Elementare Algorithmen und Objektorientierte Programmierung

Übung 4

Auszuarbeiten bis 08.04.21

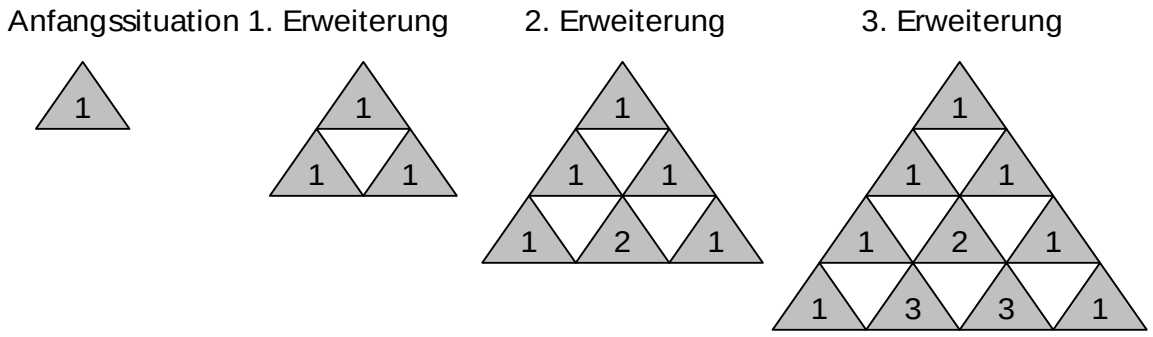
1. Rekursive Berechnung der Summe (3 Punkte)

Entwickeln Sie eine rekursive Funktion, die die Summe der ersten *n* Zahlen nach der angegebenen Formel bestimmt

$$\mathrm{sum}(n) = \begin{cases} 0 & \text{falls } n = 0 \quad (\text{Rekursionsanfang}) \\ \mathrm{sum}(n-1) + n & \text{falls } n \geq 1 \quad (\text{Rekursionsschritt}) \end{cases}$$

2. Pascal'sches Dreieck (10 Punkte)

Das Pascal'sche Dreieck ist ein Beispiel für eine Rekursion. Ausgehend von einem einzelnen gleichseitigen Dreieck wird durch sukzessive Erweiterung d.h. hinzufügen neuer Zeilen, die jeweils ein Dreieck mehr als die vorherige enthält, ein neues gleichseitiges Dreieck erzeugt.



Jedem Dreieck ist ein numerischer Wert zugeordnet. Dreiecke am Rand haben der Wert 1, andere berechnen ihren Wert aus der Summe der Nachbarn in der vorherigen Zeile.

Mathematisch lässt sich der Zusammenhang wie folgt darstellen:

$$C(n,k) = C(n-1,k-1) + C(n-1,k), n...aktuelle\ Zeile, k...aktuelle\ Spalte$$

Entwickeln Sie `eine rekursive Lösung` für dieses Problem. Als Abbruchbedingung soll die Anzahl der Erweiterungen dienen. Geben Sie die berechneten Werde in „ansprechender" Form am Bildschirm aus. Beachten Sie, dass die Werte des Dreiecks nicht gespeichert werden dürfen.

3. Strukturen in C (7 Punkte)

Neben skalaren Datentypen gibt es auch strukturierte Datentypen wie Felder und Strukturen (`struct`) . Felder sind Ihnen hinlänglich bekannt. Strukturen wurden bis jetzt nicht behandelt. Erarbeiten Sie sich das Thema selbständig und bereiten Sie dieses so vor, dass Sie Ihren Mitstudierenden die wesentlichen Aspekte dazu näherbringen können. Zeigen Sie die verschiedenen Aspekte anhand von kleinen Codebeispielen.

1 Rekursive Berechnung der Summe

Die Berechnung der summe mithilfe der recusion erfolgt mit einem immer wieder sich selber aufrufende funktion. Siehe Angabe für die Formel.

Testfälle:
n = 5, 0, -5, 100

C-Program code: Siehe beigelegtes recursiveCalc.c/.h file.

2 Pascal'sche Dreieck

Bei jeder einzelnen Row werden zuerst die Leerzeichen ausgegeben dannach werden die Nummern mithilfe einer Rekursion (Formel siehe Angabe) berechnet und auch ausgegeben.

Testfälle:
dim = 3, 6, 20, -10, 0

C-Program code: Siehe beigelegtes pascalTriangle.c/.h file.

