http://schuelerlabor.informatik.rwth-aachen.de

Marienkäfer steuern mit *Snap!*

[1]

Verfasser: Sascha Ratner, überarbeitet von Christina Schramm

Kurz-Info:

Informatischer Inhalt: Grafische Programmierung mit *Snap!*

Jahrgangsstufe: 3 bis 5

Vorwissen: Keins

KURZINFORMATION FÜR DIE LEHRKRAFT

Titel: *Marienkäfer steuern mit* Snap!

Schulstufe: *Grundschule, Unterstufe*

optimale Jahrgangsstufe: *Klasse 3 bis 5*

[1]

EINORDNUNG IN GESETZLICHE RAHMENBEDINGUNGEN

Kernlehrplan für die Sekundarstufe I – Klasse 5 und 6 in NRW: *Kompetenzbereiche: Modellieren und Implementieren, Darstellen und Interpretieren, Kommunizieren und Kooperieren; Inhaltsfelder: Algorithmen, Informatiksysteme*

Themenbereich: *grafische Programmierung, Grundbegriffe der Programmierung, Schleifen, Verzweigungen, Kommunikation*

Einbindung in den Unterricht: *Das Modul eignet sich als Einstieg in die Informatik, besonders in die Programmierung. Die Schüler*innen lernen mithilfe der grafischen Programmieroberfläche* Snap! *einfache Programmierkonzepte kennen.* Snap! *ermöglicht ihnen dabei durch den Einsatz von Bausteinen, die wie Puzzleteile miteinander verbunden werden, schnell und unkompliziert erste eigene Programme zu erstellen. Aufgrund des großen Einflusses der Informatik in sämtlichen Bereichen des alltäglichen Lebens eignet sich das Modul nicht nur als Ergänzung zum Fach Medienbildung, sondern kann auch fächerübergreifend eingesetzt werden.*

Vorkenntnisse: *keine*

Dauer: 3,5 bis 4 *Stunden (inkl. Pausen)*

Inhaltsbeschreibung: *Nach einer frontalen Einführung in die Handhabung des* Snap!*-Editors erproben die Schüler*innen dessen Funktionsweise weitestgehend eigenständig im Rahmen mehrerer Stationen. Die ersten beiden Stationen sind obligatorisch und vermitteln den Schüler*innen erste Grundlagen. So erlernen sie, Figuren zu bewegen und die Funktionsweise einfacher Bedingungen (falls-dann-sonst). Die dritte Station bietet eine Wahlmöglichkeit zwischen zwei Szenarien (Schatzsuche und Supermarkteinkauf), die jeweils einen anderen Schwerpunkt haben. Neben neuen Konzepten (Schleifen, Kommunikation) erlernen die Schüler*innen dabei auch, das bereits Gelernte in einem neuen Kontext einzusetzen.*

INHALT

LERNZIELE

Die Schüler*innen..

- erstellen mithilfe der grafische Programmierumgebung *Snap!* erste, einfache Programme.
- nutzen grundlegende Programmierbefehle bzw. -konzepte (Fortlaufend-Schleife, Falls-dann-Anweisung, Variablen), indem sie diese im Rahmen ihrer Projekte zielgerichtet anwenden.
- geben einfache Programme mündlich wieder, indem sie ihre Projektergebnisse präsentieren.

