Sequenz der Inhalte	**Zu entwickelnde Kompetenzen** Die Schülerinnen und Schüler ...	**Hinweise zu Kontexten, Medien, Materialien**
Unterrichtsvorhaben Q1-1 **Vom Problem zum Programm** a) Formulierung von einfachen Algorithmen mit den Mitteln einer imperativen Programmiersprache (Kontrollstrukturen und vordefinierte Datentypen) b) Verwendung von Klassen und Objekten als zusätzliches Strukturierungsmittel bei komplexeren Programmen, Wiederholung und Verfeinerung der Darstellung in Klassen- und Beziehungsdiagrammen c) Einführung von abstrakten Oberklassen und Wiederholung von Polymorphie d) Eventuell: Exkurs zu ereignisgesteuerter Programmierung Inhaltsfelder: - Algorithmen, - Daten und ihre Strukturierung - Formale Sprachen und Automaten - Informatiksysteme Zeitbedarf: ca. 24 Stunden	- analysieren und erläutern Algorithmen und Programme (A) - testen Programme systematisch anhand von Beispielen (I) - modifizieren Algorithmen und Programme (I) - stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D) - nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I) - 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) - interpretieren Fehlermeldungen und korrigieren den Quellcode (I) - modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M) - modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M) - verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M) - ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M) - stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D) - dokumentieren Klassen (D) - implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I) - stellen die Kommunikation zwischen Objekten grafisch dar (D)	Durch Kontrastierung mit einer anderen Sprache (z.B. Python) können aus der EF bekannte Spracheigenschaften von Java wiederholt und deutlicher herausgearbeitet werden. Für einfache Algorithmen ist eine Formulierung als Funktion (ohne Klassen und Objekte) bereits ausreichend (z.B. Kalenderproblem: Wochentagsbestimmung). Ebenso kann hier der Unterschied von iterativen und rekursiven Algorithmen bei einfachen Funktionen klarer eingeführt werden (z.B. Stringbearbeitung, Fakultät). Bei komplexeren Programmen oder wenn eine grafische Benutzeroberfläche benötigt wird, ist ein objektorientierter Ansatz sinnvoll. Die umfangreichen Kenntnisse zu Klassen- und Beziehungsdiagrammen aus der EF sollen hier gebündelt wiederholt werden. Für die Einführung abstrakter Oberklassen und der Polymorphie bietet sich ein Vektorgrafikprojekt an (vgl. Schriek, Bd. I, Kapitel 8). Dabei können die aus der EF bekannten Modellierungs- und Darstellungskompetenzen zur Objektorientierung auch gut wiederholt und geübt werden (Klassen- und Beziehungsdiagramme). Ereignissteuerung kann höchstens kurz angesprochen werden, für eine ausführliche Erarbeitung fehlt die Zeit. Exemplarisch könnte aber der Programmgenerator von sum vorgestellt werden.

Sequenz der Inhalte	**Zu entwickelnde Kompetenzen** Die Schülerinnen und Schüler ...	**Hinweise zu Kontexten, Medien, Materialien**
Unterrichtsvorhaben Q1-2 **Dynamische lineare Datenstrukturen** a) Die Datenstruktur Schlange im Anwendungskontext bis zur Nutzung der allgemeinen Klasse Queue b) Verallgemeinerung / Erweiterung zur Datenstruktur lineare Liste und die Klasse List c) Abwandlung zur Datenstruktur Stapel und die Klasse Stack Inhaltsfelder: - Daten und ihre Strukturierung - Algorithmen - Formale Sprachen und Automaten Zeitbedarf: ca. 20 Stunden	- analysieren und erläutern Algorithmen und Programme (A) - testen Programme systematisch anhand von Beispielen (I) - nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I) - interpretieren Fehlermeldungen und korrigieren den Quellcode (I) - implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I) - erläutern Operationen dynamischer (linearer oder nichtlinearer) Datenstrukturen (A) - ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekt-typen sowie lineare und nichtlineare Datensammlungen zu (M) - stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D) - analysieren und erläutern objektorientierte Modellierungen (A) - ermitteln bei der Analyse von Problemstel-lungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M)	Die Simulation eines Patientenwartezimmers (Schriek Band II, Kapitel 4) ist ein bewährtes Beispiel zur Einführung des Datenyps Warteschlange. Lineare Liste: Schriek II, Kapitel 5. Stapel: Schriek II, Kapitel 6. Die Bedeutung des Stapels für interne Programmabläufe (z.B.Sprungbefehle) im Rechner kann gut am Beispiel des Terminterpreters erforscht werden. Das Thema wird auch später (in Q1-4) mit dem stapelorientierten Modellrechner wieder aufgenommen.

