Schulinterner Lehrplan
des Freiherr-vom-Stein-Gymnasiums
zum Kernlehrplan für die gymnasiale Oberstufe
im Fach

Informatik

für die Qualifikationsphase (Grundkurs)

Stand: Oktober 2023

Vorbemerkung[1]

Die folgenden Kompetenzen aus dem Bereich *Kommunizieren und Kooperieren* werden in allen Unterrichtsvorhaben der Qualifikationsphase vertieft und sollen aus Gründen der Lesbarkeit nicht in jedem Unterrichtsvorhaben separat aufgeführt werden:

Die Schülerinnen und Schüler

- verwenden die Fachsprache bei der Kommunikation über informatische Sachverhalte (K),
- nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung von Dateien unter Berücksichtigung der Rechteverwaltung (K),
- organisieren und koordinieren kooperatives und eigenverantwortliches Arbeiten (K),
- strukturieren den Arbeitsprozess, vereinbaren Schnittstellen und führen Ergebnisse zusammen (K),
- beurteilen Arbeitsorganisation, Arbeitsabläufe und Ergebnisse (K),
- präsentieren Arbeitsabläufe und -ergebnisse adressatengerecht (K).

Weitere zentrale Kompetenzen werden in den konkreten Kompetenzerwartungen in Klammern spezifiziert.

- Argumentieren (A)
- Modellieren (M)
- Implementieren (I)
- Darstellen und Interpretieren (D)
- Kommunizieren und Kooperieren (K, s.o.)

[1] Der Lehrplan gilt in dieser Version für den Abiturjahrgang 2022. Der Abiturjahrgang 2021 erwirbt zu Beginn der Q1.1 bereits erste grundlegende Kompetenzen zu Netzwerken (Sequenzierung Punkt 1 des sechsten Unterrichtsvorhabens.). Die restliche Reihenfolge bleibt unberührt.

Erstes Unterrichtsvorhaben (Q1.I):

Thema: Wiederholung der objektorientierten Modellierung und Programmierung, Wiederholung der Datenstruktur Array, zweidimensionale Arrays

Leitfragen: Wie modelliert und implementiert man zu einer Problemstellung in einem geeigneten Anwendungskontext Java-Klassen inklusive ihrer Attribute, Methoden und Beziehungen? Wie kann man die Modellierung und die Funktionsweise der Anwendung grafisch darstellen?

Vorhabenbezogenen Konkretisierung:
Zu einer Problemstellung in einem Anwendungskontext soll eine Java-Anwendung entwickelt werden. Um die Aufgabe einzugrenzen, können (nach der ersten Problemanalyse) einige Teile (Modellierungen oder Teile von Java-Klassen) vorgegeben werden.
Die Schülerinnen und Schülern erläutern und modifizieren den ersten Entwurf und modellieren sowie implementieren weitere Klassen und Methoden für eine entsprechende Anwendung. Klassen und ihre Beziehungen werden in einem Implementationsdiagramm dargestellt.

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen

Inhaltliche Schwerpunkte:
- Objekte und Klassen
- Analyse, Entwurf und Implementierung von Algorithmen
- Syntax und Semantik einer Programmiersprache

Zu entwickelnde Kompetenzen:
Die Schülerinnen und Schüler
- analysieren und erläutern objektorientierte Modellierungen (A)
- beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A)
- modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M)
- ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M)
- modellieren ggf. abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M)
- implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I)
- nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I)
- wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I)
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I)
- stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D)
- dokumentieren Klassen (D)

Beispiele: Lotto, Kontaktbuch (1-dimensionales Array)
Wetterstation, Pausenspiel (2-dimensionales Array)

Zeitbedarf: ca. 12 Stunden (= 4 Wochen)

Zweites Unterrichtsvorhaben (Q1.I)

Thema: Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen

Leitfrage: Wie können beliebig viele linear angeordnete Daten im Anwendungskontext verwaltet werden?