FACHLICHE ANALYSE

Snap! ist ein Editor mit einer dazugehörigen Online-Gemeinschaft, der von der *Lifelong-Kindergarten-Group* am Media-Lab des MIT für junge Lernende entwickelt wurde. Für diese Zielgruppe bietet der *Snap!*-Editor den Vorteil, dass er browserbasiert, kostenlos zugänglich und einfach zu bedienen ist. Abgesehen von einer rudimentären Kompetenz im Umgang mit dem Computer (Tastatur, Maus, Browser) müssen die Nutzerinnen und Nutzer keinerlei Vorwissen mitbringen. Sie müssen nicht über Englischkenntnisse verfügen, weil im *Snap!*-Editor Deutsch als Arbeitssprache ausgewählt werden kann. Auch eine Programmiersprache muss im Vorfeld nicht erlernt werden, da der Text zur Programmierung nicht selbst geschrieben werden muss. Vielmehr werden die Schüler*innen intuitiv an die Grundzüge der Programmierung herangeführt, indem sie bereits fertige Code-Puzzle-Blöcke zu einem Programm zusammenfügen. Diese Form der grafischen Programmierung bietet u. a. den Vorteil, dass es keiner syntaktischen Fehlerüberprüfung bedarf, weil syntaktische Fehler schlicht nicht möglich sind. Wie bei einem gewöhnlichen Puzzle können nämlich nur Blöcke zusammengesteckt werden, die zueinander passen. Zusätzlich sind die Code-Puzzle-Blöcke in mehre Kategorien (Bewegung, Steuerung, Variablen etc.) unterteilt, um eine bessere Orientierung zu ermöglichen. Neben dem Menü mit den Code-Puzzle-Blöcken und der Programmieroberfläche, in der die Blöcke zusammengesetzt werden, verfügt der Editor über eine Art Bühne, auf der das geschriebene Programm ausgeführt wird. Diese Bühne wie auch die Figuren können individuell gestaltet werden. Eigene Welten und Figuren können hochgeladen werden. Im InfoSphere-Modul arbeiten die Schüler*innen mit Welten und Figuren, die durch das infoSphere-Team erstellt wurden.

Während des Moduls stoßen die Schüler*innen auf grundlegende informatische Konzepte. So müssen sie das Prinzip der Schleife anwenden, um ein Programm oder einzelne Teile zu wiederholen. Eine herausragende Bedeutung kommt dabei der Fortlaufend-Schleife zu, die als einer der Grundbausteine der meisten Programme fungiert. Die Schüler*innen werden schnell erfassen, dass Programme, die diesen Grundbaustein nicht enthalten, oft nicht wie geplant funktionieren. Falls-dann-Anweisungen, die den Kindern ebenfalls begegnen, würden ohne Verwendung des Fortlaufend-Blocks nur einmal ausgeführt und somit nicht die gewünschte Funktion erfüllen. Eine Funktion im informatischen Sinne lernen die Schüler*innen in Form so genannter Eventlistener kennen, die Events und die dazugehörigen Aktionen verknüpfen. So sollen

Programme beispielsweise erst starten, wenn die grüne Fahne angeklickt wurde. Zudem beschäftigen die Schüler*innen sich mit dem Thema Variablen. Bei der Wegsuche des Marienkäfers begegnet ihnen u. a. eine Boolschen Variablen, die angibt, ob das Objekt vor dem Käfer ein Zaun ist.

Folgende Internetbeiträge können zur weiteren Einarbeitung in die Thematik genutzt werden:

- *Visuelle Programmiersprache*. In: Wikipedia. Abgerufen von: https://de.wikipedia.org/wiki/Visuelle_Programmiersprache (eingesehen: 14.02.2022).
- *Schleifen (Programmierung)*. In: Wikipedia. Abgerufen von: https://de.wikipedia.org/wiki/Schleife_(Programmierung) (eingesehen: 14.02.2022).
- *Bedingte Anweisung und Verzweigung*. In: Wikipedia. Abgerufen von: https://de.wikipedia.org/wiki/Bedingte_Anweisung_und_Verzweigung (eingesehen: 14.02.2022).

