

Methodisch-didaktische Handreichung Low Coding



Informatik

Sekundarstufe I

1. Fachliche Ausgangslage und Relevanz	3
2. Verortung in den Rahmenlehrplänen des Faches Informatik (Sek I)	3
3. Berufsfeldkunde im Kontext „Low Coding“.	4
4. Unterrichtsverlauf: Vom Befehl zur Bedingung – Low Coding praxisnah erleben	7
5. Arbeitsmaterialien: Vom Befehl zur Bedingung – Low Coding praxisnah erleben	14
6. Musterlösung Minispiel Low Coding	21

1. Fachliche Ausgangslage und Relevanz

Ziel dieser Unterrichtsstunde ist es, Schüler:innen einen niedrigschwelligen und motivierenden Einstieg in informatische Grundkompetenzen zu ermöglichen. Im Zentrum steht das Prinzip des Low Codings, mit dem grundlegende Methoden erarbeitet werden können – etwa das Erkennen und Anwenden von Kontrollstrukturen (z. B. Schleifen, Bedingungen), das Gestalten einfacher Algorithmen sowie das logische Modellieren von Abläufen. Damit eignet sich die Einheit besonders gut als Einstieg in Programmierung und digitale Bildung – auch für Lehrende und Lernende ohne Vorerfahrungen in klassischen Programmiersprachen.

Digitale Technologien prägen zunehmend alle Lebensbereiche – von der Kommunikation über die Arbeitswelt bis hin zu Bildung, Mobilität und gesellschaftlicher Teilhabe. Der souveräne Umgang mit digitalen Werkzeugen und ein grundlegendes Verständnis ihrer Funktionsweise werden daher immer wichtiger – nicht nur für künftige IT-Fachkräfte, sondern für alle Mitglieder einer digitalisierten Gesellschaft.

Low Coding bedeutet die vereinfachte Erstellung von Softwarelösungen mithilfe visueller oder blockbasierter Entwicklungsumgebungen. Anstatt komplexe Programmiersprachen zu erlernen, können Nutzer:innen mit vorgefertigten Bausteinen, Symbolen oder Menüs logische Abläufe modellieren und digitale Prozesse automatisieren. So werden zentrale Elemente der Informatik wie Kontrollstrukturen, Algorithmen und logisches Denken mit einem spielerischen Zugang verbunden.

Besonders im schulischen Kontext eröffnet Low Coding neue Möglichkeiten: Kinder und Jugendliche können digitale Prozesse gestalten, ohne zuvor Syntaxkenntnisse erwerben zu müssen. Dabei entstehen nicht nur technische Kompetenzen, sondern auch kreative Problemlösungsstrategien, systemisches Denken und ein reflektierter Umgang mit Technologie. In visuellen Programmierungsumgebungen wie Scratch, MakeCode oder Open Roberta lassen sich z. B. kleine Spiele, Animationen oder Robotersteuerungen realisieren – mit direktem Bezug zu Lebenswelt, Fachinhalten oder gesellschaftlichen Fragestellungen. Ein Praxisbeispiel: Schüler:innen können mit Low-Coding-Tools eine digitale Hausaufgabenübersicht erstellen, ein automatisches Vokabelquiz gestalten oder eine Erinnerungs-App für Geburtstage entwickeln. Solche Projekte verdeutlichen, wie Low Coding unmittelbar an Alltagsbedürfnisse anknüpfen kann und Informatik als praktisches, kreatives Werkzeug erfahrbar wird.

Auch im Hinblick auf berufliche Orientierung und das spätere Arbeitsleben ist das Thema von hoher Relevanz: In vielen Branchen – von Wirtschaft über Verwaltung bis zur Kreativwirtschaft – wächst der Bedarf an Menschen, die digitale Abläufe verstehen, gestalten und kritisch hinterfragen können. Low-Coding-Kenntnisse ermöglichen es, sich aktiv an der digitalen Transformation zu beteiligen. Ein Praxisbeispiel: Mit einfachen Low-Coding-Tools lassen sich digitale Formulare erstellen, Urlaubsanträge automatisieren oder Arbeitsabläufe im Büro vereinfachen – Tätigkeiten, die unmittelbar zum Alltag vieler Organisationen gehören. Damit wird deutlich, wie sich schulisch erworbene Kompetenzen später in berufliche Handlungskompetenzen übertragen lassen.

Die Beschäftigung mit Low Coding verbindet somit technologische Bildung mit kreativem Gestalten, fördert digitale Mündigkeit und zeigt, dass Informatik nicht kompliziert sein muss. Sie lädt Schüler:innen ein, Technik als gestaltbares Werkzeug zu begreifen – und selbst aktiv zu werden.

