Übungsaufgaben A10 und A11

A10

Für *0 < x < 2* kann der natürliche Logarithmus *ln(x)* mit der folgenden Reihenformel beschrieben werden:

$$\ln(x) = \sum_{i=1}^{\infty} (-1)^{i-1} \frac{(x-1)^i}{i}$$

Für diese Formel ist schrittweise ein recheneffizienter Algorithmus zu entwickeln.

Teilaufgaben:

a) Schreiben Sie die ersten drei Summanden (für i=1, i=2 und i=3) auf! Die Abhängigkeit von x bleibt bestehen.

b) Schreiben Sie für die drei Summanden aus a)
   - einen Vorzeichenteil, der durch $(-1)^{i-1}$ bestimmt wird,
   - einen Zählerteil, der durch $(x-1)^i$ bestimmt wird,
   - und einen Nennerteil, der durch i bestimmt wird,

   jeweils mit eingesetzten i-Werten für i=1, i=2 und i=3.

c) Geben Sie nun an, welcher Startwert jeweils für den Vorzeichenteil, für den Zählerteil und den Nennerteil festzulegen ist, um die Summanden daraufhin in einem Zyklus zu bilden und aufzusummieren!

d) Geben Sie an, wie sich Vorzeichenteil, Zählerteil und Nennerteil beim Übergang zum darauffolgenden Summanden ändern!

e) Geben Sie einen Algorithmus zur Berechnung von *ln(x)* nach der oben beschriebenen Weise als PAP an! Dabei sind
   - *x* und eine Schranke (oft als Epsilon bezeichnet) einzugeben und zu prüfen, ob *x* im o.g. Bereich liegt und die Schranke zwischen 0 und 1 liegt.
   - die Startwerte zu setzen (gemäß c)
   - eine Summe auf Null zu setzten
   - in einem Zyklus ein Summand aus dem Vorzeichenteil, Zähler- und Nennerteil zu bilden und auf die Summe aufzuaddieren. Für den danach folgenden Zyklendurchlauf sind Vorzeichenteil, Zähler- und Nennerteil gemäß d) zu ändern.
   - Der Zyklus soll abgebrochen werden, wenn der Betrag des Summanden kleiner als eine vorab eingegebene Schranke wird.

f) Setzen Sie den in e) aufgestellten Algorithmus als C-Programm um!

A11

Die unten angegebene rekursive Berechnungsvorschrift für 2^n soll wie in den Teilaufgaben beschrieben umgesetzt und eingesetzt werden. Die Vorschrift gilt für ganzzahlige n, mit n>=0.

$$2^n = \begin{cases} 1, & wenn\ n = 0 \\ \left(2^{\frac{n}{2}}\right)^2, & wenn\ n > 0\ und\ \text{geradzahlig} \\ 2 * \left(2^{\frac{n-1}{2}}\right)^2, & wenn\ n > 0\ und\ \text{ungerade} \end{cases}$$

a) Geben Sie den Algorithmus in einem Struktogramm an! Das Struktogramm soll ein Unterprogramm repräsentieren. Der Übergabeparameter ist n. Das Ergebnis soll durch return zurückgegeben werden. Achten Sie darauf, dass nur so viele rekursive Aufrufe auszulösen sind, wie tatsächlich notwendig sind.

b) Schreiben Sie ein C-Programm mit einer Funktion, die den Algorithmus aus Teilaufgabe a) umsetzt!

c) Zählen Sie die rekursiven Aufrufe bei der Berechnung ausgewählter Potenzen! Das kann beispielsweise für 2^1 ,2^4, 2^{10}, 2^{24} gemacht werden. Ergibt die rekursive Berechnung Vorteile gegenüber anderen Berechnungswegen?

d) Erweitern Sie das C-Programm so, dass zu Testzwecken 2^n für n-Werte von aufsteigend von 0 bis 10 berechnet und ausgegeben werden! Dabei ist die Funktion aus b) zur Berechnung der einzelnen Werte zu verwenden, auch wenn es für diesen Anwendungsfall einen effizienteren Algorithmus gibt.