Konstruktion eines Optimalen Suchbaums in Zeit $\Theta(n^2)$

Bereits eingegeben seien: Die Zahl n und n Wahrscheinlichkeiten $p_1, p_2, ..., p_n$. Der m-te Wert p_m gibt die Wahrscheinlichkeit dafür an, wie oft im Mittel nach dem m-ten Element a_m in der geordneten Folge $a_1\ a_2\ a_3\ ...\ a_n$ gesucht wird.

Ausgabe ist der Wert S[1, n]. Aus den R[i, j] kann man den optimalen Suchbaum rekonstruieren, indem man am Ende des Verfahrens mit k=R[1, n] startet, a_k in die Wurzel setzt und rekursiv mit R[1, k-1] im linken und R[k+1, n] im rechten Unterbaum von a_k fortfährt (selbst ausformulieren).

Hinweise zum Programm: Die Zeilen des Programms sind von 1 bis 20 nummeriert. Die Werte S[i+1, i] = 0.0 werden nur aus technischen Gründen mitgeführt, um Abfragen zu sparen. Da die rechte Grenze R[i+1, j] nach dem Erhöhen von i um 1 zur linken Grenze R[i, j-1] wird, innerhalb derer k läuft, so kann man dort links:=rechts setzen, muss nur am Anfang rechts mit R[i, i+diff-1] für i=1, also mit R[1, diff] initialisieren. Man beachte, dass stets $i \le R[i, j] \le j$ gilt.

```
var  i, j, k, n, diff, links, rechts, wurzel: natural; min: real;                 1
     P: array [1..n] of real;                                                      2
     S: array [1..n+1, 0..n] of real;                                              3
     Γ: array [1..n, 1..n] of real;                                                4
     R: array [1..n, 1..n] of natural;                                             5
begin {n und das Feld P seien bereits eingegeben }  S[1,0] := 0.0;                 6
     for i:=1 to n do S[i+1, i] := 0.0; S[i, i] := P[i]; Γ[i, i] := P[i]; R[i, i]:=i od;   7
     for diff:=1 to n-1 do                                                         8
          rechts := R[1, diff];                                                    9
          for i:=1 to n-diff do                                                   10
               j := i + diff;  Γ[i, j] := Γ[i, j-1] + P[j];                       11
               links := rechts;  rechts := R[i+1, j];                             12
               min := S[i, links-1] + S[links+1, j];  wurzel := links;             13
               for k:=links+1 to rechts do                                        14
                    if  S[i, k-1] + S[k+1, j] < min  then                         15
                         min := S[i, k-1] + S[k+1, j];  wurzel := k  fi  od;      16
               S[i, j] := min + Γ[i, j];  R[i, j] := wurzel;                      17
          od                                                                      18
     od                                                                           19
end.                                                                              20
```