2. Verortung in den Rahmenlehrplänen des Faches Informatik (Sek I)

In Baden-Württemberg werden im „Aufbaukurs Informatik“ der Klassenstufe 7 explizit Algorithmen als Handlungsanweisungen sowie grundlegende Anweisungsfolgen und Prinzipien der algorithmischen Problemlösung behandelt, teils mit Bezug auf blockbasierte Programmierungsumgebungen, wie sie dem Prinzip

von Low Coding entsprechen. Auch prozessbezogene Kompetenzen wie das Strukturieren und Vernetzen sowie das Entwickeln und Testen einfacher Algorithmen stehen im Fokus. Typische Inhalte aus dem Bildungsplan und Beispielcurriculum umfassen das Formulieren und Anwenden von Schritt-für-Schritt-Anweisungen, das Arbeiten mit Bedingungen und Schleifen bei der Entwicklung elementarer Algorithmen in graphischen/befehlsbasierten Umgebungen und die Verwendung typischer Kontrollstrukturen (Abfolgen, Wiederholungen, Bedingungen) beim Lösen einfacher Aufgabenstellungen mit altersgerechten Systemen. Auch im saarländischen Kernlehrplan ist für die Klasse 7 die Einführung grundlegender Algorithmen und insbesondere die Arbeit mit blockbasierten Umgebungen verpflichtend vorgesehen. Im offiziellen Fachlehrplan Informationstechnologie der Realschule (Jahrgangsstufe 7) in Bayern werden explizit die Erstellung von Programmen mit algorithmischen Grundbausteinen (Anweisung, Sequenz, Auswahlstrukturen, Wiederholungsstrukturen) genannt. Dort heißt es u. a.: „Die Schülerinnen und Schüler [...] erstellen einfache Programme unter Verwendung von algorithmischen Grundbausteinen, Operatoren und Variablen. „Außerdem wird die Verwendung von Wiederholungsstrukturen mit fester Anzahl und Anfangsbedingung benannt. Im gymnasialen Bereich wird der Algorithmikaspekt innerhalb des Faches Informatik in der Jahrgangsstufe 7 ebenfalls behandelt, vorrangig als Teil der Schwerpunkte für „Natur und Technik“. Zum aktuellen Zeitpunkt finden sich in Berliner Lehrplänen für Sekundarstufe I (z. B. Rahmen-Lehrplan 1-10 für das Fach Informatik) die Themen: „formale Beschreibung und Umsetzung von Algorithmen mit geeigneten Werkzeugen“ und „grundlegende algorithmische Strukturen wie Sequenzen, Verzweigungen (Bedingungen) und Wiederholungen (Schleifen)“.

Ähnlich verhält es sich auch mit den Kernlehrplänen in Nordrhein-Westfalen. Dort werden im Fach Informatik für die Sekundarstufe I (z. B. Gesamtschule/Gymnasium, Klasse 5–7) für frühe Jahrgänge festgelegt: „Programmierung mit Grundelementen der Algorithmik: Anweisungen, Sequenzen, Bedingungen, Wiederholungen“. Speziell werden schulinterne Curricula aufgefördert, „visuelle/graphische Entwicklungsumgebungen“ zur Vermittlung dieser Inhalte einzusetzen.

3. Berufsfeldkunde im Kontext von „Low Coding“

Fachinformatiker:in für Anwendungsentwicklung

Allgemeines Berufsbild:

Fachinformatiker:innen für Anwendungsentwicklung konzipieren und programmieren Softwarelösungen, die auf spezifische Anforderungen von Unternehmen, Institutionen oder Nutzergruppen zugeschnitten sind. Dabei arbeiten sie mit verschiedenen Programmiersprachen und Entwicklungsumgebungen, testen Anwendungen, beheben Fehler und passen bestehende Systeme an neue Anforderungen an. Sie sind sowohl in der Planung als auch in der Umsetzung und Wartung von Softwareprojekten tätig – etwa im Handel, der Industrie oder im öffentlichen Dienst. Zum Beruf führt in der Regel eine dreijährige duale Ausbildung zur:zum Fachinformatiker:in für Anwendungsentwicklung. Darüber hinaus gibt es vielfältige, auch unkonventionelle Wege in den Beruf – etwa über ein Studium, eine Umschulung oder den Quereinstieg. Nach der Ausbildung ist zudem jederzeit eine Weiterbildung möglich.

Rolle im Kontext Low Coding:

Im Umfeld von Low Coding entwickeln Fachinformatiker:innen beispielsweise Vorlagen, Module oder Schnittstellen für Low-Code-Plattformen, die dann von anderen ohne tiefere Programmierkenntnisse weiterverwendet werden können. Auch in hybriden Projekten spielen sie eine wichtige Rolle, wenn Low-Code-Elemente mit klassischer Programmierung kombiniert werden – zum Beispiel wenn ein einfaches Formularsystem durch zusätzliche Funktionen erweitert werden soll. Gleichzeitig setzen sie Low-Coding-Tools auch selbst ein, etwa um schnell einen Prototypen, eine kleine interne Anwendung wie ein Ticket-System oder eine App zur Urlaubsverwaltung zu erstellen. Ihre Expertise ist dabei entscheidend, sobald

Anforderungen über die grafische Oberfläche hinausgehen und etwa Daten aus verschiedenen Systemen eingebunden oder Lösungen langfristig stabil betrieben werden müssen.