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen

Inhaltliche Schwerpunkte:
- Objekte und Klassen
- Analyse, Entwurf und Implementierung von Algorithmen
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache

Vorhabenbezogene Konkretisierung:
Nach Analyse einer Problemstellung in einem geeigneten Anwendungskontext, in dem Daten nach dem First-In-First-Out-Prinzip verwaltet werden, werden der Aufbau von Schlangen am Beispiel dargestellt und die Operationen der Klasse `Queue` erläutert. Anschließend werden für die Anwendung notwendige Klassen modelliert und implementiert. Die Klasse `Queue` wird, ebenso wie die im weiteren Verlauf verwendeten Klassen `Stack` und `List`, vom Land bereitgestellt. Anschließend wird die Anwendung modifiziert, um den Umgang mit der Datenstruktur zu üben. Anhand einer Anwendung, in der Daten nach dem Last-In-First-Out-Prinzip verwaltet werden, werden Unterschiede zwischen den Datenstrukturen Schlange und Stapel (`Stack`) erarbeitet. Um einfacher an Objekte zu gelangen, die zwischen anderen gespeichert sind, wird die Klasse `List` eingeführt und in einem Anwendungskontext verwendet. In mindestens einem weiteren Anwendungskontext wird die Verwaltung von Daten in Schlangen, Stapeln oder Listen vertieft. Modellierungen werden dabei in Entwurfs- und Implementationsdiagrammen dargestellt.

Zu entwickelnde Kompetenzen:
Die Schülerinnen und Schüler
- erläutern Operationen dynamischer Datenstrukturen (A)
- analysieren und erläutern Algorithmen und Programme (A)
- beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A)
- ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M)
- ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M)
- modellieren abstrakte und nicht abstrakte Klassen (M)
- verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M)
- modifizieren Algorithmen und Programme (I)
- implementieren Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I)
- nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I)
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I)
- testen Programme systematisch anhand von Beispielen (I)
- stellen Strukturen grafisch dar und erläutern ihren Aufbau (D)

Beispiele gemäß der Sequenzierung des Unterrichtsvorhabens:

1. Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue
   Beispiele: Sprechstunde, Verkehrskontrolle, Mensa, Disco
2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack
   Beispiele: Münzstapel, Palindrom, Biber und Teller
3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List
   Beispiele: Einkaufsliste, Vokabeltrainer
4. Vertiefung – Anwendungen von Listen, Stapeln oder Schlangen in einem weiteren Kontext
   Beispiele: MauMau, Rangierbahnhof

Zeitbedarf: ca. 48 Stunden (= 16 Wochen)

Drittes Unterrichtsvorhaben (Q1.I-II)

Thema: Rekursion, Such- und Sortieralgorithmen

Leitfrage: Wie kann man gespeicherte Informationen günstig (wieder-)finden?

Inhaltsfeld:
- Algorithmen

Inhaltliche Schwerpunkte:
- Analyse, Entwurf und Implementierung von Algorithmen
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache

Zu entwickelnde Kompetenzen:
Die Schülerinnen und Schüler
- analysieren und erläutern Algorithmen und Programme (A)
- beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A)
- beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A)
- entwickeln und implementieren iterative und rekursive Algorithmen (M)
- modifizieren Algorithmen und Programme (I)
- implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I)
- nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I)
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I)
- testen Programme systematisch anhand von Beispielen (I)
- stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D)

Sequenzierung des Unterrichtsvorhabens:
1. Das Prinzip Rekursion
2. rekursives vs. iteratives Suchen von Daten in Listen und Arrays
3. Sortieren in Listen und Arrays – Entwicklung, Implementierung und Bewertung von iterativen und rekursiven Sortierverfahren

Beispiele: Mathematische und graphische Rekursion, binäre Suche, Quicksort-Algorithmus, MinSort-Algorithmus iterativ und rekursiv

Zeitbedarf: ca. 24 Stunden (= 8 Wochen)

Viertes Unterrichtsvorhaben (Q1.II):

Thema: Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen (Bäume)

Leitfragen: Wie können Daten im Anwendungskontext mit Hilfe binärer Baumstrukturen verwaltet werden? Wie kann dabei der rekursive Aufbau der Baumstruktur genutzt werden? Welche Vor- und Nachteile haben Suchbäume für die geordnete Verwaltung von Daten?