Sequenz der Inhalte	**Zu entwickelnde Kompetenzen** Die Schülerinnen und Schüler ...	**Hinweise zu Kontexten, Medien, Materialien**
Unterrichtsvorhaben Q1-3 **Suchen und Sortieren auf linearen Datenstrukturen** a) Suchen (1) Lineare Suche in Liste und Array (2) Binäre Suche in einem Array (3) Vergleich der Effizienz von linearer und binärer Suche b) Sortieren (1) iterative Verfahren (z.B. Bubblesort, Sortieren durch Einfügen) (2) rekursive Verfahren (Quicksort, evtl. weitere Verfahren) (3) Effizienzbetrachtungen Inhaltsfelder: - Algorithmen Zeitbedarf: ca. 16 Stunden	-entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M) - implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I) - testen Programme systematisch anhand von Beispielen (I) - analysieren und erläutern Algorithmen und Programme (A) - modifizieren Algorithmen und Programme (I) - implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I) - beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A) - nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zur Erschließung, Aufbereitung und Präsentation fachlicher Inhalte (D)	Die Reihenfolge zwischen Suchen und Sortieren ist nicht festgelegt. Suchen: Schriek II, Kapitel 9 Sortieren: Schriek II, Kapitel 8 Als schneller Zugang zu Sortierverfahren hat sich auch bewährt, die Algorithmen als Python-Funktionen zu implementieren und im Python Befehlsinterpreter zu untersuchen. Dabei kann der Python-Datentyp list wie ein Array oder wie eine Liste verwendet werden. Als weitere Übung können Python-Funktionen (etwa die zu Quicksort) anschließend nach Java portiert werden.

Sequenz der Inhalte	**Zu entwickelnde Kompetenzen** Die Schülerinnen und Schüler ...	**Hinweise zu Kontexten, Medien, Materialien**
Unterrichtsvorhaben Q1-4 **Technische Aspekte von Informatiksystemen** a) Exemplarische Übersetzung von Befehlen einer „Hochsprache“ in Maschinenbefehle am Beispiel eines Modellrechners b) Aufbau und Beschreibung von Systemen vernetzter Rechner - Netzwerktopologien, Client-Server-Modell - OSI-Schichtenmodell, insbesondere am Beispiel IP-Netze - Grundwissen zum Internet Inhaltsfelder: - Informatiksysteme Zeitbedarf: ca.15 Stunden	- erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A) - beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie das Schictenmodell in Netzwerken (A) - nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zur Erschließung, Aufbereitung und Präsentation fachlicher Inhalte (D) - nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung von Dateien unter Berücksichtigung der Rechteverwaltung (K) - stellen die Kommunikation zwischen Objekten grafisch dar (D)	Programmieren im Stapelorientierten Modellrechner (SoM): Exemplarisch wird verdeutlicht, wie die Befehle und Strukturen einer Hochsprache auf die einfacheren Befehle eines „Prozessors“ übersetzt werden können. Zu Netzwerken bietet sich zunächst eine Betrachtung des schulinternen pädagogischen Netzes an (Linux-Server, bereitgestellte Dienste). Zum Thema Netzwerke siehe sonst: Schriek III, Kapitel 2. Exemplarisch könnten auch einzelne Protokolle eines Clients implementiert werden (je nach Zeit). Dem Client-Server-Modell begegnen die Schülerinnen und Schüler wieder beim Unterrichtsvorhaben Q2-2 (Datenbanken und Datensicherheit).