Kaufleute für Digitalisierungsmanagement

Allgemeines Berufsbild:

Kaufleute für Digitalisierungsmanagement verbinden betriebswirtschaftliche Prozesse mit digitaler Technologie. Sie analysieren Unternehmensabläufe, identifizieren Potenziale für Digitalisierung und schlagen IT-Lösungen vor, die Prozesse effizienter machen. Dabei sind sie häufig als Schnittstelle zwischen Fachabteilungen und IT-Abteilungen tätig. Sie planen, organisieren und begleiten die Einführung neuer Softwaresysteme – sowohl technisch als auch organisatorisch. Ein wichtiger Aspekt ihrer Arbeit ist außerdem die Analyse und Bewertung von Daten, um Optimierungspotenziale sichtbar zu machen und fundierte Entscheidungen zu treffen. Der klassische Zugang zum Beruf erfolgt über eine dreijährige duale Ausbildung zur/zum Kaufmann/Kauffrau für Digitalisierungsmanagement. Auch hier gilt: Der Quereinstieg ist möglich, ebenso wie verschiedene Weiterbildungsmöglichkeiten nach der Ausbildung.

Rolle im Kontext Low Coding:

Low-Code-Plattformen bieten Kaufleuten für Digitalisierungsmanagement die Möglichkeit, einfache Anwendungen oder Automatisierungen selbst zu erstellen – ohne auf die IT-Abteilung angewiesen zu sein. So können sie beispielsweise ein digitales Formular für Urlaubsanträge, einen kleinen Workflow zur Freigabe von Bestellungen oder eine Datenbank zur Verwaltung von Kundendaten anlegen. Ebenso können sie mit Low-Coding-Tools Daten erfassen, strukturieren und auswerten, um Trends oder Schwachstellen in Prozessen zu erkennen – etwa wenn im Personalwesen wiederholt bestimmte Abläufe stocken oder im Kundensupport ähnliche Anfragen gehäuft auftreten. Auf diese Weise nutzen sie Low Coding nicht nur zur schnellen Digitalisierung, sondern auch zur datenbasierten Prozessoptimierung in verschiedenen Unternehmensbereichen.

IT-System-Manager:in / Digital Product Manager:in

Allgemeines Berufsbild:

IT-System-Manager:innen oder Digital Product Manager:innen steuern die Entwicklung, Einführung und Weiterentwicklung digitaler Anwendungen, Produkte oder Plattformen. Dabei behalten sie sowohl technische Aspekte als auch Nutzerbedürfnisse, Geschäftsziele und rechtliche Rahmenbedingungen im Blick. Sie koordinieren Teams, erstellen Anforderungsprofile, analysieren Nutzungsszenarien und sorgen für die Integration von Lösungen in bestehende Systemlandschaften. Der Zugang zu diesem Beruf erfolgt in der Regel über ein Studium oder duales Studium, z. B. in Informatik, Wirtschaftsinformatik oder angewandter Mathematik. Auch ein Weg über eine Ausbildung zur/zum Fachinformatiker:in für Systemintegration mit anschließender Berufserfahrung oder Weiterbildung ist möglich. Wie in vielen IT-Berufen gibt es zahlreiche individuelle und alternative Wege – der Quereinstieg ist verbreitet, und Weiterbildungen sind jederzeit möglich.

Rolle im Kontext Low Coding:

Diese Fachkräfte nutzen Low-Code- oder No-Code-Plattformen (Low Code: wenige Programmieranteile; No Code: ganz ohne Programmierkenntnisse), um erste Prototypen zu entwickeln, Anforderungen zu testen oder kleine Anwendungen im agilen Produktentwicklungsprozess umzusetzen. So können sie zum Beispiel eine App für Kundenfeedback, ein Dashboard zur Auswertung von Nutzungszahlen oder einen

Prototyp für einen Online-Shop entwerfen, bevor die finale Programmierung startet. Low Coding wird gezielt eingesetzt, um Entwicklung zu beschleunigen, Abstimmungsprozesse zu verkürzen oder Fachabteilungen stärker in die Gestaltung digitaler Produkte einzubeziehen. Auf diese Weise lassen sich innovative Ideen schnell sichtbar machen und in kurzen Zyklen verbessern.

Ingenieur:innen der Elektro- und Informationstechnik

Allgemeines Berufsbild:

Der Beruf der Ingenieur:innen der Elektro- und Informationstechnik verbindet Grundlagen aus Elektrotechnik, Elektronik und Informationstechnik. Berufstätige planen, entwickeln und steuern elektrische und elektronische Systeme. Dabei reicht das Spektrum von Stromversorgung und Energietechnik über Automatisierungstechnik bis hin zu Kommunikations- und Signalverarbeitungssystemen. Anwendung findet der Beruf in vielfältigen Bereichen – etwa in der Industrieautomatisierung, im Fahrzeug- und Maschinenbau, in der Medizintechnik, in der Energiewirtschaft oder in Forschung und Entwicklung. Typische Tätigkeiten sind die Entwicklung von Steuerungssystemen, die Integration von Sensorik und Aktorik, die Optimierung von Produktionsprozessen oder die Realisierung komplexer Kommunikationssysteme. In der Regel führt ein Studium oder duales Studium in Elektro- und Informationstechnik, Elektrotechnik, Embedded Systems oder Informatik mit Schwerpunkt Informationstechnik zu diesem Beruf. Möglich ist auch ein Weg über eine Ausbildung zur:zum Elektroniker:in mit anschließendem Studium. Auch hier gibt es alternative Wege – etwa über berufliche Qualifikationen oder Weiterbildungen. Der Quereinstieg ist unter bestimmten Voraussetzungen ebenfalls möglich.

