

# AIK Programmieren 1 Übung: Rekursive Funktionen

*Klaus Kusche*

## 1.) Rekursive Ausgabe einer Zahl

Schreib eine *Funktion*, die einen **int** *als Dezimalzahl ausgibt* (so wie es **printf** macht). Als einzige Ausgabe-Funktion steht dir **putchar(c)** aus **stdio.h** zur Verfügung, wobei *c* ein *einzelnes Zeichen* (bzw. ein *ASCII-Wert*) ist (erinnere dich, wie man aus einem *Ziffernwert* zwischen 0 und 9 den *ASCII-Wert* des Ziffern-Zeichens berechnet!).

Verwende folgende *Idee*:

- Wenn die Zahl *negativ* ist, wird ein '-' ausgegeben und das Vorzeichen der Zahl umgedreht.
- Wenn die Zahl *größer als 9* ist, werden zuerst *die vorderen Stellen* (d.h. die Zahl mit der letzten Ziffer wegdividiert) mit einem *rekursiven Aufruf ausgegeben*.
- Zuletzt wird die *letzte Ziffer* der Zahl als *einzelnes Zeichen* ausgegeben.

Die Funktion hat *keinen* Returnwert.

Schreib dazu ein **main**, das beliebig viele Worte auf der Befehlszeile der Reihe nach mit **atoi** in einen **int** verwandelt und diesen mit deiner Funktion ausgibt.

## 2.) Rekursives Berechnen von $a$ hoch $n$

Gesucht ist eine Funktion **hoch**, die  $a^n$  ( $a$  ist eine Kommazahl,  $n$  eine ganze Zahl) berechnet und als Returnwert zurückliefert.

Verwende folgende *rekursive Idee*:

- Wenn  $n$  *gleich* 0 ist, ist das Ergebnis 1.
- Wenn  $n$  *kleiner* 0 ist, ist das Ergebnis  $1 / a^n$ .
- Wenn  $n$  *gerade* ist, ist das Ergebnis  $(a * a)^{n/2}$ .
- Wenn  $n$  *ungerade* ist, ist das Ergebnis  $a * (a^{n-1})$ .

Schreib dazu ein **main** zum Testen deiner Funktion

(2 Zahlen von der Befehlszeile einlesen, **hoch**-Funktion aufrufen, Ergebnis ausgeben).

### 3.) *Rekursives Ermitteln der Primteiler*

Gesucht ist ein Programm, das mit einer ganzen Zahl auf der Befehlszeile aufgerufen wird und die *Primteiler* (Primfaktoren) dieser Zahl ausgibt.

Es soll dazu eine *rekursive Funktion* mit *zwei Argumenten* und *keinem Returnwert* benutzen:

- Das *erste Argument zahl* ist das, was von der ursprünglichen Zahl noch übrig ist (d.h. die schon ausgegebenen Primteiler sind schon wegdividiert!).
- Das *zweite Argument teiler* ist der Wert, ab dem die Teiler noch gesucht werden müssen (d.h. alle Teiler *kleiner teiler* sind schon ausgegeben und wegdividiert).

Die Funktion soll alle Primteiler von *zahl* größergleich *teiler* suchen und ausgeben. Wenn man die Funktion mit der ursprünglichen Zahl und 2 für *teiler* aufruft, erhält man alle Primteiler der Zahl.

Verwende dazu folgende *Idee*:

- Ist *teiler* schon *größer* als *zahl*, wird man keine Teiler mehr finden und kann *zurückkehren*.
- Ist *zahl* *glatt* durch *teiler teilbar*, so wird *teiler* als Teiler *ausgegeben*.  
Dann werden *rekursiv* die weiteren Teiler der verbleibenden Zahl *zahl / teiler* ab *teiler* gesucht (denn *teiler* könnte *eventuell noch einmal* in *zahl* vorkommen).
- Ist *zahl nicht* durch *teiler teilbar*, so werden *rekursiv* die Teiler von *zahl* ab *teiler+1* gesucht.

Die Funktion soll sich auch bei *falschem Aufruf* vernünftig verhalten:

- Ist *zahl* kleiner 0, so wird mit *-zahl* weitergerechnet.
- Ist *teiler* kleiner 2, so wird mit *teiler* gleich 2 begonnen (Primteiler kleiner 2 gibt es nicht).