**Preparation Task "Algorithms on Graphs":** A company is hierarchically organized (in the form of a pyramid). That is, all employees, except for the CEO, are assigned to exactly one direct supervisor. The organizational structure of the company is described by the following two tables MA and ORG:

Table MA: Employee Master Data

|  |  |
| --- | --- |
| **EmployeeID** | **Name** |
| Primary Key |  |
| AH | Andrea Hansen |
| BK | Bernd Krüger |
| CD | Chris Dick |
| DX | Do Xin |
| EY | Eren Yildiz |

Table **ORG**: Structure Data

|  |  |
| --- | --- |
| **EmployeeID** | **SupervisorID** |
| Primary Key Foreign Key | Primary Key Foreign Key |
|  | Supervisor of the employee |
| AH | DX |
| BK | DX |
| CD | EY |
| DX | FZ |
| EY | FZ |

a) Illustrate the structural data from ORG graphically as a tree!

b) Describe an algorithm in pseudocode that outputs how many employees are directly subordinate to a given employee (EmployeeID) on the next hierarchy level. Use the pseudocode elements provided in the appendix.

c) Explain how one can determine the name of the company's CEO from the given tables without sketching the graph and determine the solution visually by inspection.

c1) first in natural language

c2) in pseudocode! Use the pseudocode elements provided in the appendix.

d) Describe an algorithm in pseudocode that outputs the hierarchy level of any given employee (given EmployeeID). This is determined by the path length from the employee to the top CEO, meaning that the CEO is positioned at hierarchy level 0, and the employees directly beneath them at level 1, and so on.

d1) Solve the problem without recursion.

d2) Solve the problem using recursion.

Note: The algorithms should be general enough to be applicable to hierarchical organizational structures of other companies as well!

# Pseudocode Bausteine

|  |  |
| --- | --- |
| **Bedeutung** | **Pseudocode** |
| Zuweisung | **:=** bzw **←** |
| Vergleich | **= ≠ < > ….** bzw **gleich** **ungleich** … |
| Logische Operatoren | ∧ ∨ ¬ **….** Bzw und and oder or nicht |
| Mengenoperatoren | ∪, füge ein in Menge, entferne aus Menge … |
| Bedingte Ausführung   * Unbedingt mit Einrückungen die Zuordnung verdeutlichen | If *bedingung*  then  *Ja - statement*  Else  *nein - statement*  endif  Falls *condition* dann  *Ja - statement*  andernfalls  *nein - statement*  EndeFalls  Case Ausdruck of  Wert1: stament1  …  Wertn: stamentn  default: stament  endcase |
| Generische Schleifen   * Unbedingt mit Einrückungen die Zuordnung verdeutlichen * Unbedingt mit end…Schlüssselwort das Ende der Schleife verdeutlichen | for  *wert:= 1*  to n do  *statements*  endfor  Schleife von  *wert:= 1* bis n :  *statements*  endeSchleife  ---  for all *a є A*  do  *statements*  endfor  ---  while bedingung do  *statements*  endwhile  ---  repeat  *statements*  until bedingung  wiederhole solange  *statements*  bis bedingung |
| Schleifen über Tabellen | Loop all records *rec*  of table …  *statements*  endloop  ---  Schleife über alle Datensätze  *rec*  von Tabelle …  *statements*  EndeSchleife |
| Filtern einer Tabelle | …. \_tmp ← filtere Tabelle …. mit Bedingung…  …. \_tmp ← filter Table …. with Bedingung… |
| Sortieren einer Tabelle | A\_sort ← Sortiere Tabelle A [aufsteigend] nach dem Feld ...  A\_sort ← Sortiere Tabelle A [absteigend] nach dem Feld ... |
| Lesen genau eines Datensatzes | Record ← lese Datensatz aus Tabelle …. mit Bedingung  Record ← read record in Table …. with Bedingung  Record ← lese ersten Datensatz aus Tabelle ….  Record ← lese nächsten Datensatz aus Tabelle …. |
| Erzeuge einen Datensatz oder modifizier einen Datensatz | A ← erzeuge record in Tabelle ...  A ← modifiziere record in Tabelle ... |
| Operationen auf Feldern einer Tabelle: Minimum, Maximum oder Summe | wert ← min ... des Feldes f von Tabelle ...  wert ← max ... des Feldes f von Tabelle ...  wert ← sum ... des Feldes f von Tabelle ... |
| Operationen auf Tabellen | wert ← Zähle Anzahl Datensätze der Tabelle …. |
| Operationen auf gerichteten Graphen | M ← alle Kindknoten von Knoten ….  M ← alle Elternknoten von Knoten …. |
| Allgemeine Operationen | Schreibe Output … Ausgabe …  Prüfe, ob letzte Operation erfolgreich war  Wert ← Inkrementiere Variable wert  varJahr ← bestimme Jahr aus Datum  varJahrMonat ← bestimme Jahr und Monat aus Datum  DateYFirst ← bestimme zu Jahr den ersten KalenderTag des Jahres  DateYLast ← bestimme zu Jahr den letzten KalenderTag des Jahres  DateYMFirst ← bestimme zu JahrMonat den ersten KalenderTag  DateYMLast ← bestimme zu JahrMonat den letzten KalenderTag |
| Einfügen in Fifo-Warteschlange | W ← Füge Objekt x in die FIFO-Warteschlange W hinzu |
| Entnehme aus Fifo-Warteschlange | X ← Entnehme Objekt aus der FIFO-Warteschlange W mit der längsten Wartezeit |
| Regel 1 | Wenn eine Anzahl von Datensätzen in einer Tabelle gesucht sind, die eine Bedingung erfüllen, soll diese Aufgabe vorzugsweise mit einer Filterung der Tabelle gelöst werden, indem nach der Filterung die Funktion „zähle Datensätze“ verwendet wird. |
| Regel 2 (erweitert Regel 1) | Verwende Schleifen in der Tabellenbearbeitung nur dann, wenn das unbedingt nötig ist, und bevorzuge die Funktionen auf Tabellen (zB zähle Datensätze) oder ihren Feldern (zB Summe der Spalte), ggf in Kombination mit der Filterung |
| Regel 3 | Wenn ein Wert um eins erhöht werden muss, ist der Befehl zum Inkrementieren zu verwenden. |