Rolle im Kontext Low Coding:

Im Bereich Low Coding können Ingenieur:innen der Elektro- und Informationstechnik Low-Code-Plattformen einsetzen, um Steuerungen, Simulationen und Prototypen schneller zu entwickeln und zu testen – zum Beispiel bei der Ansteuerung eines Roboters in der Produktion, bei der Konfiguration eines Messsystems oder bei der Vernetzung mehrerer intelligenter Geräte.

Gerade in der Industrie 4.0 (digitale, vernetzte Produktion) und im Internet of Things (IoT, also die Verbindung von Geräten über das Internet) bieten Low-Code-Tools die Möglichkeit, ohne aufwendige, zeilenweise Programmierung einfache Schnittstellen, Bedienoberflächen oder Datenverarbeitungsprozesse zu erstellen. Ingenieur:innen nutzen Low Coding so, um Messdaten automatisiert auszuwerten, Geräte flexibel zu konfigurieren oder Abläufe visuell darzustellen. Bei Bedarf kombinieren sie diese Anwendungen mit klassischer Programmierung, um komplexere Systeme umzusetzen.

4. Unterrichtsverlauf: Vom Befehl zur Bedingung - Low Coding praxisnah erleben

Klassenstufe: 7

Länge: 90 Minuten

Ziel der Stunde:

Die Schüler:innen verstehen grundlegende algorithmische Prinzipien wie Sequenz, Schleife und Bedingung. Sie erkennen diese Strukturen in analogen wie digitalen Handlungsabläufen wieder und können sie gezielt einsetzen. In einer spielerischen Low-Code-Umgebung wenden sie diese Prinzipien an, dokumentieren ihr Vorgehen und reflektieren Lösungsstrategien. Abschließend übertragen sie das Gelernte auf berufliche Anwendungsszenarien und beurteilen Potenziale und Herausforderungen digitaler Prozessgestaltung mit Low Coding.

Phase/Zeit	Lehrkräfte- / Lernendenverhalten	Methodisch-didaktische Hinweise	Sozialform / Medien
Einstieg 10 Minuten	Die Lehrkraft wählt eine:n Lernende:n aus und positioniert ihn gerade mit etwas Abstand vor dem Mülleimer. Impuls: „Stell dir vor, [Name des Schülers/der Schülerin] ist ein Roboter. Du möchtest den Roboter zum Mülleimer steuern, damit er Papier für dich wegwerfen kann. Gib dem Roboter Befehle, damit er zum Mülleimer kommt.“ Nach dem ersten Durchlauf verändert die Lehrkraft das Setting. Es liegt nun kein geradliniger Aufbau mehr vor, sondern es wurden Hindernisse eingebaut. Ein weiterer Lernender wird ausgewählt und fungiert als Roboter, der gesteuert werden muss. Impuls: „Gib dem Roboter erneut Befehle, damit er zum Mülleimer kommt.“ Im Anschluss stellt die Lehrkraft folgende Anschlussimpulse: • „Beschreibe, was dir auffällt.“ • „Erläutere, wie du den Roboter gesteuert hast.“ Die Schüler:innen äußern sich. Sollte der Begriff nicht fallen, ergänzt die Lehrkraft in diesem Kontext den Begriff „Befehl“.	Erzeugung von Motivation und Interesse Erste Hinführung zum Stundenthema Handlungsorientierung Spielerisches Erarbeiten erster Befehlsketten	Plenum

Phase/Zeit	Lehrkräfte- / Lernendenverhalten	Methodisch-didaktische Hinweise	Sozialform / Medien
Zielangabe	„Heute wirst du eigene Befehlsketten erstellen und erfahren, was das mit dem Thema Coding zu tun hat.“	Transparenz	Plenum
Erarbeitung 15 Minuten	Die Lehrkraft zeigt am Smartboard eine vorbereitete Folie mit einer neuen Handlungsabfolge. Impuls: „Du siehst hier, wie eben in unserem Klassenzimmer, eine Handlungsabfolge, die ausgeführt werden muss. Beschreibe, was der Roboter hier tun muss, um an das Ziel zu kommen.“ Nachdem die Handlungssequenz aufgestellt wurde, führt die Lehrkraft den Begriff der Sequenz als feste Abfolge von Schritten ein. Die Lehrkraft präsentiert das zweite Beispiel. Auch dieses wird von den Lernenden gelöst. Anschließend definiert die Lehrkraft den Prozess der Wiederholung von Sequenzen als Schleife.	Erarbeiten erster Steuerungsstrukturen Einführen der Begriffe Sequenz und Schleife Visualisierung von Handlungsabfolgen Moderiertes Unterrichtsgespräch	Plenum Smartboard M1: Power-Point-Präsentation
Arbeitsphase 25 Minuten	Die Lehrkraft teilt Arbeitsblatt M2 aus. Impuls: „Du wirst nun solche Steuerungsstrukturen selbst ausprobieren. Dazu spielst du ein Spiel. Du benötigst dazu ein Tablet oder Smartphone. Du erreichst das Spiel über den QR-Code auf dem Arbeitsblatt.“ Begleitend zum Spiel füllen die Lernenden das Arbeitsblatt aus. Darauf notieren sie Aspekte zu folgenden Leitfragen: • Welche Befehle hast du genutzt? • Wo kam eine Schleife vor? • Wie bist du vorgegangen? • Wie oft hast du getestet?	Spielerisches Lernen Entdeckendes Lernen Anwenden der erarbeiteten Prinzipien	Einzelarbeit Tablet/Smartphone M2: Arbeitsblatt „Roboter steuern mit Low Coding“  https://sieya.de/game/low-coding