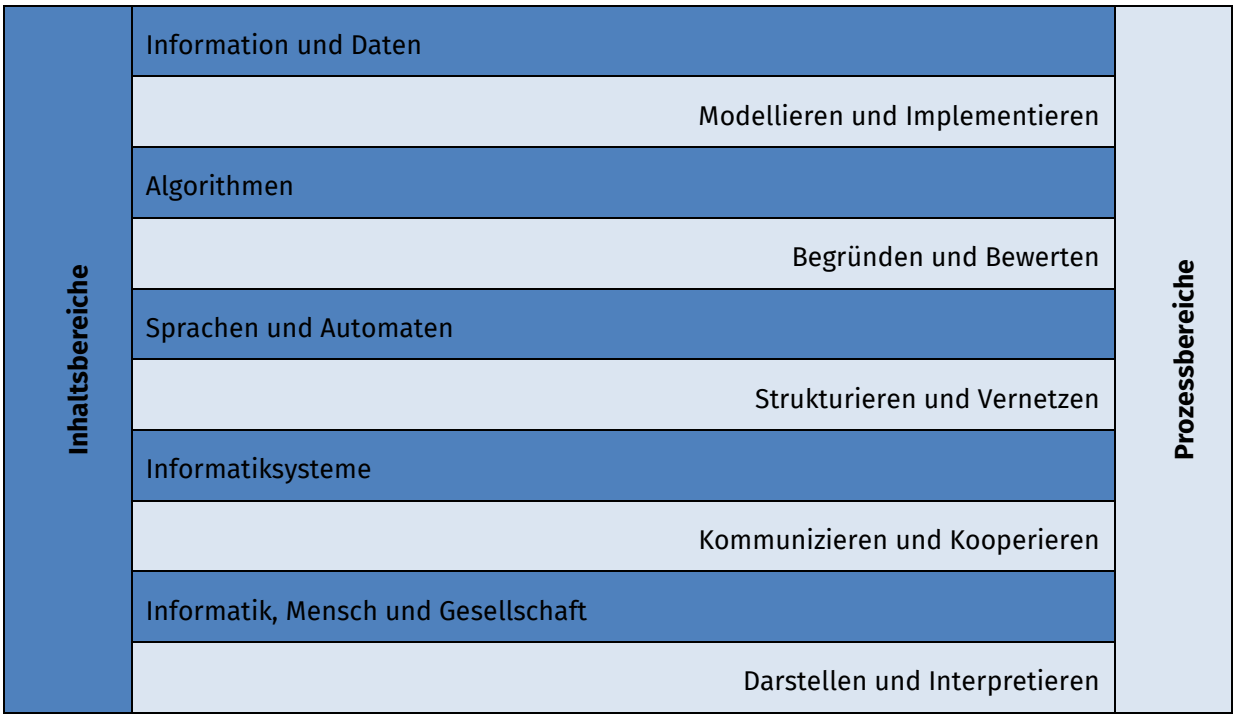
Weiterführende Links zur Programmierumgebung Snap!:

- *Snap!*. Abgerufen von: http://snap.berkeley.edu (eingesehen: 14.02.2022).
- *Snap!-Wiki*. Abgerufen von: https://de.scratch-wiki.info/wiki/Snap! (eingesehen: 14.02.2022).

EINORDNUNG IN GESETZLICHE RAHMENBEDINGUNGEN

BILDUNGSSTANDARDS PRIMARBEREICH DER GESELLSCHAFT FÜR INFORMATIK (GI) E.V.

Derzeit existieren in Deutschland keine gesetzlich bindenden Regelungen für die Einordnung von Informatik in den Grundschullehrplan, jedoch gibt es seit 2019 vom *Arbeitskreis „Bildungsstandards Primarbereich" der Gesellschaft für Informatik (GI) e. V.* erarbeitete Empfehlungen bzgl. der „Kompetenzen für informatische Bildung im Primarbereich". Diese bieten eine geeignete Basis für die Vermittlung einer informatischen Grundbildung im Primarbereich und schaffen damit zugleich die Grundlage für das Informatiklernen an weiterführenden Schulen. Die Schüler*innen sollen befähigt werden, mit den steigenden informatischen Anforderungen des Alltags umzugehen. Der frühe Kontakt mit informatischen Inhalten fördert hierbei den Zugang zur und das Interesse an Informatik und senkt damit die Hemmschwelle gegenüber informatischen Inhalten. Konkret sehen die Bildungsstandards Informatik für den Primarbereich hierfür die Entwicklung von zehn Kompetenzen vor, wobei sich diese in jeweils fünf Inhalts- und Prozessbereiche unterteilen und untereinander verzahnt sind.