3 Structuren in C

Strukturen oder auch **structs** ist ein Datentype von der Familie der strukturierten Datentypen. In diesem Datentype kann man verschiedene Variablen, von gleichem oder auch anderem Datentype, gebündelt speichern. In **structs** kann man auch strukturierte Datentypen speichern. Die Werte in **structs** sind solange rekursiv zerlegbar, bis nur noch Werte skalarer Datentypen übrig bleiben.

Syntax (C) Aufbau eines structs

Deklaration ohne Variablennamen

```
struct datum
{
  int tag;
  char monat[10];
  int jahr;
};
```

Deklaration mit Variablennamen.

```
struct datum
{
  int tag;
  char monat[10];
  int jahr;
} geburtstag, urlaub;
```

Die zweite Möglichkeit besteht darin, die Struktur zunächst wie oben zu deklarieren und Variablen der Struktur später zu erzeugen

```
struct datum geburtstag, urlaub;
```

Die Zuweisung kann komponentenweise erfolgen und eine Initialisierung erfolgt immer mit geschweifter Klammer

```
struct datum geburtstag = {7, "Mai", 2005};
```

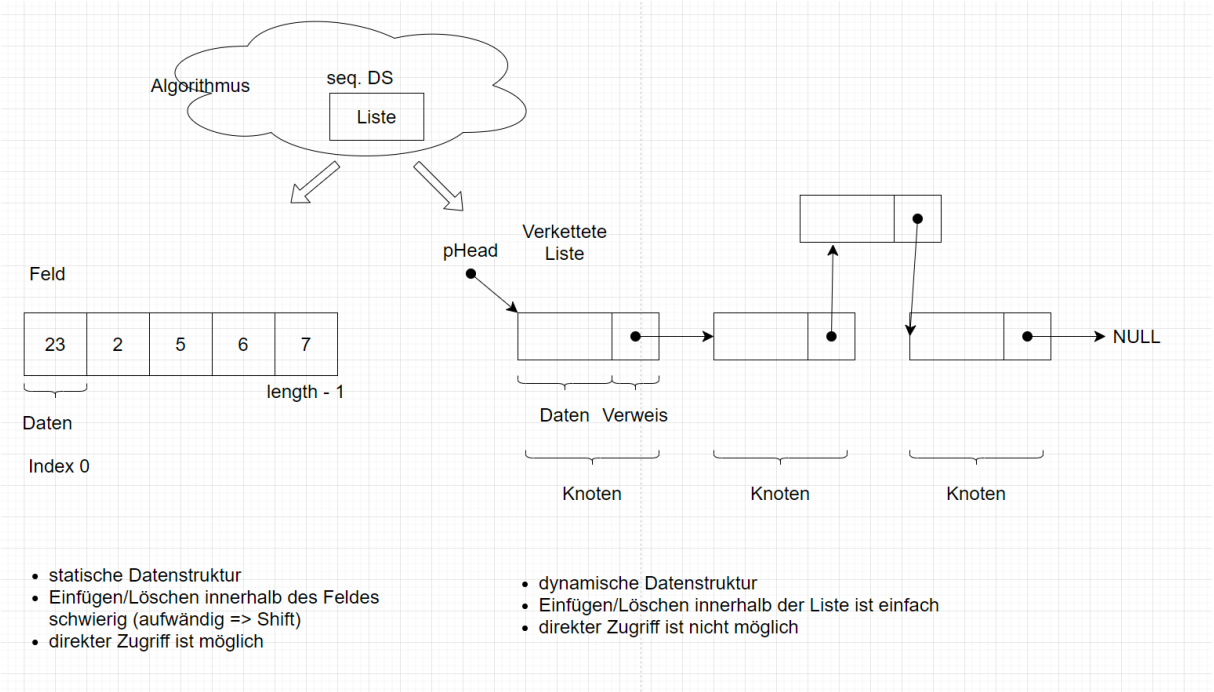
Man kann auch eine typdefinition erstellen

```
typedef struct datum
{
   // Struktur-Elemente
} datum;

datum heute;
struct datum heute;
```

Beispiel für Structs

Verkettete Liste



Rucksackpacken

C-Program code: Siehe beigelegtes pascalTriangle.c/.h file.

Tokenizer

C-Program code: Siehe beigelegtes pascalTriangle.c/.h file.