Phase/Zeit	Lehrkräfte- / Lernendenverhalten	Methodisch-didaktische Hinweise	Sozialform / Medien
Reflexion 10 Minuten	Die Schüler:innen reflektieren ihre Lösungsstrategien und die verwendeten Strukturen. Mögliche Impulse: • „Beschreibe, welche Stellen im Spiel für dich besonders schwierig waren und warum.“ • „Beurteile, wie leicht oder schwer dir das Coding gefallen ist.“ • „Erkläre, welche neuen Erkenntnisse du dabei gewonnen hast.“ • „Erläutere, in welchen Situationen du eine Schleife eingesetzt hast und warum das sinnvoll war.“ Die Schüler:innen äußern sich dazu. Impuls: „Ein Szenario haben wir bisher noch gar nicht angesprochen. Im Spiel kommen auch sogenannte Bedingungen vor, also Wenn-dann-Pfade. Beschreibe, wie dieses Prinzip funktioniert.“ Die Schüler:innen äußern sich dazu.	Reflektieren von Lösungen und Strategien Wertschätzung von Schüler:innenleistungen Bewusstmachen von Konzepten und Generalisierung der Ergebnisse	Plenum
Vertiefung 20 Minuten	Impuls: „Ihr habt euch nun schon mit Low Coding beschäftigt. Solche Programmiertechniken haben den Vorteil, dass man damit ohne viel Aufwand und Kenntnisse Anwendungen erstellen kann. Jetzt schauen wir uns an, in welchen Berufen so etwas im Alltag vorkommen kann.“ Die Lehrkraft verteilt an jede Kleingruppe zwei Stapel Karten: Berufe und Aufgaben.	Berufsorientierung Transfer Lebenswelt- und Zukunftsbezug Diskussionsimpuls	Gruppenarbeit M3: Berufe- und Aufgabenkarten

Phase/Zeit	Lehrkräfte- / Lernendenverhalten	Methodisch-didaktische Hinweise	Sozialform / Medien
	Impuls: „In diesem Stapel findet ihr Karten mit Berufsbeschreibungen – zum Beispiel Fachinformatiker:in oder IT-System-Manager:in. Im zweiten Stapel stehen konkrete Aufgaben, bei denen Low Coding eine Rolle spielt – etwa: ‚Erstellt ein digitales Formular für den Kundenservice‘ oder ‚Automatisiert eine Urlaubsanfrage‘. Eure Aufgabe ist es, die Berufe mit den passenden Aufgaben zu verbinden. Überlegt: Wer könnte sich in einem Unternehmen um welche Aufgabe kümmern? Und warum passt das zusammen? Wenn ihr eine Zuordnung gefunden habt, notiert bitte eine kurze Begründung. Warum ist diese Aufgabe typisch für diesen Beruf – oder: Warum könnte man sich das gut vorstellen?“ Die Schüler:innen bearbeiten die Aufgabe. Zum Abschluss bittet die Lehrkraft die Gruppen, ihre spannendste Zuordnung kurz zu präsentieren. Impuls: „Stellt uns eine Kombination vor, die ihr besonders interessant fandet. Begründet, warum Beruf und Aufgabe zusammenpassen und wie das Low Coding in der Tätigkeit genutzt wird.“ Weiterführende Impulse: • „Überlegt, welcher der Berufe euch persönlich am meisten anspricht – und warum.“ • „Erkläre, welche Aufgaben du dir gut für dich selbst vorstellen kannst.“ • „Stelle dar, welche beruflichen Tätigkeiten gut zu deinen Interessen und Stärken passen.“		

Phase/Zeit	Lehrkräfte- / Lernendenverhalten	Methodisch-didaktische Hinweise	Sozialform / Medien
Sicherung 10 Minuten	Die Lehrkraft leitet die Sicherung mit folgendem Impuls ein. Impuls: „Wir wiederholen jetzt in einem Code-Check spielerisch das, was ihr heute über Befehle, Sequenzen, Schleifen, Bedingungen und Low Coding gelernt habt. Ich lese euch Aussagen vor – ihr entscheidet, ob sie richtig oder falsch sind.“ Die Lehrkraft liest nacheinander vorbereitete Aussagen vor, die Lernenden reagieren per Daumen hoch (stimmt) oder Daumen runter (stimmt nicht). Richtige Aussagen (✓): • Eine Sequenz ist eine feste Abfolge von Befehlen. • Eine Schleife wiederholt eine bestimmte Befehlsfolge. • Eine Bedingung funktioniert nach dem Wenn-dann-Prinzip. • Beim Low Coding kann man Anwendungen mit wenigen Programmierkenntnissen erstellen. • Man kann Low Coding nutzen, um Arbeitsprozesse zu automatisieren. Falsche Aussagen (X): • Low Coding bedeutet, dass man alle Befehle manuell programmieren muss. • Eine Schleife führt nur einen Befehl einmal aus. • Eine Bedingung ist das Gleiche wie eine Schleife. • Sequenzen enthalten immer Bedingungen. • Nur Informatiker:innen können mit Low Coding arbeiten.	Spielerische Wiederholung der Begriffe und Prinzipien Sichtbarkeit der Ergebnisse	Plenum

Phase/Zeit	Lehrkräfte- / Lernendenverhalten	Methodisch-didaktische Hinweise	Sozialform / Medien
	Abschließender Impuls: „Was glaubt ihr: Wird Low Coding in der Zukunft wichtiger oder eher nicht so wichtig? Erklärt eure Meinung. Überlegt dann gemeinsam, welche Chancen und welche möglichen Probleme das für bestimmte Berufe bringen könnte.“ Die Schüler:innen äußern sich dazu. Die Lehrkraft fasst die Erkenntnisse, falls nötig, noch einmal abschließend zusammen.		

