Lehrplan Informatik Gymnasium Norwestösterreich

Jahrgangsstufe 9

Lernbereich 1: Funktionen und Datenflüsse, Tabellenkalkulationsprogramm (ca. 18 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- beherrschen den grundlegenden Umgang mit einem Tabellenkalkulationsprogramm (u. a. durch den adäquaten Einsatz von Formeln unter Nutzung sinnvoller Adressierung und passender Gestaltung des Layouts) und bewältigen damit geeignete Problemstellungen (z. B. die Auswertung von Daten oder einfache kaufmännische Berechnungen). Dies fördert das Verständnis, in welchem Kontext Tabellenkalkulationsprogramme nutzbringend eingesetzt werden können.
- analysieren datenverarbeitende Prozesse hinsichtlich der zu durchlaufenden Verarbeitungsvorgänge und abstrahieren insbesondere Prozesse mit mehreren Eingaben und einer Ausgabe als Funktionen. Dadurch vertiefen sie ihr Bewusstsein, dass man häufig durch Berechnungen nach eindeutigen Vorschriften aus vorhandenen Daten neue Informationen gewinnen kann.
- entwickeln neue Funktionen durch Verkettung gegebener Funktionen. Sie wenden damit ein grundlegendes Konzept der funktionalen Modellierung an.
- visualisieren die durch Funktionen ausgelösten Datenflüsse mithilfe von Datenflussdiagrammen.
- setzen Funktionen und Datenflussdiagramme zur effizienten Verarbeitung in Formeln eines Tabellenkalkulationssystems um und können damit durch geeignete Eingaben ihre Lösungsansätze konkret überprüfen.
- lösen praxisnahe Aufgabenstellungen, beispielsweise aus dem kaufmännischen Bereich oder der Mathematik, sachgerecht durch Anwendung der funktionalen Sichtweise, realisieren ihre Lösung mit einem Tabellenkalkulationssystem und überprüfen die Qualität der Lösung anhand verschiedener Eingaben.

Inhalte zu den Kompetenzen:

- Tabellenkalkulationsprogramm: Tabellenblatt, Zelle, Formel, Funktion (auch vordefinierte Funktion), relative und absolute Adressierung, elementare Datentypen (Zahl, Text, Wahrheitswert), Repräsentation von Daten durch Diagramme
- Funktion: Interpretation als Daten verarbeitender Prozess, vordefinierte Funktionen (u. a. bedingte und logische Funktion), Verkettung von Funktionen
- Datenflussdiagramm: Repräsentation einer Funktion, Datenfluss, Ein- und Ausgabe, Verteiler
- Fachbegriffe: Formel, Adressierung (relativ, absolut), Funktion (vordefiniert, bedingt, logisch, verkettet), Datentyp, Datenflussdiagramm, Verteiler

Lernbereich 2: Datenmodellierung und relationale Datenbanksysteme (ca. 26 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- analysieren und strukturieren mithilfe objektorientierter Konzepte Datenbestände einfacher Beispiele aus der Praxis (z. B. Lagerbestand oder Kundenverwaltung) und stellen das daraus entwickelte Datenmodell graphisch dar.
- überführen objektorientierte Datenmodelle in entsprechende relationale Modelle und setzen diese in ein Datenbanksystem um.
- konzipieren geeignete SQL-Abfragen, um zielgerichtet Informationen aus einer relationalen Datenbank zu gewinnen.
- erkennen Redundanzen und Anomalien in einer relationalen Datenbank und beurteilen die dadurch entstehende Problematik im Hinblick auf die Konsistenz des Datenbestands.
- vergleichen, beispielsweise im Hinblick auf Abfragemöglichkeiten, Datenbanksysteme mit anderen Abspeicherungsvarianten (z. B. Tabellenkalkulationssystemen) und schärfen dadurch ihr Bewusstsein, wann der Einsatz einer Datenbank sinnvoll ist.