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen

Inhaltliche Schwerpunkte:
- Objekte und Klassen
- Analyse, Entwurf und Implementierung von Algorithmen
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache

Zu entwickelnde Kompetenzen
Die Schülerinnen und Schüler
- erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A)
- analysieren, erläutern und modifizieren Algorithmen und Programme (A, I)
- beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A)
- ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M)
- ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M)
- implementieren iterative und rekursive Algorithmen und stellen sie dar (I, D)
- nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I)
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I)
- testen Programme systematisch anhand von Beispielen (I)
- stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D)

Beispiele gemäß möglicher Sequenzierung des Unterrichtsvorhabens:
1. **Analyse der Baumstruktur mit grundlegenden Begriffen** (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit)
   Beispiel: Termbaum
2. **Die Datenstruktur Binärbaum im Anwendungskontext unter Verwendung der Klasse `BinaryTree`**
   (a) Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten
   (b) Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf
   (c) Implementierung der Traversierungen unter Nutzung der Klasse `BinaryTree`
   Beispiele: Ahnenbaum, Codierbaum (Morsealphabet)
3. **Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse `BinarySearchTree`**
   Beispiele: Nutzerverwaltung, Nicknames, Kochbuch

Zeitbedarf: ca. 30 Stunden (10 Wochen)

Fünftes Unterrichtsvorhaben (Q1.II):

Thema: Prinzipielle Arbeitsweise eines Computers

Leitfragen: Was sind die strukturellen Hauptbestandteile eines Computers und wie kann man sich die Ausführung eines maschinennahen Programms mit diesen Komponenten vorstellen?

Inhaltsfelder:
- Informatiksysteme

Inhaltliche Schwerpunkte:
- Einzelrechner und Rechnernetzwerke

Sequenzierung des Unterrichtsvorhabens:
Von-Neumann-Architektur und die Ausführung maschinennaher Programme
a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher
b) Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms

Zu entwickelnde Kompetenzen:
Die Schülerinnen und Schüler
- erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A)

Zeitbedarf: ca. 6 Stunden (= 2 Wochen)

Sechstes Unterrichtsvorhaben (Q2.I):

Thema: Kommunikation in Netzstrukturen

Leitfragen: Wie werden Daten in Netzwerken übermittelt?

Inhaltsfelder:
- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
- Einzelrechner und Rechnernetzwerke
- Nutzung von Informatiksystemen

Zu entwickelnde Kompetenzen:
Die Schülerinnen und Schüler
- beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A)
- entwickeln virtuelle Netzwerke (M)
- untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A)
- untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A)

Sequenzierung des Unterrichtsvorhabens:
1. Daten in Netzwerken und Sicherheitsaspekte in Netzen
   (a) Definition und Klassifizierungen von Netzwerken (Netztopologien, Aufgabenteilung, räumliche Ausdehnung)
   (b) TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz
   (c) Virtueller Bau eines Netzwerkes: IP-Adressen, Subnetting, filius
2. Analyse und Weiterentwicklung eines gegebenen Protokolls in einer Client-Server-Architektur
3. ggf. Implementation einer Client-Server-Anwendung, etwa eines Einkaufsservers, unter Verwendung der vom Land bereitgestellten Netzwerk-Klassen und -Implementationen; Entwicklung eines auf die konkrete Anwendung bezogenen Protokolls

Zeitbedarf: ca. 15 Stunden (5 Wochen)

Siebtes Unterrichtsvorhaben (Q2.I):

Thema: Sicherheit und Datenschutz (Kryptografie)

Leitfragen: Wie können Daten sicher verschlüsselt werden?

Inhaltsfelder:
- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
- Sicherheit
- Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht
- Nutzung von Informatiksystemen, Wirkungen der Automatisierung

Konkretisierung des Unterrichtsvorhabens: symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-Verfahren) als Methoden, Daten im Netz verschlüsselt zu übertragen

Zu entwickelnde Kompetenzen:
Die Schülerinnen und Schüler
- analysieren, erläutern und ggf. implementieren Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A, I)

Zeitbedarf: ca. 15 Stunden (5 Wochen)

Achtes Unterrichtsvorhaben (Q2.I):

Thema: Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Leitfragen: Wie können Fragestellungen mit Hilfe einer Datenbank beantwortet werden? Wie entwickelt man selbst eine Datenbank für einen Anwendungskontext?

