Mündlichen Abiturprüfung – Informatik

Thema: OOP, Algorithmen, Daten Banken, Formale Sprachen

30. April 2025

Aufgabe 1: Analyse eines Algorithmus

Ein unbekannter Algorithmus zur Berechnung einer speziellen mathematischen Funktion von Zahlen wird bereitgestellt. Der Name der Klasse lautet `Algorithmus`.

Gegebener Java-Code

```
public class Algorithmus {

 public static int berechne(int b, int e) {
  if (e == 0) {
   return 1;
  }
  return b * berechne(b, e - 1);
 }

 public static void main(String[] args) {
  int b = 3;
  int e = 3;
  System.out.println("Ergebnis: " + berechne(b, e));
 }
}
```

Teilaufgaben

1. Beschreiben Sie die Funktionsweise des Algorithmus. Welche mathematische Berechnung führt die Methode `berechne` der Klasse `Algorithmus` durch?

2. Welche Art der Implementierung wurde gewählt und welche Vor- oder Nachteile ergeben sich daraus?

3. Bestimmen Sie die Zeitkomplexität des Algorithmus.

4. Vergleichen Sie die rekursive Lösung mit einer iterativen Variante hinsichtlich Aufwand, Verständlichkeit und praktischer Anwendung.

Aufgabe 2: Datenmodellierung und Normalisierung

Im SchulPortal der Heinrich-von-Kleist-Schule werden Schüler- und Elterndaten in einer relationalen Datenbank gespeichert. Die ursprüngliche Tabellenstruktur sieht wie folgt aus:

Tabelle 1: Ursprüngliche nicht normalisierte Tabelle

SchuelerID	Vorname	Nachname	Adresse (Str., Stadt, PLZ)	Klasse	ElternID	Name_Elternteil	Telefonnummer_Eltern
1	Anna	Müller	Hauptstr. 12, Berlin, 10115	10A	E101	Peter Müller	0301234567
2	Lukas	Becker	Marktstr. 5, Hamburg, 20095	10B	E102	Sabine Becker	0407654321
3	Anna	Müller	Hauptstr. 12, Berlin, 10115	10A	E102	Sabine Becker	0407654321
4	Marie	Schmitt	Lindenallee 9, München, 80331	10C	E103	Julia Schmitt	0893344556

Teilaufgaben:

1. Identifizieren Sie Redundanzen und Anomalien in der Tabelle.
2. Zerlegen Sie die nicht normalisierte Tabelle schrittweise und normalisieren Sie sie bis zur 3. Normalform. Geben Sie die neu entstandenen Tabellen jeweils an.
3. Erläutern Sie, welche konkreten Vorteile die Normalisierung in diesem Fall für das SchulPortal bringt.

Aufgabe 3: Analyse einer formalen Grammatik für deutsche Sätze

Gegeben sei folgende formale Grammatik G:

- Nichtterminale: {`<satz>`, `<subjekt>`, `<prädikat>`, `<objekt>`, `<substantivgruppe>`, `<artikel>`, `<`
- Terminale: {`Friedrich`, `Roland`, `der`, `die`, `das`, `liest`, `kauft`, `lernt`, `Buch`, `Schokolade`, `Informati`
- Produktionsregeln:

$$
\begin{aligned}
\texttt{<satz>} &\rightarrow \texttt{<subjekt><prädikat><objekt>} \\
\texttt{<subjekt>} &\rightarrow \texttt{<eigenname>} \mid \texttt{<substantivgruppe>} \\
\texttt{<substantivgruppe>} &\rightarrow \texttt{<artikel><substantiv>} \\
\texttt{<prädikat>} &\rightarrow \texttt{<verb>} \\
\texttt{<objekt>} &\rightarrow \texttt{<substantivgruppe>} \\
\texttt{<verb>} &\rightarrow \texttt{liest} \mid \texttt{kauft} \mid \texttt{lernt} \\
\texttt{<artikel>} &\rightarrow \texttt{der} \mid \texttt{die} \mid \texttt{das} \\
\texttt{<eigenname>} &\rightarrow \texttt{Friedrich} \mid \texttt{Roland} \\
\texttt{<substantiv>} &\rightarrow \texttt{Buch} \mid \texttt{Schokolade} \mid \texttt{Informatik}
\end{aligned}
$$