Inhalte zu den Kompetenzen:

- Objektorientiertes Datenmodell: Objekt, Klasse, Attribut, Beziehung, Kardinalität
- Relationales Modell: Tabellenschema, Datenbankschema, Primär- und Fremdschlüssel, Datentyp
- Relationales Datenbanksystem: Datenbank, Datenbankmanagementsystem, Tabelle, Datensatz
- Redundanz und Konsistenz von Datenbeständen, Anomalien
- Abfrage: Interpretation als Funktion, Ergebnistabelle als Ergebnis einer Abfrage, Abfrage über mehrere Tabellen durch Verknüpfungen
- Abfragesprache am Beispiel von SQL: select, from, where; Verknüpfung von Bedingungen; Abfrage über mehrere Tabellen
- Fachbegriffe: (relationales) Datenbanksystem, (relationale) Datenbank, Datenbankmanagementsystem, Tabellenschema, Datensatz, Abfrage, Datenbankschema, Primärschlüssel, Fremdschlüssel, Ergebnistabelle, Redundanz, Konsistenz, Anomalie

Lernbereich 3: Projekt: Entwicklung einer Datenbank (ca. 6 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- erstellen im Team einen einfachen Projektplan für die Entwicklung einer Datenbank zu einem praxisbezogenen Anwendungsbeispiel, z. B. einer Bibliotheksverwaltung.
- entwerfen zu einem festgelegten Szenario eine Datenbank und halten sich dabei an die für die Datenbankentwicklung typische Vorgehensweise (Analyse des Szenarios, Entwicklung eines objektorientierten Datenmodells, Überführung in ein relationales Modell, Umsetzung in der Datenbank, Füllen der Datenbank mit Daten).

- testen mithilfe von geeigneten SQL-Abfragen, ob die entwickelte Datenbank den Anforderungen genügt.

Inhalte zu den Kompetenzen:

- Grundlagen der Projektarbeit: Planung (u. a. einfacher Projektplan mit Zeitplanung und Aufgabenverteilung in Hinblick auf Parallelisierung der Arbeiten), Durchführung, Test, Dokumentation

Lernbereich 4: Datenschutz und Datensicherheit (ca. 6 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- bewerten die Forderungen des Datenschutzes bei großen Datenbeständen unter den Aspekten von Notwendigkeit und Problematik, beispielsweise bei der Fahndung nach Straftätern.
- nutzen das Internet verantwortungsvoll unter Berücksichtigung ihrer Kenntnisse über Möglichkeiten und Risiken dieses Mediums und reflektieren dabei, wodurch der Schutz persönlicher Daten erhöht und die Gefahr des Missbrauchs minimiert werden kann. Insbesondere wägen sie kriteriengeleitet ihren Umgang mit datenbankgestützten Portalen ab.
- treffen geeignete Maßnahmen zum Schutz ihres Datenbestands vor fremden Zugriffen sowie vor Datenverlust.

Inhalte zu den Kompetenzen:

- Datenschutzgesetze: Zweck, Grundsätze (z. B. Verbotsprinzip mit Erlaubnisvorbehalt), Rechte von Betroffenen
- Datenschutz: Schutz personenbezogener Daten (insbesondere im Kontext der Mehrbenutzerproblematik bei Datenbanken), Datenmissbrauch (z. B. Identitätsdiebstahl)
- Datensicherheit: Schutz vor Datenverlust (z. B. durch einfache Backupstrategien), Benutzerkonten, Passwortschutz
- Fachbegriffe: Datensicherheit, Datenschutz

Lehrplan Informatik Gymnasium Bayern 6.- 8. Klasse

Jahrgangsstufe 10

Lernbereich 1: Objekt und Klasse, Algorithmik (ca. 26 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- analysieren Objekte aus ihrer Erfahrungswelt (z. B. Fahrzeuge, Personen) hinsichtlich ihrer Eigenschaften (Attribute) und Fähigkeiten (Methoden) und abstrahieren sie zu Klassen. Sie stellen Objekte und Klassen graphisch dar, um anschließend die Klassen mit einer objektorientierten Sprache implementieren zu können.
- implementieren den Rahmen einer Klasse und deklarieren die zugehörigen Attribute.
- verwenden an geeigneter Stelle bei der Implementierung Wertzuweisungen, um Attributwerte ändern zu können und interpretieren diese als Zustandsänderung des zugehörigen Objekts. Dabei gewährleisten sie den kontrollierten Zugriff auf Attribute und verstehen dadurch die Grundidee der Datenkapselung.
- deklarieren in den objektorientierten Modellen bzw. Programmen Methoden mit sinnvollen Signaturen.
- formulieren zu überschaubaren ablauforientierten Problemstellungen (z. B. Ausgabe beim Geldautomaten) unter Verwendung der Kontrollstrukturen geeignete Algorithmen und notieren diese fachgerecht mithilfe von Pseudocode oder geeigneten graphischen Darstellungen.
- implementieren Methoden auf der Grundlage gegebener Algorithmen objektorientiert, wobei sie sich des Unterschiedes zwischen Methodendefinition und Methodenaufruf bewusst sind.
- analysieren, interpretieren und modifizieren Algorithmen, wodurch sie die Fähigkeit erlangen, fremde Programme flexibel und kritisch zu beurteilen und zu bewerten.
- wenden den Datentyp Feld bei einfachen Problemstellungen fachgerecht zur Speicherung und Verwaltung gleichartiger Daten an.

