Musteraufgabe:

In einem Geschichtsrätsel werden die folgenden Daten der Rechentechnik gefragt:

1. Geburt von Blaise Pascal
2. Der erste Mikroprozessor
3. Modell der Turing-Maschine
4. Babbages Differenziermaschine
5. Einführung der modernen Lochkartentechnik
6. Erfindung der Leibnizschen Rechenmaschine
7. Erfindung des Abakus

Da es mehr auf die geschichtlichen Zusammenhänge als auf die genauen Daten ankommt, sollen die Ereignisse nur in die richtige chronologische Reihenfolge gesetzt werden. Die Bewertung des Rätsels ergibt sich aus der längsten (nicht notwendig ununterbrochenen) Kette richtig geordneter Ereignisse.

Beispiele:
Die Jahreszahlen
1) 1725 **2)** 1431 **3)** 1925 **4)** 1615 führen zur Folge 2 4 1 3.

Die richtige Reihenfolge sei 1 2 3 4 5 6 7.
Die Reihe 1 2 3 4 6 5 7 ergibt dann 6 Punkte für 1 2 3 4 5 7, das Ereignis 6 ist falsch eingeordnet.
Die Reihe 3 4 5 6 1 7 2 ergibt 5 Punkte für 3 4 5 6 7, die Ereignisse 1 und 2 sind falsch eingeordnet.

Aufgabe:
Bringe zunächst die Daten in die richtige chronologische Reihenfolge.
Schreibe ein Programm, welches zuerst die Zahlenfolge für die richtige Lösung einliest und dann beliebig viele Zahlenfolgen für Ratversuche. Zu jeder geratenen Zahlenfolge gibt es die Punktzahl aus.
Gib Deinem Programm die richtige Reihenfolge der obigen Ereignisse aus der Geschichte der Rechentechnik als erste Eingabe. Welche Punktzahlen erreichen die folgenden Ratversuche?

a) 2 3 5 4 6 1 7
b) 7 1 6 4 5 3 2
c) 1 2 3 4 5 6 7

Lösungsidee:
Die richtige Folge (okfolge) wird mittels einer Funktion f auf die Folge 1 2 3 4 5 6 7 abgebildet. Für die Folge 7 1 6 5 3 2 4 gilt also beispielsweise f(7) = 1, f(1) = 2, f(6) = 3 usw. (Dies ist viel einfacher als es klingt, da man nur statt eines Elementes seinen Index in der Folge betrachten muß.) Auf die geratene Folge testfolge wird die gleiche Funktion f angewendet. In der so erhaltenen Folge wird nun die längste monoton steigende Teilfolge gesucht. Diese Teilfolge muß nicht zusammenhängend sein. Zum Beispiel ist in der Folge 4 2 7 5 6 1 3 die gesuchte Teilfolge 2 5 6.
Eine (ineffiziente, aber für kleine Folgen praktikable) Methode, die gesuchte Teilfolge zu finden, ist, systematisch alle möglichen Teilfolgen zu erzeugen, sie zu prüfen, ob sie aufsteigend sind, und sich jeweils die Länge der längsten bisher gefundenen aufsteigendenTeilfolge zu merken.

Alle Teilfolgen kann man folgendermaßen erzeugen: Sei n (im Programm maxAnzahl) die Länge der Gesamtfolge (hier 7). Man speichert nacheinander die Binärdarstellung aller Zahlen von 1 bis 2^n- 1 in einem Array (binarray) der Größe n. Für jede Zahl von 1 bis 2^n- 1 stehen also an bestimmten Stellen im binarray Einsen. Man betrachtet nun in der Testfolge genau die Elemente, die an den gleichen Positionen wie die Einsen im binarray stehen und überprüft nur diese Elemente, ob sie aufsteigend sind. Im angegebenen Programm wird nicht einmal dies geprüft, sondern nur die Länge der längsten aufsteigenden Teilfolge innerhalb dieser Folge berechnet.
Beispiel: binarray 1 0 0 0 1 0 1, testfolge 1 3 4 7 6 5 2, betrachtet werden 1, 6 und 2. Hierin hat die längste aufsteigende Folge die Länge 2. Die längste so ermittelte Teilfolge (bzw. eine längste Teilfolge, wenn es mehrere gleichlange Teilfolgen gibt) ist gerade die gesuchte und ihre Länge entspricht der Punktzahl, die mit der Testfolge erzielt wurde.