In dem vorliegenden Modul werden vorrangig die folgenden Kompetenzen abgedeckt:

Prozessbereiche

1. *Modellieren und Implementieren:* Die Schüler*innen entwickeln ein informatisches Modell zu einer gegebenen Aufgabenstellung und setzen dieses mit geeigneten, altersgerechten Werkzeugen um.
2. *Begründen und Bewerten:* Die Schüler*innen stellen Fragen und erklären informatische Zusammenhänge unterschiedlicher Komplexität mit ihren eigenen Worten. Dazu nutzen sie zunehmend Fachsprache.
3. *Strukturieren und Vernetzen:* Die Schüler*innen strukturieren die vorgegebenen Sachverhalte und zerlegen diese in einzelne Bestandteile. Sie verknüpfen die informatischen Inhalte zur Lösung der Aufgabenstellung. Dabei vollziehen sie die Zusammenhänge nach.
4. *Kommunizieren und Kooperieren:* Die Schüler*innen bearbeiten die Aufgabenstellungen gemeinsam und tauschen sich über ihre eigenen Denkprozesse aus. Hierzu verwenden sie sowohl Umgangs- als auch zunehmend Fachsprache.
5. *Darstellen und Interpretieren:* Die Schüler*innen stellen ihre Ergebnisse vor und gehen dabei auf eigene Denkprozesse und Vorgehensweisen ein. Außerdem sind sie in der Lage, die Ergebnisse und Vorgehensweisen der anderen Schüler*innen zu interpretieren.

Inhaltsbereiche

1. *Algorithmen:* Die Schüler*innen entwerfen und realisieren einfache Algorithmen mittels grafischer Grundbausteine zur Lösung vorgegebener Aufgabenstellungen. Sie verwenden dabei aktiv grundlegende Befehle und sind in der Lage, andere Algorithmen zu interpretieren.
2. *Informatiksysteme:* Die Schüler*innen nutzen Informatiksysteme (z. B. Computer) selbstständig und zielgerichtet.

Dieses Modul für junge und im Bereich der Informatik noch unerfahrene Schüler*innen verfolgt das übergeordnete Ziel der „informatischen Grundbildung als wichtigen Bestandteil der Allgemeinbildung“ (KLP, S. 8). Ausgangspunkt des Informatikunterrichts der Klassen 5 bis 6 sollen laut KLP altersgerechte Fragestellungen mit lebensweltlichem Bezug sein, mit denen die Kinder sich aktiv und selbstständig auseinandersetzen (vgl. KLP, S. 8). Frage- bzw. Problemstellungen, die diesen Anforderungen entsprechen, begegnen den Kindern im Rahmen der unterschiedlichen Projekte, die das Kernstück des Moduls bilden. Hier programmieren die Schüler*innen eine Spielfigur in Gestalt eines Käfers so, dass diese zu einem Blumentopf läuft, begleiten einen Piratenpapagei auf Schatzsuche oder Dotty und Gran bei ihren Einkäufen. Dass am Ende der Projekte stets ein funktionierendes, beobachtbares Programm steht, wirkt schüleraktivierend und motivierend. Eine Überforderung der noch jungen und programmierunerfahrenen Schüler*innen wird vermieden, indem vor der weitestgehend selbstständigen Bearbeitung der Projekte eine frontale, vorentlastende Einführung in die Arbeit mit dem *Snap*!-Editor erfolgt. Zudem liegt während der Arbeit an den Projekten ein Hilfsblatt (Blatt 0) bereit, das die relevanten Komponenten des Editors zusammenfasst und jederzeit konsultiert werden kann. Auch der *Snap!*-Editor selbst ist so konzipiert, dass er eine altersgemäße Auseinandersetzung gewährleistet, denn er stellt eine blockbasierte Programmiersprache zur Verfügung, sodass die Schüler*innen sich nicht mit einer komplexen textuellen Form der Programmierung auseinandersetzen müssen. Die Möglichkeit der Sprachwahl ist ebenso ein Aspekt, der den Einstieg erleichtert.