Inhalte zu den Kompetenzen:

- Objektorientierte Konzepte: u. a. Objekt, Klasse, Attribut, Attributwert, Methode
- Variablenkonzept: Übergabeparameter und lokale Variable bei der Definition von Methoden, Deklaration von Attributen
- Wertzuweisung als Möglichkeit des direkten Zugriffs zur Änderung von Variablenwerten
- Datenkapselung: Grundidee am Beispiel des kontrollierten Zugriffs auf Attribute
- Standardmethoden zum Geben und Setzen von Attributwerten, Signatur einer Methode, Methodenkopf, Methodenrumpf, Methodendefinition versus Methodenaufruf, Konstruktor als spezielle Methode
- Algorithmus: Repräsentation durch Pseudocode und graphische Darstellung
- Feld über einfache, gleichartige Datentypen als Beispiel für einen zusammengesetzten Datentyp: Index, Feldelement, Feldlänge

- Datentypen: ganze Zahlen, Fließkommazahlen, Wahrheitswerte, Zeichen, Zeichenketten, Felder
- Fachbegriffe: Übergabeparameter, lokale Variable, Wertzuweisung, Konstruktor, Signatur, Methodenkopf, Methodenrumpf, Feld, Index, Feldelement, Feldlänge, Datenkapselung

Lernbereich 2: Zustandsdiagramme (ca. 6 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- analysieren Szenarien aus ihrer Erfahrungswelt, beispielsweise den Ablauf bei Ampelschaltungen oder die Bedienung von Audiogeräten bzw. Getränkeautomaten, und erkennen dabei, dass sich zeitliche Abläufe übersichtlich durch Zustände und Übergänge zwischen Zuständen darstellen lassen.
- modellieren in geeigneter Weise mithilfe von Zustandsdiagrammen Vorgänge, die durch Zustände und Zustandsübergänge beschrieben werden können.
- setzen Zustandsdiagramme in objektorientierte Programme sachgerecht um und können damit die Stimmigkeit ihres Modells in einer konkreten Simulation testen.

Inhalte zu den Kompetenzen:

- Zustand eines Objekts, festgelegt durch die zugehörige Attributwertmenge; Zustandsübergang als Änderung mindestens eines Attributwerts
- Zustandsdiagramm (Zustandsübergangsdiagramm): Zustand, Zustandsübergang, auslösende Aktion (Ereignis), Bedingung, ausgelöste Aktion
- Fachbegriffe: Zustandsdiagramm (Zustandsübergangsdiagramm), Zustand, Zustandsübergang, auslösende Aktion (Ereignis), ausgelöste Aktion

Lernbereich 3: Beziehungen zwischen Objekten (ca. 14 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- analysieren und modellieren das Kommunikationsverhalten innerhalb eines Systems (beispielsweise einer Autovermietung) und erkennen, dass wesentliche Abläufe eines Systems auf der Kommunikation zwischen den beteiligten Objekten basieren.
- implementieren die im Klassendiagramm festgelegten Beziehungen sachgerecht durch entsprechende Referenzattribute, um während der Laufzeit des Programms die Kommunikation zwischen den entsprechenden Objekten durch den Aufruf geeigneter Methoden zu ermöglichen.
- modellieren und implementieren mithilfe eines Feldes im Rahmen eines einfachen Beispiels aus der Praxis (etwa der Mitarbeiterverwaltung eines Betriebs) eine Liste zur Verwaltung gleichartiger Objekte; hierbei setzen sie ausgewählte Methoden zur Verwaltung der Liste um, u. a. das Einfügen und Entfernen eines Elements am Anfang oder Ende der Liste.

