Übungsserie 5

Rekursion

Seminar in KW 49

1 Rekursion für einfache Zählschleife

Gegeben sind die ungeraden Zahlen im Intervall von $[a, b]$, wobei a und b selbst ungerade sind.

- Schreiben Sie eine rekursive Funktion **int** `berechnetSumme`, die die ungeraden Zahlen in dem Intervall aufaddiert! Die Aufgabe soll ohne die Verwendung von Schleifen gelöst werden.
- Kommentieren Sie bei der Funktion den **Rekursionsschritt** sowie die **Abbruchbedingung**!
- Handelt es sich bei der Rekursion um lineare Rekursion oder Baumrekursion?

2 Pascalsches Dreieck

Das Pascalsche Dreieck ist gegeben durch die Vorschrift:

$$p(n,k) = \begin{cases} 1 & k = 1 \vee k = n \\ p(n-1,k-1) + p(n-1,k) & \text{sonst.} \end{cases} \tag{1}$$

- Schreiben Sie eine rekursive Funktion **int** `pascal`, die die Zahl in Zeile n und Spalte k des Dreiecks bestimmt.
- Schreiben Sie eine Funktion **void** `druckeDreieck`, das die ersten N Zeilen des Dreiecks auf dem Bildschirm ausgibt.
- Schreiben Sie ein Hauptprogramm, das den Benutzer nach der Anzahl an Zeilen fragt und mithilfe der Funktion druckeDreieck die Anzahl an Zeilen ausgibt.
- Kommentieren Sie bei der Funktion den **Rekursionsschritt** sowie die **Abbruchbedingung**!
- Handelt es sich bei der Rekursion um lineare Rekursion oder Baumrekursion?

3 Zusatzaufgabe: Quersumme

Die Quersumme[1] $Q(z)$ einer ganzen Zahl z kann rekursiv definiert werden.

$$Q(z) = \begin{cases} 0 & z = 0 \\ Z(z) + Q(z \div 10) & \text{sonst} \end{cases} \tag{2}$$

wobei $\div$ die Ganzzahldivision ist und $Z(z)$ den Wert der letzten Ziffer darstellt.
Die Ganzzahldivision in C++ wird implementiert durch den „/“-Operator auf Integerwerten.
$Z(z)$ ist in C++ durch die Modulooperation darstellbar, sodass `z` % 10 den Wert der Ziffer darstellt.

- Schreiben Sie eine Funktion **int** `quersumme`(**int**), die die Quersumme berechnet!
- Handelt es sich bei der Rekursion um lineare Rekursion oder Baumrekursion?

[1] Üblicherweise wird die Quersumme über eine Schleife bestimmt. Diese Aufgabe dient hat als Ziel, eine typische Schliefe rekursiv darzustellen.

4 Zusatzaufgabe: Größter gemeinsamer Teiler (ggT)

Hier ist ein alternatives Verfahren zu dem Verfahren aus der Vorlesung zur Bestimmung des ggT zugrunde gelegt. Der größte gemeinsame Teiler $ggT(a, b)$ zweier natürlicher zahlen a, b ist definiert als

$$ggT(a,b) := \begin{cases} ggT(b,a) & a < b \\ b & (a \mod b) \equiv 0 \\ ggT(b, a \mod b) & \text{sonst.} \end{cases} \tag{3}$$

Die Modulooperation ist erneut durch `a % b` gegeben.

- Schreiben Sie eine Funktion `int ggT(int, int)`, die den größten gemeinsamen Teiler nach der angegebenen Vorschrift berechnet!
- Handelt es sich bei der Rekursion um lineare Rekursion oder Baumrekursion?

5 Zusatzaufgabe: Fibonaccizahlen

Fibonaccizahlen sind eines der klassischen Beispiele für Rekursion. In Aufgabe 3 haben wir diese Iterativ implementiert. Der Grund dafür liegt einerseits darin, dass wir zu dem Zeitpunkt noch keine Funktionen kannten, zum anderen, dass die iterative Lösung effizienter ist als die rekursive. Die Aufgabe ist jetzt, die klassische Lösung rekursiv zu implementieren.

Die Fibonacci-Folge ist definiert als die Folge der Zahlen $\text{Fib}(n)$, $n \in \mathbb{N}$ gegeben durch:

$$\text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \tag{4}$$

mit

$$\text{Fib}(1) = 1, \tag{5}$$
$$\text{Fib}(2) = 1. \tag{6}$$

- Schreiben Sie eine Funktion `fibonacci`, die rekursiv die n-te Fibonaccizahl bestimmt.
- Handelt es sich bei der Rekursion um lineare Rekursion oder Baumrekursion?

6 Zusatzaufgabe: Nullstellensuche mit Bisektion

Im vergangen Aufgabenblatt haben Sie das Newtonverfahren zur Nullstellensuche bereits kennengelernt. Ein wesentlich einfacheres Verfahren ist das Verfahren über die Bisektion.

Sei $f(x) : \mathbb{R} \to \mathbb{R}$ eine monotone Funktion in dem Intervall $[a, b]$ und $f(a) < 0 < f(b)$. Dann lässt sich die Nullstelle durch folgendes Verfahren bestimmen: Berechne $c = \frac{a+b}{2}$. Bestimme den Funktionswert von $f_c = f(c)$. Ist $f_c < 0$ so muss die Nullstelle im Intervall $[c, b]$ liegen. Ist $f_c > 0$, so muss die Nullstelle im Intervall $[a, c]$ liegen. Die Nullstelle ist dann gefunden, wenn $f_c - 0 < \epsilon$ bei vorgegebenem Epsilon.

Die Aufgabe dreht sich um die Bestimmung der Nullstelle von $f(x) = -\cos(x)$ im Intervall $[0, 2]$.

- Implementieren Sie das Verfahren (genannt Bisektion) **rekursiv**.
- Implementieren Sie zum Vergleich das Verfahren **iterativ**.

In der Analysis wird häufig die Kreiszahl π definiert über $\frac{\pi}{2}$ als die kleinste positive Nullstelle der Funktion $\cos(x)$ (alternativ, die Nullstelle der Funktion $\cos(x)$ im Intervall $[0, 2]$). Nutzen Sie die Reihenentwicklung des Cosinus (oder die Funktion `std::cos`) zur Bestimmung von π mit Hilfe der Bisektion.