Im Folgenden werden diejenigen Kompetenzbereiche und Inhaltsfelder des KLP zusammengefasst, die in dem Modul besonders berücksichtigt sind:

Kompetenzbereiche

Modellieren und Implementieren: Nach der Einführung in die Handhabung des Editors arbeiten die Schüler*innen in Zweierteams an Projekten, die in ihrer Komplexität ansteigen, und setzen die Problemstellungen in kleinen Programmen weitestgehend eigenständig um. Indem sie sich hierbei ständig über die nächsten Arbeitsschritte austauschen, lernen sie „Sachverhalte und Abläufe unter informatischem Blickwinkel zu beschreiben“ (KLP, S. 12). Eine didaktische Reduktion der Problemstellung findet durch die detaillierten Anleitungen zur Erstellung des Programmmodells und dessen Übertragung in die blockbasierte Programmiersprache statt (Implementierung). „Durch den Implementierungsprozess wird das Ergebnis [der] Modellbildung erlebbar und überprüfbar“ (KLP, S. 12).

Darstellen und Interpretieren: Am Ende des Moduls präsentieren einzelne Teams ihre Ergebnisse. In diesen Kurzpräsentationen üben die Schüler*innen, Ergebnisse geeignet darzustellen. Ihr Publikum gibt ihnen Rückmeldung, sodass sie gleichzeitig auch erlernen, die Ergebnisse anderer korrekt zu interpretieren und ein konstruktives Feedback zu geben.

Kommunizieren und Kooperieren: Die Schüler*innen arbeiten im Rahmen der Projekte ständig in Zweierteams zusammen, müssen in den Erarbeitungsprozessen also stets miteinander kommunizieren und kooperieren, um ihr gemeinsames (Projekt-)Ziel zu erreichen.

Inhaltsfelder

Algorithmen: Die Schüler*innen erfahren, dass sich hinter Phänomenen, die sie aus Computerspielen kennen (z. B. eine Figur zu einem Gegenstand laufen lassen) Programme verbergen und diese eine logische Abfolge von Anweisungen sind, die von einem Computer befolgt wird. Indem sie die unterschiedlichen Projekte programmieren, lernen sie algorithmische Grundstrukturen (Schleifen, einfache Bedingungen) kennen und diese zielführend anzuwenden. Die Implementierung der entwickelten Algorithmen erfolgt mittels des *Snap!*-Editors in einer altersgerechten Programmierumgebung.

Informatiksysteme: Neben dem Computer benutzen die Schüler*innen den *Snap!*-Editor (Software) zielgerichtet und verstehen die Funktionsweise der Blockbausteine, die sie für ihre Programmierung nutzen.

BENÖTIGTE UNTERRICHTLICHE VORAUSSETZUNGEN

Die Schüler*innen sollten über eine dem Alter angemessene Lesekompetenz verfügen. Rudimentäre Fähigkeiten im Umgang mit dem Computer sind ebenfalls hilfreich.

EINBETTUNG IN DEN SCHULUNTERRICHT

Das Modul bietet sich an, um jungen Schülerinnen und Schülern einen ersten Kontakt mit der Informatik zu ermöglichen. Aufgrund der Verwendung des *Snap!*-Editors eignet es sich besonders für einen Einstieg in die (grafische) Programmierung.

DIDAKTISCHE/METHODISCHE SCHWERPUNKTSETZUNG

BESCHREIBUNG DES MODULABLAUFS

Zunächst werden die Schüler*innen im InfoSphere – Schülerlabor Informatik der RWTH Aachen von den Betreuerinnen und Betreuern, die sich und das Schülerlabor kurz vorstellen, begrüßt. Nach einer Transparentmachung des Modulablaufs erhalten die Kinder im Plenum eine Einführung in die Programmieroberfläche *Snap!,* bevor sie in Partnerarbeit die Projekte, die im Folgenden kurz vorgestellt werden, bearbeiten und Snap! dabei aktiv nutzen. Die eingeführten und für die Bearbeitung der Projekte notwendigen Komponenten der Programmieroberfläche sind auf einem Hilfsblatt (Blatt 0) zusammengefasst. Dieses Hilfsblatt können die Schüler*innen während ihrer Projektarbeit jederzeit konsultieren.