Inhalte zu den Kompetenzen:

- Interpretation von Klassen als Datentypen
- Umsetzung von Beziehungen zwischen Klassen mit der Kardinalität 1:1 und 1:n, Referenzattribut zur Implementierung einer Beziehung, Referenz auf ein Objekt
- Kommunikation zwischen Objekten durch Methodenaufrufe
- Fachbegriffe: Referenz, Referenzattribut, Kardinalität von Beziehungen

Lernbereich 4: Projekt: Entwicklung eines objektorientierten Programms (ca. 10 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- erstellen im Team einen einfachen Projektplan für die Entwicklung eines überschaubaren objektorientierten Programms zu einem vorgegebenen praxisbezogenen Anwendungsbeispiel, z. B. zur Entwicklung eines einfachen Spiels oder einer einfachen Verkehrssimulation.
- modellieren geeignet das Szenario zum gegebenen Anwendungsbeispiel mithilfe eines Klassendiagramms und evtl. weiterer Diagramme. Dabei arbeiten sie ggf. mit vorgegebenen Modellen bzw. Modellausschnitten; insbesondere achten sie auf die Vereinbarung notwendiger Schnittstellen.
- implementieren fachgerecht, arbeitsteilig und auf Grundlage der vorhandenen Modellierung das objektorientierte Programm zum gegebenen Szenario.
- testen das entwickelte Programm hinsichtlich der Anforderungen des gegebenen Szenarios und dokumentieren das Ergebnis ihrer Projektarbeit auf für Dritte nachvollziehbare Weise.

Inhalte zu den Kompetenzen:

- Grundlagen eines Softwareentwicklungsprojekts: Analyse, Modellierung, Implementierung, Test, Dokumentation
- Schnittstelle als wichtige Voraussetzung arbeitsteiliger Softwareentwicklung
- Fachbegriff: Schnittstelle

Jahrgangsstufe 11

Lernbereich 1: Generalisierung (ca. 8 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- analysieren und ordnen zweckmäßig hierarchische Strukturen aus ihrer Erfahrungswelt (z. B. Klassifizierung von Tieren) und erstellen entsprechende Generalisierungshierarchien in Form von Klassenmodellen.
- implementieren mithilfe einer objektorientierten Sprache Generalisierungshierarchien unter Berücksichtigung von Vererbung; dabei verwenden sie auch abstrakte Klassen.
- nutzen zur flexiblen Anpassung verschiedener Verhaltensweisen an den jeweiligen Kontext der Anwendungssituation (z. B. bei der rollenabhängigen Berechnung des Gehalts der

Mitarbeiter in einem Unternehmen) zielführend das Konzept der Polymorphie durch Überschreiben von Methoden in Unterklassen.

Inhalte zu den Kompetenzen:

- Generalisierungshierarchie: Ober- und Unterklasse, graphische Darstellung der hierarchischen Klassenstruktur
- Generalisierung und Spezialisierung als unterschiedliche Sichtweisen auf dieselbe Klassenbeziehung, Vererbung von Attributen und Methoden auf Unterklassen
- Abstrakte Klasse: Definition und grundlegende Konzeption, abstrakte Methode
- Polymorphismus und Überschreiben von Methoden
- Fachbegriffe: Vererbung, Generalisierung, Spezialisierung, Polymorphismus, Oberklasse, Unterklasse, abstrakte Klasse, abstrakte Methode

Lernbereich 2: Die rekursive Datenstruktur Liste (ca. 27 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- modellieren mithilfe einfach verketteter Listen lineare Datenstrukturen aus verschiedenen Situationen ihres Lebensumfeldes (z. B. Warteschlangen, Listen mit Personendaten). Sie nutzen dabei das Softwaremuster Kompositum und erkennen so den Vorteil einer bewährten Modellierungsstrategie.
- entwickeln unter Verwendung des Kompositums Algorithmen für die einfach verkettete Liste, um Elemente hinzuzufügen, zu löschen bzw. Berechnungen über die Listenelemente durchzuführen. Sie nutzen dabei das Prinzip der Rekursion als eine naheliegende Problemlösungsstrategie.
- implementieren fachgerecht einfach verkettete Listen und die zugehörigen Algorithmen mithilfe einer objektorientierten Programmiersprache.
- bewerten und vergleichen in konkreten Anwendungssituationen dynamische Listenstrukturen mit der statischen Struktur Feld und schärfen damit ihr Bewusstsein für einen zielgerichteten Einsatz der Datenstrukturen.
- nutzen bei der Bearbeitung von Anwendungssituationen aus der Praxis durch fachgerechte Anpassung an die konkrete Aufgabenstellung die durch Trennung von Struktur und Inhalt bedingte universelle Einsetzbarkeit verketteter Listen.