- Startsymbol: `<satz>`

Teilaufgaben:

1. Leiten Sie die folgenden Sätze mit der Grammatik ab:
   - `Friedrich liest das Buch`
   - `der Friedrich der das Buch liest`
2. Die Erweiterung der Grammatik, um Sätze wie `"der Friedrich der die Schokolade kauft"` zu akzeptieren, wäre aus Sicht einer formalen Sprache fehlerhaft. Diskutieren Sie, in welchen Situationen es trotzdem sinnvoll sein könnte, solche Erweiterungen vorzunehmen, beispielsweise bei der Entwicklung von KI-gestützten Korrektur- oder Schreibprogrammen.

Lösungen

Lösung zu Aufgabe 1: Analyse eines Algorithmus

1. **Funktionsweise:** Der Algorithmus berechnet die Summe aller natürlichen Zahlen von n bis 1, also
$$1 + 2 + 3 + \cdots + n$$
Es handelt sich um die Berechnung der sogenannten *arithmetischen Reihe*.

2. **Art der Implementierung:** Die Implementierung ist **rekursiv**. Der Vorteil: der Code ist sehr kompakt und leicht verständlich. Der Nachteil: bei sehr großen n kann ein *Stack Overflow* auftreten, weil für jede Rekursionsstufe ein Funktionsaufruf auf dem Stack liegt.

3. **Funktionsweise bei $n = 5$:**
$$\begin{aligned}
\texttt{berechne(5)} &= 5 + \texttt{berechne(4)} \\
&= 5 + (4 + \texttt{berechne(3)}) \\
&= 5 + (4 + (3 + \texttt{berechne(2)})) \\
&= 5 + (4 + (3 + (2 + \texttt{berechne(1)}))) \\
&= 5 + (4 + (3 + (2 + (1 + \texttt{berechne(0)})))) \\
&= 5 + (4 + (3 + (2 + (1 + 0)))) \\
&= 5 + (4 + (3 + (2 + 1))) \\
&= 5 + (4 + (3 + 3)) \\
&= 5 + (4 + 6) \\
&= 5 + 10 \\
&= 15
\end{aligned}$$

4. **Anzahl der Aufrufe:** Für `berechne(5)` werden insgesamt 6 Aufrufe benötigt (für $n = 5, 4, 3, 2, 1, 0$).

5. **Zeitkomplexität:** Jeder Funktionsaufruf reduziert n um 1, also linear viele Aufrufe. $\rightarrow$ Zeitkomplexität: $\mathcal{O}(n)$

6. **Vergleich zu iterativer Berechnung:** Eine Schleife wie

```
int summe = 0;
for (int i = 1; i <= n; i++) {
 summe += i;
}
```

benötigt ebenfalls $\mathcal{O}(n)$ Zeit, aber keinen Rekursions-Stack und ist deshalb effizienter im Speicherverbrauch.

Lösung zu Aufgabe 2: Datenmodellierung und Normalisierung – SchulPortal

1. **Redundanzen und Anomalien:**

   - **Redundanz:**
     - Die Daten eines Elternteils (z. B. Name, Telefonnummer) erscheinen mehrfach, wenn ein Elternteil mehreren Schülern zugeordnet ist.
     - Die Adresse wird mehrfach gespeichert, obwohl sie logisch eine eigene Entität ist.
   - **Anomalien:**
     - *Änderungsanomalie:* Wenn sich die Telefonnummer eines Elternteils ändert, muss dies an mehreren Stellen manuell angepasst werden.
     - *Einfügeanomalie:* Ein neuer Elternteil kann nicht in das System aufgenommen werden, ohne einen verknüpften Schüler einzutragen.
     - *Löschanomalie:* Wird ein Schüler gelöscht, können auch wichtige Elterninformationen unbeabsichtigt verloren gehen.

2. **Schrittweise Normalisierung bis zur 3. Normalform (3NF):**

   Schritt 1: 1. Normalform (1NF)

   - Die Spalte `Adresse` enthält zusammengesetzte Werte (Straße, Stadt, PLZ) und ist daher **nicht atomar**.
   - Zur Erfüllung der 1NF müssen diese Bestandteile auf separate Attribute verteilt werden.
   - **Ergebnis:** Nach Aufspaltung in `Straße`, `Stadt`, `PLZ` erfüllt die Tabelle die 1NF.