Projekt 1: Lauf Käfer, lauf! (Einführung)

Im Rahmen des Einführungsprojekts sollen die Schüler*innen in die Arbeit mit der grafischen Programmierumgebung einsteigen. Sie programmieren einen Käfer so, dass dieser selbstständig zum Blumentopf läuft. Während die Kinder im ersten Aufgabenteil noch frei zwischen den ihnen zur Verfügung stehenden Befehle wählen und diese in beliebiger Anzahl nutzen können, ist die Anzahl beim zweiten Aufgabenteil begrenzt. Dies erhöht den Schwierigkeitsgrad und soll den Kindern effiziente Programmierung näherbringen. Als Hilfestellung sind auf dem Arbeitsblatt kurze Erklärungen zu den notwendigen Programmierblöcken, die auch in den Folgeprojekten relevant sind, aufgelis-

tet. Die Schüler*innen lernen das Zusammenspiel der einzelnen Befehle kennen, indem sie ihr erstes eigenes kleines Programm entwerfen.

Projekt 2: Achtung, Hindernis!

Die zweite Station bringt den Schüler*innen die Programmierkonzepte der Verzweigungen und Bedingungen näher. Der Blumentopf ist jetzt hinter einem Fragezeichen versteckt, während sich unter den anderen Fragezeichen Zäune befinden. Es reicht also nun nicht mehr aus, den Käfer zum Blumentopf zu steuern, sondern die Schüler*innen müssen mithilfe von Verzweigungen und Bedingungen prüfen, was sich unter dem jeweiligen Fragezeichen verbirgt, sodass der Käfer zum Schluss selbstständig den Blumentopf findet. Dieser wird immer zufällig platziert, sodass es auch nicht ausreicht, sich den Weg zu merken.

Projekt 3

In der dritten und letzten Station können die Schüler*innen zwischen zwei verschiedenen Szenarien wählen, die jeweils einen anderen Schwerpunkt haben. Leistungsstarke Schüler*innen können auch beide Szenarien bearbeiten (Binnendifferenzierung).

Projekt 3a: Schatzsuche

In diesem Projekt liegt der Fokus auf Schleifen (while-Schleifen). Die Schüler*innen sollen den Papagei so programmieren, dass er selbstständig zum Schatz läuft und diesen öffnet. Auf seinem Weg kann der Papagei Objekte (wie den Schlüssel zur Schatztruhe) aufheben. Da das Raster der ersten beiden Stationen wegfällt und durch eine Schatzkarte ersetzt wird, können die Schüler*innen nicht mehr unmittelbar sehen, wie viele Schritte sie zu den jeweiligen Objekten benötigen und müssen somit einen anderen Lösungsansatz finden. Daher wird die while-Schleife eingeführt. Diese können sie nutzen und sich an den einzelnen Objekten orientieren, indem sie z. B. so lange geradeaus gehen, bis sie den Berg berühren etc.. Neben den while-Schleifen lernen die Schüler*innen, die bereits zuvor erlernten Befehle in einem neuen Kontext einzusetzen.

Projekt 3b: Supermarkt

In diesem Szenario helfen die Schüler*innen der Spielfigur Dotty, den Einkauf zu erledigen. Da Dotty den Einkaufszettel vergessen hat, muss Gran helfen und die Artikel der Einkaufsliste durchgeben; die Figuren müssen kommunizieren. Daher lernen die Schüler*innen, wie man Nachrichten verschickt und empfängt. So lernen sie einfache Interaktion zwischen verschiedenen Objekten kennen und wenden diese praktisch an.

Nach der Bearbeitung der Projekte stellen einzelne Schülerteams ihre Projektergebnisse vor; die anderen Schüler*innen geben ihnen Rückmeldung.