Die richtige Reihenfolge lautet 7 1 6 4 5 3 2.
Daraus folgen für **a)** 1 Punkt, **b)** 7 Punkte und **c)** 3 Punkte.

Halbformale Programmbeschreibung:
```
lies okfolge ein;
wiederhole:
    lies testfolge ein;
    wende f auf testfolge an;
    (* f : Indizes der Elemente der okfolge *)
    speichere so erhaltene Folge in testfolge;
    für i := 1 bis 2^n- 1:
        schreibe i in Binärdarstellung in binarray;
        betrachte nur Elemente aus testfolge,
            die an gleicher Position wie die
            Einsen in binarray stehen;
        berechne für diese Elemente
            längste aufsteigende Teilfolge;
        aktmax := Länge dieser Teilfolge;
        wenn aktmax > gesamtmax:
            aktualisiere gesamtmax;
    gib gesamtmax aus
bis keine weiteren Versuche gewünscht;
gib okfolge aus.
```

Programmtext:
```
PROGRAM Raetselecke;

(* Es werden keinerlei Fehlerabfragen gemacht, *)
(* falsche Eingaben sind also möglich!         *)

CONST
  maxAnzahl = 7;

VAR
  okfolge, testfolge : ARRAY[1..maxAnzahl] OF
      INTEGER;
  laenge, jj : INTEGER;
  weiter : CHAR;

PROCEDURE liesOkFolge;
  VAR i, zahl : INTEGER;
  BEGIN
    FOR i:=1 TO maxAnzahl DO
      BEGIN
        write ('Bitte die ',i,'-te Zahl der richtigen Folge ');
        readln(zahl);
        okfolge[zahl] := i
      END;
    writeln
  END; (* liesOkFolge *)

PROCEDURE liesTestFolge;
  VAR i, j, zahl : INTEGER;
  BEGIN
    FOR i := 1 TO maxAnzahl DO
      BEGIN
        write ('Bitte den ',i,'-ten Testkandidaten ');
        readln(zahl);
        j := 1;
        WHILE okfolge[j] <> zahl DO
          j := j + 1;
        testfolge[i] := j
      END;
    writeln
  END; (* liesTestFolge *)

FUNCTION berechneLaenge : INTEGER;
  VAR
    binarray : ARRAY[1..maxAnzahl] of INTEGER;
    i, j, aktmax, aktlaenge, maxlaenge,
      hilfmax, ii: INTEGER;

  BEGIN
    hilfmax := 1;
    FOR i  := 1 TO maxAnzahl DO
      (* berechne 2 hoch maxAnzahl *)
      hilfmax := hilfmax * 2;
    hilfmax := hilfmax - 1;
    maxlaenge := 0;
    FOR i := 1 TO hilfmax DO
      BEGIN
      ii := i;
      FOR j := maxAnzahl DOWNTO 1 DO
      (* fülle binarray *)
        BEGIN
        binarray[j] := ii mod 2;
        ii := ii div 2
        END;
      aktmax := 0;
      aktlaenge := 0;
      FOR j := 1 TO maxAnzahl DO
        IF binarray[j] = 1 THEN
          IF testfolge[j] > aktmax THEN
          BEGIN
            aktmax := testfolge[j];
            aktlaenge := aktlaenge + 1
          END;
        IF aktlaenge > maxlaenge THEN
          maxlaenge := aktlaenge

    END;
      berechneLaenge := maxlaenge
  END; (* okCount *)

BEGIN  (* Hauptprogramm *)
  liesOkFolge;
  weiter := 'j';
  WHILE (weiter = 'j') OR (weiter = 'J') DO
  BEGIN
    liesTestFolge;
    laenge := berechneLaenge;
    writeln ('Die Punktzahl ist ',laenge);
    writeln
    write ('Neuer Versuch? (j/n): ');
    readln(weiter);
    writeln
  END;
  writeln;
  write('Die richtige Reihenfolge ist  ');
  FOR jj := 1 TO maxAnzahl DO
    write(okfolge[jj],'  ');
  writeln;
  writeln
END.
```

(Die Programmdokumentation und die Bildschirmausgabe entfallen aus Platzgründen.)