Lernfeld / Modul	Lernziele: Die SuS	Methodische Hinweise / Material	Ca. Zeit in DS
Kryptologie	• beschreiben das Prinzip der polyalphabetischen Substitution, u. a. am Beispiel des Vigenère-Verfahrens. • beurteilen die Sicherheit eines gegebenen symmetrischen Verschlüsselungsverfahrens. • beschreiben und unterscheiden die Prinzipien der symmetrischen und asymmetrischen Verschlüsselung. • beschreiben Anwendungsbereiche für symmetrische bzw. asymmetrische Verschlüsselungsverfahren. • erläutern das Prinzip von digitalen Signaturen und Zertifikaten. • eA: entwerfen und implementieren ein symmetrisches Verschlüsselungsverfahren. • eA: erläutern die prinzipielle Funktionsweise eines modernen symmetrischen Blockchiffreverfahrens.	• Crypttool	gA: 8 eA: 13
Algorithmik	• analysieren die Funktionsweise eines gegebenen Algorithmus ○ Tracetabellen • stellen Algorithmen in schriftlich verbalisierter Form dar ○ Struktogramme • eA: beurteilen die Effizienz von Algorithmen unter Abschätzung des Speicherbedarfs und der Zahl der Operationen. • verwenden geeignete Variablentypen zur Speicherung von Werten.	• Nutzung von Sachverhalten aus der Einführungsphase (Kryptologie?!) und diese implementieren	gA: 11 eA: 20

	• unterscheiden zwischen lokalen und globalen Variablen. • unterscheiden zwischen primitiven Datentypen und Objektreferenzen.		
	• verwenden Übergabeparameter und Rückgabewerte in Operationen. • eA: erläutern das Konzept der Rekursion an gegebenen Beispielen. • eA: entwerfen und implementieren rekursive Algorithmen. • eA: erläutern die Strategie „Teile und herrsche“ beim Entwurf rekursiver Algorithmen		
Datenstrukturen	• erläutern das Prinzip, mehrere Daten des gleichen Typs in Reihungen zu verwalten, zu suchen und zu sortieren. • entwerfen und implementieren Algorithmen unter Verwendung von ein- und zweidimensionalen Reihungen. • erläutern das Prinzip der Datenstrukturen Stapel, Schlange und dynamische Reihung. • entwerfen und implementieren Algorithmen unter Verwendung der Datenstrukturen Stapel, Schlange und dynamische Reihung.	•	gA: 8 eA: 12
Ende Halbjahr 1			
Klassen und Objekte	• entwerfen und implementieren Algorithmen unter Verwendung von gegebenen und eigenen Klassen/Objekten. • eA: entwerfen Klassen und deren Beziehungen (Assoziation, Vererbung) und stellen diese durch Klassendiagramme dar	• Beispiele ○ eA: Modellieren eines Rollenspiels • Programme ○ Java-Editor (nur Windows) ○ NetBeans (für eA)	gA: 4 eA: 6

Datenbanken	• erläutern den Aufbau relationaler Datenbanken unter Verwendung der Begriffe Datensatz, Attribut, Primärschlüssel, Fremdschlüssel und Tabelle. • nennen Beispiele für Einfüge-, Änderungs- und Löschanomalien. • untersuchen ein gegebenes Datenbankschema auf Anomalien und Redundanzen. • formulieren einfache Abfragen und Verbundabfragen über mehrere Tabellen. • formulieren Abfragen an Datenbanken unter Verwendung von Aggregatfunktionen. • eA: interpretieren ein gegebenes ER-Dia- gramm. • eA: modellieren Datenbanken unter Verwendung des ER-Modells. • eA: setzen ein ER-Modell in ein relationales Schema um. • eA: beurteilen und verändern eine gegebene Datenbankmodellierung.	• SQL Island • XERDI •	gA: 9 eA: 12
Projektarbeit: Entwicklung einer GUI-basierten Anwendungssoftware mithilfe von Java oder Python in Kleingruppen; Nutzung von Modellierungs- und Dokumentationstechniken (UML, Javadoc) bzw. Einbeziehung von Datenbank-Operationen			Ca. 5 Wochen
Datenschutz	• diskutieren die Chancen und Risiken der automatisierten Datenanalyse.		1-2

