Lösungen:

a) Pascal-Dreieck, also die Binomialkoeffizienten
b) Wichtig: Das Erreichen einer (auch hier gegebenen) Abbruchbedingung. Beim rekursiven Aufruf werden die Argumente (teilweise) dekrementiert.
c)

```
function Pas3 (Reihe,Wert: Integer): Integer;
    var a: array[0..1000,0..1000];
        i,k,erg: integer;
   begin
   erg:=1;
   for i:= 0 to Reihe do
       for k:= 0 to i do
           if  ((k = 0)  or  (k = i))  then
                 a[i,k]:=1
            else
                 a[i,k]:=a[i-1,k-1] +a [i-1,k];
   Pas3:= a[Reihe,Wert];
   end;
```

(oder besonders schlau: Pas3:=Reihe!/(Wert!*(Reihe-Wert)! Bzw. Pas3:=Reihe *nCr* Wert; die Funktionen sind nicht unbedingt implementiert.)

Für eine ausreichende Leistung sollten die Funktionswerte richtig berechnet werden sowie das Erreichen der Abbruchbedingung genannt werden; für eine gute Leistung sollte die Iteration richtig skizziert werden.

Fragen:

- Welche Funktion wird durch Pas3 berechnet?
- Anzahl der Funktionsaufrufe für (2,1), (3,2)?
- Darstellung als Struktogramm
- Vor- und Nachteile von Rekursionen und Iterationen: Geschwindigkeit, Speicher, Eleganz.