   Schritt 2: 2. Normalform (2NF)

   - Die Ausgangstabelle enthält funktionale Abhängigkeiten, die nur von einem Teil des zusammengesetzten Primärschlüssels (SchuelerID, ElternID) abhängen.
   - Beispiel: Vorname, Nachname, Klasse hängen nur von der SchuelerID ab.
   - Zerlegung in separate Tabellen:
     - `Schueler(SchuelerID, Vorname, Nachname, AdresseID, Klasse)`
     - `Elternteil(ElternID, Name_Elternteil, Telefonnummer)`
   - Einführung einer Zuordnungstabelle für n:m-Beziehungen:
     - `SchuelerEltern(SchuelerID, ElternID)`

Schritt 3: 3. Normalform (3NF)

- Die Adresse besteht noch aus mehreren Attributen (Straße, Stadt, PLZ), die zusammengehören, aber nicht direkt funktional von SchuelerID abhängen.
- Diese transitive Abhängigkeit wird durch eine separate Adresse-Tabelle beseitigt:
  - `Adresse(AdresseID, Straße, Stadt, PLZ)`
- Die Adresse wird über einen Fremdschlüssel mit `Schueler` verknüpft:
  - `Schueler(SchuelerID, Vorname, Nachname, AdresseID (FK), Klasse)`

Endgültige Tabellenstruktur mit Primär- und Fremdschlüsseln:

- `Adresse(AdresseID, Straße, Stadt, PLZ)`
- `Schueler(SchuelerID, Vorname, Nachname, AdresseID (FK), Klasse)`
- `Elternteil(ElternID, Name_Elternteil, Telefonnummer)`
- `SchuelerEltern(SchuelerID (FK), ElternID (FK))`

(FK = Fremdschlüssel)

3. **Vorteile der Normalisierung:**
   - **Redundanzfreiheit:** Daten wie Adresse oder Telefonnummer müssen nur einmal gespeichert werden.
   - **Datenkonsistenz:** Änderungen an einer Stelle wirken sich automatisch korrekt aus.
   - **Bessere Wartbarkeit:** Klare Trennung von Entitäten vereinfacht Erweiterungen.
   - **Datenintegrität:** Fremdschlüsselbeziehungen sichern korrekte Verknüpfungen.
   - **Flexibilität:** Mehrere Schüler können denselben Elternteil oder dieselbe Adresse referenzieren.

Lösung zu Aufgabe 3: Analyse einer formalen Grammatik

1. **Ableitungen:**
   - `Friedrich liest das Buch:`

$$
\begin{aligned}
\texttt{<satz>} &\Rightarrow \texttt{<subjekt><prädikat><objekt>} \\
&\Rightarrow \texttt{<eigenname><prädikat><objekt>} \\
&\Rightarrow \texttt{Friedrich <verb> <objekt>} \\
&\Rightarrow \texttt{Friedrich liest <substantivgruppe>} \\
&\Rightarrow \texttt{Friedrich liest <artikel><substantiv>} \\
&\Rightarrow \texttt{Friedrich liest das Buch}
\end{aligned}
$$

- `der Friedrich der das Buch liest:`
  Keine korrekte Ableitung möglich: -Artikel „der“ kann nicht vor Eigenname „Friedrich“ stehen. -Struktur weicht ab: doppelter Artikel und falsche Reihenfolge.

2. **Sprachbeschreibung:**

   Die Grammatik erzeugt Sätze der Struktur:

   (Eigenname oder Artikel + Substantiv) + Verb + (Artikel + Substantiv)

   also einfache Sätze wie: - Friedrich liest das Buch - die Schokolade kauft der Roland

   (aber keine Relativsätze oder verschachtelten Konstruktionen.)

3. **Diskussion KI und Erweiterung:**

   Formale Grammatiken verlangen eine strikt festgelegte Satzstruktur. KI-gestützte Korrektur- oder Schreibprogramme müssen jedoch auch mit umgangssprachlichen, ungrammatischen oder verschachtelten Sätzen umgehen können. Daher ist es sinnvoll, die Grammatik zu erweitern oder zu lockern, um:

   - Eingabefehler besser zu erkennen und zu korrigieren
   - Benutzerfreundlichkeit zu erhöhen
   - realistische Alltagssprache zu verarbeiten