Methodisch-didaktische Umsetzung der Stunde

Die Stunde folgt einem handlungsorientierten und schüleraktivierenden Aufbau, der grundlegende algorithmische Prinzipien wie Sequenz, Schleife und Bedingung spielerisch vermittelt und mit digitalen Werkzeugen erfahrbar macht. Ziel ist es, den Lernenden erste Zugänge zum Thema Low Coding zu ermöglichen und sie zugleich für digitale Prozessgestaltung und deren Relevanz im beruflichen Kontext zu sensibilisieren.

Der Einstieg erfolgt über ein motivierendes Bewegungsszenario im Klassenraum, bei dem zunächst ein:e Schüler:in als „Roboter“ agiert. Die Mitschüler:innen geben verbale Befehle, um eine einfache Route zum Mülleimer zu steuern. Anschließend wird mit einem zweiten Lernenden eine Variation durchgeführt: Diesmal sind Hindernisse eingebaut, sodass die Befehle präziser und strukturierter formuliert werden müssen. Auf spielerische Weise wird so sichtbar, wie wichtig klare Anweisungen sind, wenn Abläufe gesteuert werden sollen. Dieser Einstieg schafft nicht nur eine hohe Aktivierung, sondern veranschaulicht auf einfache Weise zentrale Grundkonzepte des Programmierens. Die abstrahierende Einführung des Begriffs „Befehl“ erfolgt situativ und schülernah.

In der Erarbeitungsphase werden diese Konzepte anhand vorbereiteter Visualisierungen weiter systematisiert. Gemeinsam mit der Klasse analysiert die Lehrkraft Schrittfolgen und führt anschließend die Begriffe „Sequenz“ und „Schleife“ als wiederholende Steuerungsstruktur ein. Die gemeinsame Begriffsbildung erfolgt ausgehend von konkreten Beispielen und wird durch Visualisierung am Smartboard unterstützt. Damit wird der Grundstein für die eigenständige Anwendung algorithmischer Prinzipien gelegt.

Die anschließende Arbeitsphase nutzt ein digitales Spiel, das die erarbeiteten Strukturen aufgreift und den Schüler:innen ermöglicht, Low Coding direkt und eigenständig auszuprobieren. Während des Spiels dokumentieren die Lernenden ihr Vorgehen mithilfe eines Arbeitsblatts, das zur Reflexion über genutzte Befehle, Strategien und erste Schleifenanwendungen anregt. Diese Phase bietet Raum für individuelles und entdeckendes Lernen und erlaubt zugleich eine begleitete Fehlerkultur durch wiederholtes Testen und Anpassen.

Die Reflexionsphase greift zentrale Erfahrungen aus dem Spiel auf und vertieft sie durch gezielte Impulse zur Selbstreflexion. Besonderer Fokus liegt auf der Einführung des Begriffs „Bedingung“ (Wenn-dann-Strukturen), der aus dem Spielverlauf heraus aufgegriffen und gemeinsam analysiert wird. So wird die dritte grundlegende Steuerungsstruktur im Prozess des Reflektierens bewusstgemacht.

Im Anschluss weitet die Vertiefungsphase den Blick in Richtung Berufsorientierung. Die Schüler:innen erhalten Karten mit Berufsbeschreibungen sowie konkreten Aufgabenstellungen, bei denen Low Coding eine Rolle spielt. In Gruppen kombinieren sie passende Paare und begründen ihre Auswahl. Durch diesen Transfer erkennen die Lernenden, dass die erarbeiteten Prinzipien nicht nur in Spielen, sondern auch in realen Berufsfeldern von Bedeutung sind. Der Austausch über persönliche Interessen und Stärken fördert zusätzlich die Selbstorientierung im Hinblick auf zukünftige Berufswahlprozesse.

Abgeschlossen wird die Stunde durch eine spielerische Wiederholung in Form eines Code-Checks. Die Lehrkraft liest richtige und falsche Aussagen zu den zentralen Begriffen vor. Die Lernenden reagieren per Daumenzeichen, wodurch ein niedrigschwelliger, nonverbaler Rückblick auf die Inhalte ermöglicht wird. Abschließend diskutieren die Schüler:innen die Bedeutung von Low Coding für zukünftige Berufswelten und benennen Chancen sowie Herausforderungen digitaler Automatisierung.

Die methodische Stärke der Stunde liegt in der engen Verzahnung von spielerischem, handlungsorientiertem Lernen, konzeptueller Begriffsbildung und berufsbezogenem Transfer. Die gewählten Sozialformen und Materialien ermöglichen eine hohe Aktivierung, individuelle Lernwege und motivierende Differenzierung. Gleichzeitig werden zentrale Zukunftskompetenzen wie digitale Problemlösefähigkeit, algorithmisches Denken und Berufswahlkompetenz gefördert.