Inhalte zu den Kompetenzen:

- Liste als dynamische Datenstruktur zur Verwaltung von Datenbeständen mit flexibler Anzahl von Elementen versus Feld als statische Datenstruktur.
- Rekursion, rekursive Abläufe: rekursiver Aufruf, Abbruchbedingung, Aufrufsequenz
- einfach verkettete Liste: allgemeines Prinzip, rekursive Struktur, ausgewählte und soweit möglich rekursiv definierte Methoden (u. a. zum Einfügen, Entfernen und Suchen von Elementen sowie zur Bestimmung der Listenlänge)
- Trennung von Struktur und Daten/Inhalt
- Kompositum (Composite Pattern) als Beispiel eines Softwaremusters

- Grundprinzip von Stapel (LIFO) und Warteschlange (FIFO) als Spezialfälle der verketteten Liste
- Fachbegriffe: statische Datenstruktur, dynamische Datenstruktur, (einfach verkettete) Liste, Rekursion, rekursive Methode, rekursiver Aufruf, Abbruchbedingung, Aufrufsequenz, Kompositum, LIFO, FIFO, Softwaremuster

Lernbereich 3: Die rekursive Datenstruktur Baum (ca. 15 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- modellieren unter Berücksichtigung des Softwaremusters Kompositum und des Prinzips der Trennung von Struktur und Daten geordnete Binärbäume zu verschiedenen Problemstellungen ihres Erfahrungsbereiches (z. B. digitales Wörterbuch), in denen eine effiziente Datenhaltung wichtig ist. Durch den erneuten Einsatz des Kompositums erkennen sie die universelle Verwendbarkeit von Softwaremustern.
- entwickeln rekursive Algorithmen zur Verwaltung der Daten, die in einem Binärbaum abgespeichert sind (insbesondere zur Traversierung eines Binärbaums sowie zum Einfügen und Suchen von Elementen in einem geordneten Binärbaum), und wenden diese Algorithmen an konkreten Beispielen an.
- implementieren fachgerecht auf Grundlage gegebener Modelle geordnete Binärbäume mithilfe einer objektorientierten Programmiersprache.
- bewerten und vergleichen geordnete Binärbäume mit verketteten Listen hinsichtlich der Effizienz bei Suchanfragen. Ihnen wird damit bewusst, dass insbesondere ein ausbalancierter geordneter Binärbaum eine in Hinblick auf die Suche sehr effiziente Datenstruktur ist.
- nutzen bei der Bearbeitung von verschiedenen Anwendungssituationen aus der Praxis (z. B. Speicherung unterschiedlicher Daten wie Lexikoneinträge oder Kundeninformationen in Binärbäumen) eine bereits implementierte Version eines geordneten Binärbaums und passen diese fachgerecht an die konkrete Aufgabenstellung an. Sie vertiefen dabei ihr Verständnis, dass insbesondere durch das Konzept der Trennung von Struktur und Daten grundsätzlich eine Wiederverwendbarkeit der bereits vorliegenden Implementierung möglich ist.

Inhalte zu den Kompetenzen:

- Baum: Wurzel, Knoten, Kante, Blatt, Pfad, Höhe, Ebene; Binärbaum, Eigenschaften von Binärbäumen: vollständig, balanciert, entartet
- geordneter Binärbaum: Grundkonzept, Einfügen und Suchen von Elementen
- Traversierungsstrategien, d. h. Verfahren zur Auflistung aller Elemente eines Binärbaums: Präorder, Inorder, Postorder
- Fachbegriffe: Höhe, Ebene, Binärbaum (vollständig, balanciert, entartet, geordnet), Traversierung, Präorder, Inorder, Postorder

Lernbereich 4: Die Datenstruktur Graph (ca. 13 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- modellieren sachgerecht vernetzte Strukturen (Graphen) im Rahmen praktischer Fragestellungen, z. B. zur Planung von Verkehrsrouten. Dadurch gewinnen sie einen nachhaltigen Einblick in die umfassende Rolle, die Graphen in vielen Bereichen des Alltags spielen.
- klassifizieren Graphen allgemein und an konkreten Beispielen anhand ihrer Eigenschaften.
- implementieren mithilfe einer objektorientierten Programmiersprache und unter Verwendung einer Adjazenzmatrix auf fachgerechte Weise die Datenstruktur Graph.
- erläutern allgemein und an konkreten Beispielen die Idee der Tiefensuche, formulieren den zugehörigen Algorithmus und wenden diesen an konkreten Beispielen an.
- beurteilen die Einsetzbarkeit der Tiefensuche hinsichtlich vorgegebener Anforderungen (z. B. Erreichbarkeit sämtlicher Knoten in einem Graphen, kürzester Weg zwischen zwei Knoten).
- implementieren die Tiefensuche und modifizieren den Algorithmus in geeigneter, vom Anwendungskontext abhängiger Weise (z. B. bei der Auswahl aller Knoten mit bestimmten Eigenschaften).

