Übung 6

Aufgabe 20

Welche der nachfolgenden Variablendefinitionen mit Initialisierung sind korrekt, welche Wirkung besitzen sie bzw. welche Unterschiede bestehen zu den anderen:

Einfache Felder:

a) int i[] = {1,2,3};

b) int i[3] = {1,2,3};

c) int i[3] = {1,2};

d) int i[] = {1,2};

Aufgabe 21

Entwickeln Sie eine Funktion, die einen internen Puffer besitzt, aus dem sie bei jedem Aufruf das jeweils nächste Zeichen zurückliefert. Falls im Puffer keine Zeichen mehr vorrätig sind soll der Puffer wieder gefüllt werden (Eingabe einer Zeichenkette vom Benutzer).

Aufgabe 22

Erstellen Sie ein Programm, das die Determinante einer 3x3-Matrix ganzer Zahlen nach der Regel von Sarrus berechnet. Diese lautet:

$$\det(A) = \begin{vmatrix} a_{0,0} & a_{0,1} & a_{0,2} \\ a_{1,0} & a_{1,1} & a_{1,2} \\ a_{2,0} & a_{2,1} & a_{2,2} \end{vmatrix} = \begin{aligned} & a_{0,0}a_{1,1}a_{2,2} + a_{1,0}a_{2,1}a_{0,2} + a_{2,0}a_{0,1}a_{1,2} \\ & - a_{0,2}a_{1,1}a_{2,0} - a_{1,2}a_{2,1}a_{0,0} - a_{2,2}a_{0,1}a_{1,0} \end{aligned}$$

Aufgabe 23(Hausaufgabe Programmierübung 1)

Binomialkoeffizienten

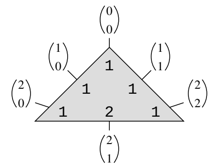
Die Binomialkoeffizienten $\binom{n}{k}$ lassen sich durch folgende rekursive Berechnungsvorschrift ermitteln:

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

$$\binom{0}{0} = 1$$

$$\binom{n}{k} = 0 \qquad \text{für} \begin{cases} k > n \\ k < 0 \\ n < 0 \end{cases}$$

Diese Berechnungsvorschrift spiegelt sich im Pascalschen Dreieck wider: In Zeile n an k-ter Stelle steht der Wert des Binomialkoeffizienten $\binom{n}{k}$, der sich aus der Addition der beiden darüberliegenden Werte ergibt:



a) Schreiben Sie eine Funktion, die **rekursiv** – analog der oben angegebenen Formel – den Binomialkoeffizienten $\binom{n}{k}$ berechnet.

b) Die Binomialfunktion aus a) soll in einem eigenen Modul liegen. Erstellen Sie eine zugehörige Header-Datei, die den Funktionsprototypen (= Funktionsdeklaration) der Binomialfunktion enthält.

c) Implementieren Sie – in einer eigenen Datei – ein kleines Programm zum Testen des Binomialmoduls. Das Testprogramm soll wiederholt Werte für n und k einlesen, $\binom{n}{k}$ berechnen und das Ergebnis ausgeben. Überlegen Sie sich ein geeignetes Abbruchkriterium.

In der Woche vom 19.6. bis 23.6. finden keine Tafelübungen statt. In diser Woche wird in den Rechnerübungen die Programmieraufgabe 2 behandelt.

Übungen zu "Einführung in die Programmierung für Ingenieure (C)" im SS 2000

Aufgabe 24(Hausaufgabe Programmierübung 2)

Schreiben Sie ein Programm, das mit Hilfe des "Bubble-Sort"-Algorithmus ein Array (= Feld) von Zahlen sortiert. Der Benutzer soll nach der Anzahl der zu sortierenden Zahlen gefragt werden. Diese werden anschließend von ihm eingegeben.

Sie sollen ein Modul entwerfen, das die benötigten Funktionen zum initialisieren des Arrays (einlesen der Werte vom Benutzer), zur Ausgabe sowie zum sortieren des Arrays enthält (vergessen Sie nicht die Header-Datei mit den Funktionsdeklarationen). Außerdem müssen häufig zwei Werte des Arrays ihren Platz tauschen, was ebenfalls von einer Funktion des Moduls erledigt werden soll. Beachten Sie, daß sich das Array, auf dem die Funktionen arbeiten, nicht im Modul befindet, sondern als Funktionsparameter übergeben werden soll. Zudem muß den Funktionen mitgeteilt werden, wieviele Elemente des Feldes belegt sind.

Zum Testen des Moduls müssen Sie ein kleines Programm implementieren, in dem ein Array definiert wird (bitte nicht global!) und die oben genannten Anforderungen (Einlesen der Werte, Sortieren, etc) mit Hilfe des Moduls erledigt. Das Array soll so angelegt werden, daß es in jedem Fall groß genug für die aufzunehemenden Zahlen ist (z.B. 1000 Elemente).

Beschreibung des "Bubble-Sort"-Algorithmus:

Die Werte im Array werden in aufsteigender Reihenfolge sortiert, indem das Array von vorne durchlaufen wird und je zwei benachbarte Werte ihren Platz tauschen, falls der erste größer als der zweite ist. So steigt in einem Durchlauf das größte Element wie eine "Blase" ans Ende des Arrays. Das wird wiederholt, bis in einem Durchlauf keine Elemente mehr getauscht werden, also die Elemente in der richtigen Reihenfolge sind. Beachten Sie auch, daß nach jedem Durchlauf ein Element mehr an der richtigen Position steht, d.h. es müssen nur die Elemente vom Anfang des Arrays bis zu dieser Position betrachtet werden...