5. Arbeitsmaterialien

Vom Befehl zur Bedingung - Low Coding praxisnah erleben

Roboter steuern mit Low Coding

- 1 Scanne den QR-Code und rufe das Spiel „The last one coding“ auf.



- 2 Spiele alle Level. Notiere dir währenddessen oder danach die wichtigsten Punkte zum Spiel. Orientiere dich dabei an den folgenden Aspekten:

a. Erkläre, wie du vorgegangen bist.

b. Nenne ein Beispiel für eine Schleife im Spiel. Erkläre ihre Funktion.

c. Im Spiel gibt es noch ein weiteres Prinzip beim Coding: die Bedingung. Erkläre, was das Prinzip bedeutet und wie du vorgegangen bist.

Elektroniker:in für Automatisierungstechnik

richtet Maschinen so ein, dass sie
selbstständig arbeiten können,
z.B. in Industrieanlagen oder Gebäuden

Mechatroniker:in

baut Maschinen zusammen und repariert sie,
wenn etwas nicht funktioniert

Fachinformatiker:in für Systemintegration

sorgt dafür, dass Computer und Programme
gut zusammenarbeiten

IT-System-Manager:in

plant, wie Computer und Programme in einer
Firma eingesetzt werden

Kauffrau/Kaufmann für Digitalisierungsmanagement

hilft einer Firma, mit digitalen Ideen schneller
und besser zu arbeiten

Ingenieur für Elektro- und Informationstechnik

plant, wie elektrische Geräte funktionieren
und wie man sie steuert



ein Formular bauen, mit dem Urlaubsanträge online abgeschickt werden können

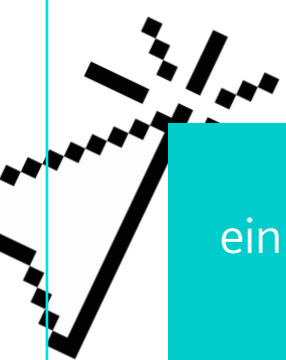
eine einfache Steuerung für ein Fließband in einer Fabrik erstellen

eine App bauen, die Messwerte von Sensoren zeigt, z.B. Temperatur und Licht

dafür sorgen, dass fehlende Ersatzteile automatisch bestellt werden

ein Diagramm erstellen, das den Stromverbrauch einer Maschine zeigt





ein System einrichten, das Techniker:innen automatisch eine Nachricht schickt, wenn etwas kaputtgeht

ein kleines Online-Training für Mitarbeitende gestalten
– ohne selbst Programmieren zu müssen

einen Online-Shop so verbinden, dass er automatisch den Lagerbestand anzeigt

am Computer testen, wie eine Maschine am besten arbeitet – ohne sie in echt anzuhalten

ein Programm bauen, das Kundendaten automatisch in eine Tabelle einträgt



M3: Musterlösung Zuordnung Berufe- und Aufgabenkarten

Beruf	Aufgaben
Elektroniker:in für Automatisierungstechnik	• eine einfache Steuerung für ein Fließband in einer Fabrik erstellen • ein System einrichten, das Techniker:innen automatisch eine Nachricht schickt, wenn etwas kaputtgeht
Mechatroniker:in	• dafür sorgen, dass fehlende Ersatzteile automatisch bestellt werden
Fachinformatiker:in für Systemintegration	• einen Online-Shop so verbinden, dass er automatisch den Lagerbestand anzeigt • ein Programm bauen, das Kundendaten automatisch in eine Tabelle einträgt
IT-System-Manager:in / Digital Product Manager:in	• ein Formular bauen, mit dem Urlaubsanträge online abgeschickt werden können • ein kleines Online-Training für Mitarbeitende gestalten – ohne selbst programmieren zu müssen
Kauffrau/Kaufmann für Digitalisierungsmanagement	• ein Diagramm erstellen, das den Stromverbrauch einer Maschine zeigt
Ingenieur:in für Elektro- und Informationstechnik	• am Computer testen, wie eine Maschine am besten arbeitet – ohne sie in echt anzuhalten • eine App bauen, die Messwerte von Sensoren zeigt, z. B. Temperatur und Licht

Hinweis: Die dargestellten Zuordnungen sind die naheliegendsten und orientieren sich an den Kernaufgaben der jeweiligen Berufe. Aufgrund der Breite und Überschneidungen der Berufsfelder sind auch andere Zuordnungsmöglichkeiten denkbar und können im Unterricht thematisiert werden.

6. Musterlösung Minispiel Low Coding

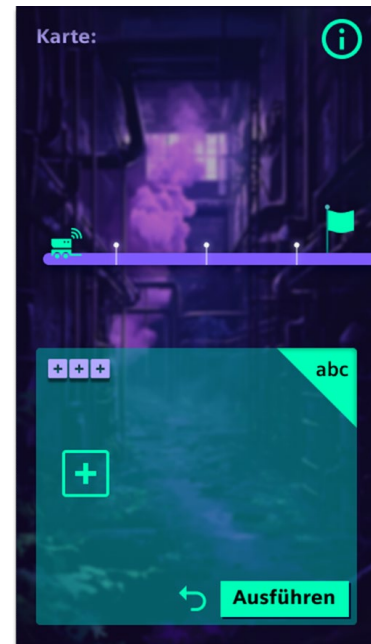
In einem Dystopie-Szenario nehmen die Schüler:innen als Teil eines Einsatzkommandos an einer Roboter-Rettungsmission teil und steuern die Maschine mit Hilfe von Low Coding aus einer Fabrik.

