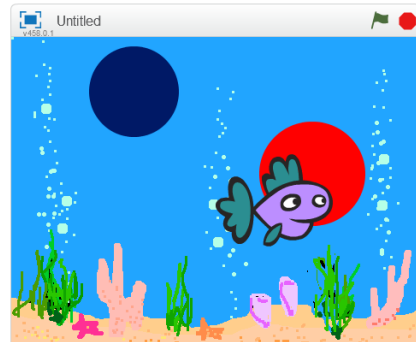
c.



Wird ausgeführt

Wird nicht ausgeführt

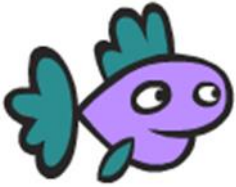
d.



Wird ausgeführt


Wird nicht ausgeführt

Aufgabe 3:

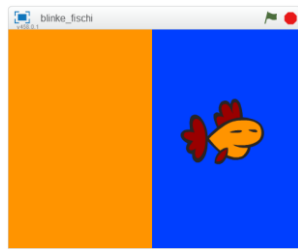
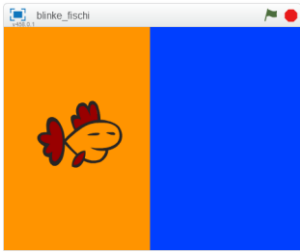


Du darfst dich jetzt mit deinem Partner an einen Computer setzen.

Das Programm „Scratch“ ist bereits geöffnet und du siehst meinen Freund Nepomuk auf dem Bildschirm. Er freut sich schon sehr dich zu sehen.

Wenn du auf die grüne Fahne  klickst, siehst du ihn über den Bildschirm schwimmen.

Nepomuk ist ein bisschen schüchtern, er versteckt sich gerne. Auf der linken Seite des Aquariums ist er durch seine orangene Farbe fast nicht zu sehen, aber auf der rechten Seite siehst du ihn sofort.



Nepomuk möchte auf der blauen Seite des Aquariums seine Farbe zu Blau ändern, damit er auch dort umherschwimmen kann, ohne sofort gesehen zu werden.



Kannst du ihm helfen?

Wenn du nicht weiterkommst, kannst du dir Tippkarten für die Aufgaben a, b und c holen.

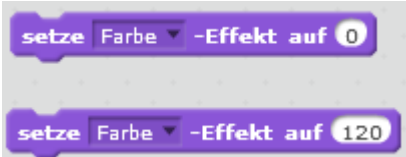


- Schau dir die Skripte bei Aufgabe 2 noch einmal genau an. Wie kann das Programm erfahren, auf welcher Farbe sich Nepomuk befindet?
(Hilfe: Um auf dem Block „Wird Farbe ... berührt?“ die Farbe zu ändern, musst du erst auf das kleine Quadrat und dann auf die gewünschte Farbe im Bild klicken.)
- Was genau soll passieren, wenn sich Nepomuk auf der blauen/orangenen Seite befindet?
(Hilfe: Die Farbe Orange ist bei Nepomuk der Farbeffekt 0, die Farbe Blau der Farbeffekt 120.)
- Das Programm muss nach dem Klick auf die grüne Fahne **immer wieder** prüfen, auf welcher Farbseite sich Nepomuk befindet.

- ★ d. Wenn Nepomuk entdeckt wird und dein **Mauszeiger** ihn **berührt**, erschrickt er und ruft „**Oje, jetzt hast du mich entdeckt!**“
Kannst du dies mit einem weiteren Skript umsetzen?


Tippkarten Aufgabe 3a:


I. Du brauchst zweimal den hellblauen Fühlen-Block „Wird Farbe ... berührt?“.	II. 
III. Denk daran, dass die Fühlen-Blöcke nicht alleine stehen können.	IV. 

Tippkarten Aufgabe 3b:

I. Du brauchst zweimal den violetten Aussehen-Block „setze Farbe-Effekt auf...“.	II. 
III. Füge die beiden Aussehen-Blöcke an die entsprechende Stelle in dein Skript ein. 	IV. 

Tippkarten Aufgabe 3c:

I. Du brauchst einen Block, der die grüne Fahne als Start-Taste festlegt. Außerdem brauchst du einen Block, der eine fortlaufende Wiederholung bewirkt.	II. 
--	---

III. Lege den Steuerungsblock „wiederhole fortlaufend“ um dein Skript und den „Wenn  angeklickt“-Block ganz an den Anfang über dein Skript.

IV.



(Die Tippkarte IV zur Aufgabe 3c stellt die Lösung der gesamten Aufgabe dar.)

Was habe ich heute gemacht?

Hier hast du Platz das Skript, welches wir zum Farbwechsel von Nepomuk erarbeitet haben, abzumalen. Der erste Block ist schon eingezeichnet.



Erinnerungsplakat „Die Scratch-Umgebung“