Inhalte zu den Kompetenzen:

- Eigenschaften von Graphen: gerichtet, ungerichtet, zusammenhängend, unzusammenhängend, bewertet (gewichtet), unbewertet, mit Zyklen, zyklenfrei, Erreichbarkeit von Knoten
- Adjazenzmatrix, zweidimensionales Feld
- Algorithmus zum Graphendurchlauf am Beispiel der Tiefensuche
- Fachbegriffe: gerichtet, ungerichtet, zusammenhängend, unzusammenhängend, bewertet (gewichtet), unbewertet, mit Zyklen, zyklenfrei, Erreichbarkeit (von Knoten), zweidimensionales Feld, Adjazenzmatrix, Tiefensuche

Lernbereich 5: Softwaretechnik - Praktische Softwareentwicklung (ca. 21 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- erläutern und strukturieren unter Verwendung des Wasserfallmodells den Ablauf eines Softwareentwicklungsprojekts und orientieren sich dabei an dessen typischen Phasen.
- planen, strukturieren und koordinieren die Durchführung eines Softwareprojekts zu einer umfangreichen Aufgabenstellung aus der Praxis (z. B. Software zur Inventarverwaltung oder für einen Online-Routenplaner), indem sie sich an einem etablierten Vorgehensmodell der Softwareentwicklung (z. B. Wasserfallmodell) orientieren. Sie erhalten so einen realistischen Einblick in eine bewährte Vorgehensweise bei der Durchführung komplexer Projekte, wie sie beispielsweise im Berufsalltag auftreten.
- führen das Softwareprojekt entsprechend ihrer Planung im Team durch und berücksichtigen dabei bewährte Softwarearchitekturen, z. B. Model-View-Controller (MVC). Sie setzen in diesem Zusammenhang geeignete Modellierungstechniken der Informatik (z. B. Klassen-, Zustandsdiagramme) situationsgerecht ein und implementieren den Systementwurf, ggf. unter Nutzung passender rekursiver dynamischer Datenstrukturen und geeigneter Programmbibliotheken.

- prüfen und bewerten im laufenden Entwicklungsprozess mithilfe von praktischen Tests zur frühzeitigen Fehlererkennung die Richtigkeit der Softwarekomponenten hinsichtlich der in der Planung erstellten Spezifikation.
- erstellen eine fachgerechte Dokumentation des Softwareprojekts und präsentieren die Ergebnisse der Projektarbeit in geeigneter Weise.

Inhalte zu den Kompetenzen:

- Grundlagen der Projektplanung: Zielsetzung, Arbeitsteilung, Schnittstellen, Meilensteine, Lasten- und Pflichtenheft
- Wasserfallmodell als klassisches Beispiel eines Vorgehensmodells in der Softwareentwicklung mit den typischen Phasen: Analyse, Entwurf, Implementierung, Test, Bewertung und Abnahme
- Grundkonzept des Softwaremusters Model-View-Controller (MVC)
- Verwendung von Frameworks oder Bibliotheken, z. B. zur Nutzung einer Datenbank oder von Dateien zur persistenten Datenspeicherung
- Fachbegriffe: Vorgehensmodell, Wasserfallmodell, Phasen, Meilenstein, Lastenheft, Pflichtenheft, Softwaremuster Model-View-Controller (MVC)

Jahrgangsstufe 12

Inf12 Lernbereich 1: Formale Sprachen und Endliche Automaten (ca. 16 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- entwickeln formale Sprachen zu Beispielen aus dem Alltag (z. B. Autokennzeichen, E-Mail-Adressen oder Gleitkommazahlen), um ein Verständnis für die Notwendigkeit von klaren Regeln bei der Definition dieser Sprachen zu gewinnen und damit Mehrdeutigkeiten, wie sie in natürlichen Sprachen vorkommen, zu vermeiden.
- definieren formale Sprachen durch Grammatiken und verwenden zur Darstellung der Produktionsregeln insbesondere die Erweiterte Backus-Naur-Form (EBNF) und Syntaxdiagramme.
- entwerfen zur formalen Beschreibung von regulären Sprachen endliche erkennende Automaten.
- implementieren mithilfe einer objektorientierten Programmiersprache fachgerecht deterministische endliche Automaten und nutzen diese zur automatisierten Überprüfung der Zugehörigkeit von Wörtern zu einer regulären Sprache.
- erläutern an selbst gewählten Beispielen, dass es Sprachen gibt, die nicht regulär sind, und erkennen daran, dass es weitere Sprachkategorien in der Informatik gibt. Damit wird den Schülerinnen und Schülern bewusst, dass für die automatisierte Verarbeitung von nicht regulären Sprachen, wie z. B. höheren Programmiersprachen, das Modell des endlichen Automaten nicht ausreicht und weitere Modellkonzepte notwendig sind.