Das Spiel ist in fünf Level unterteilt, die aufeinander aufbauend grundlegende Prinzipien des Low Codings vermitteln. Spieler:innen steuern darin einen Roboter durch eine alte Fabrik, indem sie passende Code-Blöcke auswählen und kombinieren. So lernen sie auf spielerische und einfache Weise zentrale Konzepte der Informatik kennen – etwa Sequenzen, Schleifen oder Bedingungen – und wenden diese an, um den Roboter geschickt an Hindernissen vorbeizumaneuvrieren.

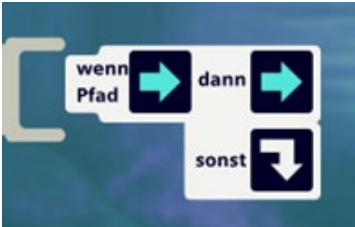
Auf einer Karte sehen die Spieler:innen, welche Route der Roboter nehmen muss. Für jedes Level dürfen nur eine bestimmte Menge an Blöcken verwendet werden. Fehlermeldungen geben kleine Hinweise, was an der momentanen Block-Kombination nicht funktioniert. Die Spieler:innen haben zusätzlich zu der Block-Darstellung die Möglichkeit, ihren gebauten Code als Text anzuschauen.

Nach Abschluss des fünften Levels ist das Spiel gewonnen – der Roboter hat es aus der einstürzenden Fabrik geschafft.

Genereller Hinweis: Bei Verständnisproblemen kann es helfen, sich die Strecke auf Papier aufzuzeichnen und mit einer Büroklammer als Roboter die Anweisungen Schritt für Schritt auszuführen und damit nachzuvollziehen.



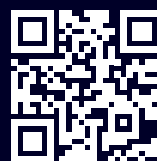
Level	Auftrag	Lösung
1.	Der Roboter soll eine gerade Strecke über drei Abschnitte fahren. Konzept: Befehl (einzelne Anweisung: „geradeaus fahren“) + Sequenz (mehrere Befehle hintereinander)	
Hinweis: Auch wenn schon weitere Blöcke zum Abbiegen zur Auswahl stehen, muss hier nur der Geradeaus-Block verwendet werden.		
2	Der Roboter soll einem Hindernis ausweichen. Konzept: Befehl (Drehung, Vorwärtsbewegung) + Sequenz (Abfolge von Befehlen, um Hindernis zu umfahren)	
Hinweis: Es ist wichtig, den Roboter, nachdem er sich dreht auch mit dem entsprechenden Block weiter geradeaus fahren zu lassen. Sonst dreht er sich nur nach rechts und links auf der Stelle.		
3	Der Roboter soll mehreren Hindernissen ausweichen. Konzept: Schleife (wiederholte Ausführung einer Sequenz, um Platz zu sparen und Wiederholung zu verdeutlichen) + Befehl + Sequenz.	
Hinweis: Da nicht genügend Blockslots zu Verfügung stehen, muss hier der Schleifen-Block eingesetzt werden, auch wenn theoretisch einfach die Lösung von Level 2 übernommen und erweitert werden könnte, um den Roboter zum Ziel zu bewegen. Der gleiche Ansatz wird auch in den folgenden Levels verwendet, um die Nutzung der komplexeren Blöcke zu erzwingen, da dies auch im richtigen Coding die Effizienz, Robustheit und Übersichtlichkeit erhöht.		

Level	Auftrag	Lösung
4	Der Roboter soll mehreren Hindernissen ausweichen und dabei mehrere Abzweigungen vermeiden. Konzept: <i>Bedingung (wenn ... dann ...), um eine Entscheidung an einer Abzweigung zu treffen + Schleife (für wiederholtes Umfahren von Hindernissen).</i>	
Hinweis: Die „umgekehrte“ Variante, wo der Rechts-Abbiegen-Block und der Geradeaus-Block im Vergleich zur Musterlösung vertauscht wären, führt bereits im ersten Schritt zu Problemen: Der Roboter fährt geradeaus und dreht sich danach, ohne weitere Prüfung nach rechts und damit zur Wand. Auch wenn das Programm von hier aus noch weiterlaufen kann, würde der Roboter letztlich an der ersten Kreuzung in einer Endlosschleife gefangen werden.		
5	Der Roboter soll den kürzesten Weg zum Ziel nehmen. Konzept: <i>Bedingung (wenn ... dann ... sonst ...), um verschiedene Pfade zu vergleichen und den kürzesten auszuwählen + Sequenz</i>	
Hinweis: Der Pfad mit den wenigsten Kurven wird hier als der schnellste Weg zum Ziel betrachtet, auch wenn es einen anderen Pfad mit der gleichen Distanz (Menge an Abschnitten) zum Ziel gibt.		

Herausgeber

Siemens Aktiengesellschaft
Communications and Government Affairs
Werner-von-Siemens-Straße 1
80333 Munich
Germany

Commercial Registry Number:
HRB 6684, Amtsgericht München



SIEMENS