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
- Datenbanken
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache
- Sicherheit

Zu entwickelnde Kompetenzen:
Die Schülerinnen und Schüler
- erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A)
- analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A)
- analysieren und erläutern eine Datenbankmodellierung (A)
- erläutern die Eigenschaften normalisierter Datenbankschemata (A)
- bestimmen Primär- und Sekundärschlüssel (M)
- ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M)
- modifizieren eine Datenbankmodellierung (M)
- modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M)
- bestimmen Primär- und Sekundärschlüssel (M)
- überführen Datenbankschemata in vorgegebene Normalformen (M)
- verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einem Datenbanksystem zu extrahieren (I)
- ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D)
- stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D)
- überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D)

Beispiele / mögliche Sequenzierung des Unterrichtsvorhabens

1. **Modellierung von relationalen Datenbanken**
   (a) Entity-Relationship-Diagramm
   - Ermittlung von Entitäten, zugehörigen Attributen, Relationen und Kardinalitäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms
   - Erläuterung und Modifizierung einer Datenbankmodellierung

   (b) Entwicklung einer Datenbank aus einem Datenbankentwurf
   - Modellierung eines relationalen Datenbankschemas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln

   (c) Redundanz, Konsistenz und Normalformen
   - Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz in einer Anwendungssituation
   - Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten)

   Beispiele: Schulverwaltung, Fahrradverleih, Reederei, Buchungssystem

2. **Nutzung von relationalen Datenbanken**
   (a) Aufbau von Datenbanken und Grundbegriffe
   - Entwicklung von Fragestellungen zur vorhandenen Datenbank
   - Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema

   (b) SQL-Abfragen
   - Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (SELECT (DISTINCT) …FROM, WHERE, AND, OR, NOT) auf einer Tabelle
   - Analyse und Erarbeitung von SQL-Abfragen auf einer und mehrerer Tabelle zur Beantwortung der Fragestellungen (JOIN, UNION, AS, GROUP BY,ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (…), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL)
     Für die Erlernung, Anwendung und Überprüfung von SQL-Abfragen kann auf ein Online-Lern-Tutorial zurückgegriffen werden

   (c) Implementation einer Datenbankanwendung unter Verwendung der vom Land bereitgestellten Datenbank-Klassen und einer geeigneten Datenbanksimulation, etwa SQLite.

   Beispiele: cia-Datenbank, Nordwind-Datenbank, VideoCenter, Schulbuchausleihe

Zeitbedarf: ca. 30 Stunden (10 Wochen)

Neuntes Unterrichtsvorhaben (Q2.II):

Thema: (Deterministische und nicht-deterministische) Endliche Automaten und formale Sprachen

Leitfragen: Wie kann man (endliche) Automaten genau beschreiben? Wie können alltägliche Kontexte oder informatische Problemstellungen mit endlichen Automaten modelliert werden? Wie können Sprachen durch Grammatiken beschrieben werden? Welche Zusammenhänge gibt es zwischen formalen Sprachen, endlichen Automaten und regulären Grammatiken?

Inhaltsfelder:
- Endliche Automaten und formale Sprachen

Inhaltliche Schwerpunkte:
- Endliche Automaten
- Grammatiken regulärer Sprachen
- Möglichkeiten und Grenzen von Automaten und formalen Sprachen

Sequenzierung des Unterrichtsvorhabens:
1. Endliche Automaten
   (a) Vom Automaten in den Schülerinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten
   (b) Untersuchung, Darstellung und Entwicklung endlicher Automaten
   Beispiel: Kaugummiautomat, Schatzsuche
2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen
   (a) Erarbeitung der formalen Darstellung regulärer Grammatiken
   (b) Untersuchung, Modifikation und Entwicklung von Grammatiken
   (c) Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen, die durch Grammatiken gegeben werden
   Beispiel: Die Software *Exorcizer* zur selbstständigen Entwicklung eines von Automaten für eine vorgegebene Sprache
   (d) Entwicklung regulärer Grammatiken zu endlichen Automaten
3. Grenzen endlicher Automaten

Zu entwickelnde Kompetenzen:
Die Schülerinnen und Schüler
- analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A)
- analysieren und erläutern Grammatiken regulärer Sprachen (A)
- zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A)
- ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A)
- entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M)
- entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M)
- entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M)
- modifizieren Grammatiken regulärer Sprachen (M)
- entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M)
- stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D)
- ermitteln die Sprache, die ein endlicher Automat akzeptiert (D)
- beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D)

Zeitbedarf: ca. 24 Stunden (8 Wochen)

Zehntes Unterrichtsvorhaben Q2-II:

Thema: Grenzen der Automatisierbarkeit

Leitfragen: Welche Möglichkeiten bieten Informatiksysteme und wo liegen ihre Grenzen?

Inhaltsfelder:
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
- Grenzen der Automatisierung

Zu entwickelnde Kompetenzen:
Die Schülerinnen und Schüler
- untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A)

Zeitbedarf: ca. 6 Stunden (2 Wochen)