**Formatierung für eine KI-Verarbeitung:**

Pseudocode Patterns, die für den Algorithmen-Entwurf vorzugsweise verwendet werden sollten. Jede Gruppe von Operationen wir durch ein "\*"-Zeichen eingeleitet. Am Ende der Liste folgen noch Regeln, die zu beachten sind.

\* Zuweisung

Pseudocode: ← bzw =

\* Logischer Vergleich

Pseudocode: <, >, <=, >=, =, <> (bzw. "gleich", "ungleich")

\* Logische Operatoren

Pseudocode: ∧ (und), ∨ (oder), ¬ (nicht)

\* Mengenoperationen

Pseudocode: füge ein in Menge, entferne aus Menge

\* Bedingte Ausführung

Pseudocode:

If bedingung then

Ja - statement

else

Nein - statement

EndIf

oder:

Falls bedingung dann

Ja - statement

andernfalls

Nein - statement

EndFalls

\* Case-Ausdruck

Pseudocode:

Case Ausdruck

Wert1: statement1

Wert2: statement2

default: statementDefault

EndCase

\* Generische Schleifen

Pseudocode:

\*\* einfache Schleife

for wert := 1 to n do

statements

endfor

- oder:

Schleife von wert := 1 bis n

statements

endSchleife

\*\* Für "solange"-Schleifen:

while bedingung do

statements

endwhile

\*\* Für "repeat-until"-Schleifen:

repeat

statements

until bedingung

\* Schleifen über Tabellen

Pseudocode:

Loop all records rec of table

statements

endloop

\* Filtern einer Tabelle

Pseudocode:

A\_ tmp ← filtere Tabelle „A“ mit Bedingung ...

Sortieren einer Tabelle

Pseudocode:

A\_sort ← Sortiere Tabelle „A“ [aufsteigend] nach dem Feld ...

A\_sort ← Sortiere Tabelle „A“ [absteigend] nach dem Feld ...

\* Lesen genau eines Datensatzes

Pseudocode:

Record ← lese Datensatz aus Tabelle ... mit Bedingung ...

\* Erzeuge neuen Datensatz oder modifiziere bestehenden Datensatz in einer Tabelle

Pseudocode:

A ← erzeuge record in Tabelle ...

A ← modifiziere record in Tabelle ...

\* Operationen auf Feldern einer Tabelle: Minimum, Maximum oder Summe

Pseudocode:

wert ← min ... des Feldes f von Tabelle ...

wert ← max ... des Feldes f von Tabelle ...

wert ← sum ... des Feldes f von Tabelle ...

\* Operationen auf Tabellen

Pseudocode:

wert ← Zähle Anzahl Datensätze der Tabelle ….

\* Operationen auf Graphen

Pseudocode:

M ← alle Knoten von Knoten ...

\* Allgemeine Operationen

Pseudocode:

\*\* Schreibe Output ... Ausgabe ...

\*\* Prüfe, ob letzte Operation erfolgreich war (ergibt logischen Wert true/false)

\*\*Wert ← Inkrementiere Variable wert

\*\* varJahr ← bestimme Jahr aus Datum

\*\* varJahrMonat ← bestimme Jahr und Monat aus Datum

\*\* DateYFirst ← bestimme zu Jahr den ersten KalenderTag des Jahres

\*\* DateYLast ← bestimme zu Jahr den letzten KalenderTag des Jahres

\*\* DateYMFirst ← bestimme zu JahrMonat den ersten KalenderTag

\*\* DateYMLast ← bestimme zu JahrMonat den letzten KalenderTag

\* Operationen auf Fifo-Warteschlange

\*\* Einfügen in FIFO-Warteschlange  
Pseudocode:  
W ← Füge Objekt x in die FIFO-Warteschlange W

\*\* Entnehmen aus FIFO-Warteschlange  
Pseudocode:  
X ← Entnehme Objekt aus der FIFO-Warteschlange W mit der längsten Wartezeit

## Hinweis: Bei der Bearbeitung von Tabellen sollen folgende Regeln verwendet werden:

\*\* Regel 1: Wenn eine Anzahl von Datensätzen in einer Tabelle gesucht sind, die eine Bedingung erfüllen, soll diese Aufgabe vorzugsweise mit einer Filterung der Tabelle gelöst werden, in dem nach der Filterung die Funktion „zähle Datensätze“ verwendet wird.

\*\* Regel 2 (erweitert Regel 1): Verwende Schleifen in der Tabellenbearbeitung nur dann, wenn das unbedingt nötig ist, und bevorzuge die Funktionen auf Tabellen oder ihren Feldern.

\*\* Regel 3: Wenn ein Wert um eins erhöht werden muss, ist der Befehl zum Inkrementieren zu verwenden.