BEGRÜNDUNG DES METHODEN- UND MEDIENEINSATZES

Ein Grundanliegen des InfoSphere – Schülerlabor Informatik besteht in der Förderung des selbständigen Arbeitens der Schüler*innen zur Lösung informatischer Probleme. Daher erfolgt die Hauptarbeit in Form von kleinen Projekten. Zu den einzelnen Projekten werden Arbeitsblätter zur Verfügung gestellt. Diese beinhalten neben den Aufgabenstellungen auch Erklärungen zu den einzelnen Konzepten sowie Tipps für die Lösung der Aufgaben. Teilweise sind diese Tipps verborgen

hinter Klappboxen, die die Schüler*innen aufklappen können, wenn sie merken, dass sie Unterstützung benötigen. Außerdem steht den Schüler*innen in allen Phasen der Bearbeitung ein Hilfsblatt (Blatt 0) zur Verfügung, das einen Überblick über alle relevanten Elemente der Programmierumgebung gibt. Auf dieser Basis sollen die Schüler*innen die Aufgaben möglichst eigenständig und ohne direkte Anleitung durch die Betreuer*innen bewältigen und sich dabei gegenseitig unterstützen. Die Betreuer*innen nehmen dabei eine beratende und unterstützende Rolle ein und können bei etwaigen Problemen um Hilfe gebeten werden. Bei der Programmieroberfläche Snap! handelt es sich um eine Weboberfläche, die eine Pseudosprache mit einfachen und kindgerechten Begriffen nutzt.

DIDAKTISCHE PRINZIPIEN NACH BAUMANN UND HUBWIESER

Das Konzept des Schülerlabors sieht für alle Module ein selbständiges und aktives Lernen der Schüler*innen mithilfe von Arbeitsmaterialien und kurzen Lehrvorträgen vor. Somit liegt ein besonderer Fokus stets auf dem *Prinzip des aktiven Lernens*. In diesem Modul wird dies maßgeblich mittels des Projektlernens erreicht.

Zu Beginn des Moduls wird den Schüler*innen der Modulablauf und die Zielsetzung erklärt. Zusätzlich wird zu Beginn jedes Arbeitsblattes das Ziel des jeweiligen Projekts noch einmal erläutert. Somit wird das *Prinzip der Zielvorstellung* erfüllt. Die Struktur der Arbeitsblätter ist stets identisch und wird den Schüler*innen zu Beginn transparent gemacht; Lernziele, Aufgabenstellungen und theoretische Hintergründe sind farblich markiert und ermöglichen so eine einfache Orientierung (*Strukturierung*).

Die grafische Programmierumgebung Snap!, in der die Schüler*innen ihre Programme wie Puzzle zusammenfügen, bietet einen altersgerechten Zugang zum Einstieg in die Programmierung (*Prinzip der Stufenmäßigkeit*). Auch wenn die Schüler*innen ihre kleinen Programme mittels eines Blockeditors zusammensetzen, beschäftigen sie sich mit grundlegenden informatischen Konzepten. Wer Konzepte, wie beispielsweise die while-Schleife, mithilfe der kindgerechten Blockprogrammierung verinnerlicht hat, kann diese später auch auf komplexere Sachverhalte übertragen und so sicherlich einfach Zugang zu einer textuellen Programmiersprache finden. Dies entspricht dem *Spiralprinzip*. Durch den Einsatz der grafischen Programmieroberfläche Snap! in Verbindung mit lebensnahen, spielerischen Szenarien (z. B. Schatzsuche, Supermarkteinkauf) ist außerdem das *Prinzip der Lebensnähe und Aktualität* gegeben.

Grundlegende informatische Konzepte werden nicht abstrakt, sondern stets kontextualisiert eingeführt und angewandt (*Operatives Prinzip*). Auch werden die erlernten Konzepte nicht isoliert in einzelnen Aufgabenstellungen behandelt, sondern in zunehmend komplexer werdenden Projekten wiederholt und damit auf neue Kontexte übertragen (*Prinzip der Stabilisierung*).