Inhalte zu den Kompetenzen:

- Formale Sprache als Menge von Zeichenketten über einem Alphabet: Zeichen, Zeichenvorrat (Alphabet), Wort (Zeichenkette), Syntax, Semantik
- Grammatik: Terminal, Nichtterminal, Produktionsregel, Startsymbol
- Notation formaler Sprachen: Syntaxdiagramm und erweiterte Backus-Naur-Form (EBNF)
- Ableitung eines Wortes einer formalen Sprache als Folge von Regelanwendungen, Ableitungsbaum
- erkennender endlicher Automat: Zustandsmenge, Eingabealphabet, Zustandsübergang, Startzustand, Endzustand, Fangzustand (Fehlerzustand); reguläre Sprache
- Fachbegriffe: formale Sprache, Alphabet, Grammatik, Terminal, Nichtterminal, Produktionsregel, Startsymbol, Syntaxdiagramm, reguläre Sprache, Ableitung, Ableitungsbaum, erweiterte Backus-Naur-Form (EBNF), erkennender endlicher Automat, deterministischer endlicher Automat, Eingabealphabet, Startzustand, Endzustand, Fangzustand (Fehlerzustand), Zustandsübergang, Syntax, Semantik

Lernbereich 2: Kommunikation von Prozessen (ca. 7 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- strukturieren Kommunikationsvorgänge durch Aufteilung in geeignete, aufeinander aufbauende Schichten und erhalten so auch ein grundlegendes Verständnis für die Bedeutung von Protokollen bei der Prozesskommunikation.
- sind aufgrund ihrer Kenntnisse der wesentlichen Prinzipien elektronischer Kommunikation in Netzwerken in der Lage, einfache Fehleranalysen bei Kommunikationsstörungen in Netzwerken (z. B. Nichterreichbarkeit eines Servers aufgrund falscher Adressierung) durchzuführen.

Inhalte zu den Kompetenzen:

- Kommunikation zwischen Prozessen, Protokolle zur Beschreibung dieser Kommunikation, Schichtenmodell
- Rechnernetz, Client-Server-Modell, Adressierung (MAC-Adresse, IP-Adresse, Port)
- Fachbegriffe: Prozess, Protokoll, Schichtenmodell, Client-Server-Modell, MAC-Adresse, IP-Adresse, Port

Lernbereich 3: Modellierung nebenläufiger Prozesse (ca. 13 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- analysieren und bewerten nebenläufige Prozesse im Hinblick auf ihren grundsätzlichen Ablauf und auf die Nutzung gemeinsamer Ressourcen; sie erkennen dabei insbesondere, dass der gemeinsame Zugriff auf Ressourcen durch Synchronisation geregelt werden muss.
- untersuchen anhand entsprechender Petrinetze Modelle nebenläufiger Prozesse hinsichtlich Konflikt- und Verklemmungssituationen und lösen diese in einfachen Fällen durch geeignete Modifikationen.

- modellieren mithilfe von Petrinetzen typische nebenläufige Szenarien (z. B. Geschäftsabläufe, Lagerverwaltung mit mehreren Lieferanten, Steuerung von Roboteranlagen, Verkehrsregelung an Straßenkreuzungen). Dadurch gewinnen sie ein vertieftes Verständnis dafür, dass derartige reale Vorgänge - beispielsweise aus Effizienzgründen - möglichst in parallel ablaufende Prozesse zerlegt werden, die aber insbesondere bei Nutzung gemeinsamer Ressourcen koordiniert werden müssen.