Ende Halbjahr 2 (Ende Jahrgang 12)			

Automatentheorie (Voraussetzung: Klassen und Objekte) (wichtig: mindestens einmal implementieren, als Programmieraufgabe)	• DEA einführen
	  ○ Zustandsgraphen erstellen
	  ○ Zustandsgraphen hinsichtlich der Funktion analysieren
		• Beispiele:
		  ○ Kaffeeautomat, Aufbau Email-Adresse
		• Implementierung
		  ○ Überprüfung / Formular Email-Adresse (Übergriff Algorithmik)
		• Programme
		  ○ AtoCC
			4

	• Mealy-Automaten (als Erweiterung der DEA)
	  ○ Zustandsgraphen erstellen
	  ○ Zustandsgraphen hinsichtlich der Funktion analysieren
		• Beispiele
		  ○ Kaffeeautomat (gr. / kl. Kaffee)
		• Implementierung mit 2D-Array
		• Programme
		  ○ AtoCC
			3

	• Grenzen endlicher Automaten
	  ○ gA: optionale Erweiterung möglich
	  ○ eA: Kellerautomaten (Sinn & Abgrenzung)
	• beschreiben den Aufbau und die Funktionsweise eines deterministischen endlichen Automaten (DEA).
	• beschreiben den Aufbau und die Funktionsweise eines endlichen Automaten mit Ausgabe (Mealy-Automat).
	• entwickeln und implementieren Automatenmodelle in Form von Zustandsgraphen.
	• analysieren die Funktion eines durch einen Zustandsgraphen vorgegebenen Automaten.
	• erläutern die Grenzen endlicher Automaten bei der Problemlösung.
	• eA: beschreiben den Aufbau und die Funktionsweise eines Kellerautomaten als
		• Beispiele:
		  ○ Kellerautomaten zur Überprüfung von geschachtelten Ausdrücken (Überleitung zu formalen Sprachen)
		  ○ eA: Palindrome
		• Implementierung
		  ○ offen
		• Programme
		  ○ AtoCC
			gA: 1
			eA: 3

	Erweiterung des Modells des endlichen Automaten.		
Formale Sprachen (nur eA) (Voraussetzung: Automatentheorie)	• eA: nennen Eigenschaften formaler Sprachen im Vergleich zu natürlichen Sprachen. • eA: beschreiben die von einer Grammatik erzeugte Sprache. • eA: entwerfen reguläre und kontextfreie Grammatiken für formale Sprachen. • eA: erläutern den Zusammenhang zwischen regulären Grammatiken und endlichen Automaten		gA: - eA: 6
Codierung & Datenübertragung	• beschreiben Möglichkeiten, Daten zu komprimieren, u. a. Lauflängencodierung, Huffman-Codierung. • eA: entwerfen und implementieren ein Kompressionsverfahren zu einem gegebenen Sachverhalt. • eA: erläutern die Vor- und Nachteile verlustfreier und verlustbehafteter Kompression von Daten. • eA: erläutern Möglichkeiten der Fehlererkennung und der Fehlerkorrektur bei der Datenübertragung, u. a. Paritäts- bit, (7,4) Hamming-Code.	• Zweite **Projektarbeit** im Bereich Mikrocontroller-Programmierung mit Arduino: Idealerweise unter Nutzung selbst entwickelter Protokolle (c**a. 3 Wochen**)	gA: 7 eA: 11
	• entwerfen und implementieren ein Protokoll zur Übertragung von Daten über einen Kommunikationskanal		
Mögliche vertiefende Inhalte, wahlweise:			
Technische Informatik	• Logische Schaltungen (Logische Gatter, Aaddierer, Inkrementierer, FlipFlops)	NAND-Game	
Theoretische Informatik	• Boolsche Ausdrücke & Algebra • Normalformen • Vereinfachung mit KV-Diagrammen		