Die einzelnen Projekte können die Schüler*innen in ihrem eigenen Tempo bearbeiten (*Prinzip des individuellen Lerntempos*). Bei Schwierigkeiten im Rahmen der Arbeit mit der Programmieroberfläche können die Schüler*innen jederzeit selbstständig ein Hilfsblatt konsultieren. Außerdem gibt es auf den Arbeitsblättern zu den Projekten mit einem höheren Schwierigkeitsgrad Klappboxen, die Tipps enthalten. Diese können die Schüler*innen bei Bedarf aufklappen (*Differenzierung*). Außerdem stehen die Betreuer*innen den Schüler*innen jederzeit zur Seite, wenn sie Hilfe benötigen.

Dass am Ende jedes Projekts ein funktionierendes und beobachtbares Programm steht, wirkt sich positiv auf die Motivation aus (*Motivierung*).

VERLAUFSPLAN DES MODULS

Zeit	Phase	Inhalt	Medium	Sozialform
0:00-0:05	Begrüßung	*Vorstellung InfoSphere und Betreuerteam*	Surface Hub mit Modul-PowerPoint	LV
0:05-0:20	Einstieg	*Vorstellung des Moduls, der Programmieroberfläche (Snap!) und Arbeitsmaterialien*	Surface Hub, Internet, *Snap!*-Editor, ABs	LV/UG
0:20-0:50	Erarbeitung I	*Projekt 1: Erstellung eines ersten Programms*	Laptop, Internet, *Snap!*-Editor, Arbeitsblätter (AB0, AB1)	PA
0:50-1:05	Pause			
1:05-1:50	Erarbeitung II	*Projekt 2: Kennenlernen von Verzweigungen und Anwendung in einem weiteren Programm*	Laptop, Internet, *Snap!*-Editor, Arbeitsblätter (AB0, AB2)	PA
1:50-2:00	Pause			
2:00-3:00	Erarbeitung III	*Station 3: Auswahl zwischen zwei Szenarien mit unterschiedlichen Schwerpunkten (Schleifen bzw. Kommunikation)*	Laptop, Internet, *Snap!*-Editor, Arbeitsblätter (AB0, AB3a und 3b)	PA
3:00-3:20	Sicherung	*Vorstellung der Ergebnisse*	Surface Hub, Laptop, *Snap!*-Editor	SV
	Eventualphase	*Evaluation (ca. 20 Minuten einplanen)*	Fragebögen für Grundschulevaluation	
3:20-3:30	Verabschiedung			

Legende: Lehrervortrag (LV), Unterrichtsgespräch (UG), Partnerarbeit (PA), Schülervortrag (SV)

QUELLENVERZEICHNIS

Gesellschaft für Informatik e.V. – „Arbeitskreis „Bildungsstandards Informatik im Primarbereich“ (2019): *Kompetenzen für informatische Bildung im Primarbereich.* In: informatikstandards.de. Abgerufen von: https://informatikstandards.de/fileadmin/GI/Projekte/Informatikstandards/Dokumente/v142_empfehlungen_kompetenzen-primarbereich_2019-01-31.pdf (eingesehen: 14.02.2022).

Ministerium für Schule und Bildung des Landes Nordrhein-Westfalen (2021): *Kernlehrplan für die Sekundarstufe I – Klassen 5 und 6 in Nordrhein-Westfalen.* Abgerufen von: https://www.schulentwicklung.nrw.de/lehrplaene/lehrplan/256/si_kl5u6_if_klp_2021_07_01.pdf (07.03.2022).

Programmierumgebung Snap!. Abrufbar von: http://snap.berkeley.edu/.

ABBILDUNGSVERZEICHNIS

ANHANG

- **Arbeitsblätter**
  - AB0_Programmierumgebung
  - AB1_Einfuehrung.pdf
  - AB2_Hindernisse.pdf
  - AB3a_Schatzsuche.pdf
  - AB3b_Supermarkt.pdf
  - AB3a_Schatzsuche-Abdeckung.pdf
  - AB3b_Supermarkt-ABdeckung.pdf
- **Musterlösungen** zu den Supermarkt- und Schatzsuche-Programmen, die **Welten** zu den einzelnen Projekten sowie die **editierbaren Versionen** der Arbeitsblätter stellen wir Lehrkräften gerne auf Anfrage zur Verfügung (schuelerlabor@informatik.rwth-aachen.de).