Inhalte zu den Kompetenzen:

- Prozess, Nebenläufigkeit, Synchronisation, Verklemmung
- Petrinetz (Stellen-Transitions-Netz) zur Analyse und Modellierung einfacher nebenläufiger Systeme bzw. Vorgänge: Stelle, Transition, Marke, Kapazität
- Klassische Probleme der Prozesssynchronisation: Erzeuger-Verbraucher-Problem, Leser-Schreiber-Problem
- Fachbegriffe: nebenläufige Prozesse, Synchronisation, Verklemmung, Erzeuger-Verbraucher-Problem, Leser-Schreiber-Problem, Petrinetz (Stellen-Transitions-Netz), Stelle, Transition, Marke, Kapazität

Lernbereich 4: Funktionsweise eines Rechners (ca. 17 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- beschreiben, orientiert am Von-Neumann-Modell, den schematischen Aufbau sowie die grundsätzliche Funktionsweise eines Computersystems und erläutern die Bedeutung des Von-Neumann-Modells in diesem Kontext.
- stellen natürliche Zahlen im Binär- und Hexadezimalsystem dar und führen entsprechende Umrechnungen durch. Sie bauen damit ein Grundverständnis auf, wie Informationen in einem Speicher abgelegt und wie verschiedene hardwarenahe Repräsentationen (z. B. die MAC-Adresse oder die RGB-Darstellung von Farben) interpretiert werden können.
- setzen mithilfe einer Registermaschinensimulation auf Assemblerebene einfache Algorithmen um, die grundlegende Kontrollstrukturen enthalten, und testen diese Programme. Sie erhalten so ein Verständnis dafür, wie Programme, die mit höheren Programmiersprachen verfasst sind, auf maschinennaher Ebene repräsentiert werden.
- erläutern, z. B. unter Betrachtung der aktuellen Speicherbelegung, die prinzipielle Abarbeitung von Programmen bei einer Registermaschine. Sie vertiefen dabei ihr Verständnis über die grundsätzliche Funktionsweise eines Rechners.

Inhalte zu den Kompetenzen:

- von Neumann-Modell als grundlegendes Modell für moderne Rechner: Prozessor (Rechenwerk, Steuerwerk), Speicher (u. a. Abgrenzung Arbeits- versus Permanentspeicher), Ein- und Ausgabeeinheit, Bussystem
- Binär- bzw. Hexadezimalsystem als Grundlage der Codierung von Information in einem Speicher: Bit, Byte, Binär- bzw. Hexadezimalcodierung natürlicher Zahlen, Stellenwertsystem
- einfaches Modell einer (auf der von-Neumann-Architektur basierenden) Registermaschine: Akkumulator, Befehlsregister, Befehlszähler, Statusregister, Befehlszyklus

- algorithmische Grundbausteine auf Assemblerebene: Sequenz, ein- und zweiseitig bedingte Anweisung, Wiederholungen
- Fachbegriffe: Binärsystem, Hexadezimalsystem, Bit, Byte, von-Neumann-Modell, Prozessor, Speicher, Ein- und Ausgabeeinheit, Bussystem, Registermaschine, Akkumulator, Befehlsregister, Befehlszähler, Statusregister, Befehlszyklus

Lernbereich 5: Praktische Grenzen der Berechenbarkeit (ca. 10 Std.)

Kompetenzerwartungen

Die Schülerinnen und Schüler ...

- bewerten durch einfache Abschätzungen mithilfe von Zeitmessungen und Zählverfahren (z. B. Zählen der Aufrufe bei rekursiven Algorithmen, Zählen der zeitkritischen Anweisungen) den Laufzeitaufwand überschaubarer Algorithmen. Dadurch wird deutlich, dass unterschiedliche Algorithmen zur Lösung eines Problems dieses unterschiedlich schnell lösen.
- begründen mithilfe geeigneter Beispiele (z. B. Brute-Force-Verfahren zur Entschlüsselung unbekannter Passwörter), dass die Sicherheit moderner Verschlüsselung auf den praktischen Grenzen der Berechenbarkeit beruht. Damit wird bei den Schülerinnen und Schülern das Bewusstsein geschärft, dass ein hoher Laufzeitaufwand ein zentrales Kriterium für den Schutz vor Entschlüsselung ist.

Inhalte zu den Kompetenzen:

- Laufzeitaufwand von Algorithmen (linear, exponentiell, quadratisch als Beispiel für polynomiales Laufzeitverhalten, logarithmisch), Best-Case, Average-Case, Worst-Case
- Brute-Force-Verfahren
- Fachbegriffe: Laufzeitverhalten, Brute-Force-Verfahren, Best-Case, Average-Case, Worst-Case