8 Algorithmen auf Datenstrukturen

Datenstrukturen realisieren in der einfachsten Form einen Datentyp. Ein Datentyp ist eine Zusammenfassung von Objekten, einschließlich der darauf zulässigen Operationen.

Datentypen

Grundlegende Datentypen sind Felder, Arrays, Listen, Stacks, Queues, Bäume und Graphen. Dazu kommen noch zusammengesetzte Datentypen.

8.1 Permutationen

Permutationen

Jede vollständige Zusammenstellung einer endlichen Anzahl von Elementen in beliebiger Reihenfolge heißt Permutation. Aus der Mathematik ergeben sich für n Elemente n! Permutationen. Da n! eine sehr schnell wachsende Funktion ist, lassen sich Berechnungen auf Permutationen nur im unteren Zahlenbereich sinnvoll auf Rechenanlagen einsetzen.

Bevor wir zu einem Anwendungsbeispiel kommen, wollen wir uns zunächst mit der Bestimmung von Permutationen befassen. Ähnlich, wie wir n! rekursiv auf (n-1)! zurückgeführt haben, lässt sich dies auch bei den Permutationen bewerkstelligen. Setzt man die Permutationen n-1 voraus, so erhält man n Permutationen, indem die Zahl n an jede mögliche Stelle eingefügt wird.

Betrachten wir diesen Algorithmus in Struktogrammform. Dazu benutzen wir der Einfachheit halber die natürlichen Zahlen.

Tab. 8-1 Dateneingabe zur Erzeugung von Permutationen

Eingabe der Anzahl n
i=1 (1) n
x(i)=i
Permutation(1)

Harald Nahrstedt, Algorithmen für Ingenieure, ISBN 3-8348-0015-5.

Zunächst wird nach der Eingabe der Anzahl n ein Vektor x() definiert und mit natürlichen Zahlen von 1 bis n gefüllt.

Tab. 8-2 Rekursiver Algorithmus zur Erzeugung von Permutationen

Permutation (k)	
y = x(k)	
i=k (1) n	
x(k)=x(i)	
x(i)=y	
Ist k < n	
Ja	Nein
Permutation (k+1)	Ausgabe von x(i)
x(i)=x(k)	
x(k)=y	

Schon für diese Berechnung legen wir ein Tabellenblatt Permutationen an und programmieren diesen Algorithmus.

Code 8-1 Erzeugung von Permutationen natürlicher Zahlen

```
Option Explicit

Dim n, x(), j As Integer

Sub Permut_Start()
   Dim i As Integer

   ThisWorkbook.Worksheets("Permutationen").Cells.Clear
   n = InputBox("Anzahl")
   ReDim x(n)
   j = 0
   For i = 1 To n
      x(i) = i
```


Harald Nahrstedt, Algorithmen für Ingenieure, ISBN 3-8348-0015-5.

```
   Next i
   Call Permut(1)
End Sub

Sub Permut(k As Integer)
   Dim i, y As Integer

   y = x(k)
   For i = k To n
      x(k) = x(i)
      x(i) = y
      If k < n Then
         Call Permut(k + 1)
      Else
         Call Permut_Ausgabe
      End If
      x(i) = x(k)
   Next i
   x(k) = y
End Sub

Sub Permut_Ausgabe()
   Dim i As Integer

   j = j + 1
   For i = 1 To n
      Cells(j, i) = x(i)
   Next i
End Sub
```

Aufgerufen wird die Prozedur Permut_Start über einen Menüpunkt.

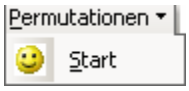

Abb. 8-1 Menü Permutationen

Das Programm liefert für n=5 genau 5!=120 Permutationen. Abbildung 8-2 zeigt die ersten Zeilen der Berechnung.

	A	B	C	D	E
1	1	2	3	4	5
2	1	2	3	5	4
3	1	2	4	3	5
4	1	2	4	5	3
5	1	2	5	4	3
6	1	2	5	3	4
7	1	3	2	4	5
8	1	3	2	5	4
9	1	3	4	2	5
10	1	3	4	5	2

Abb. 8-2 Permutationen von n=5

Wenden wir uns nun einem Anwendungsbeispiel zu.

Beispiel: Fließbandarbeit

Fließbandarbeit

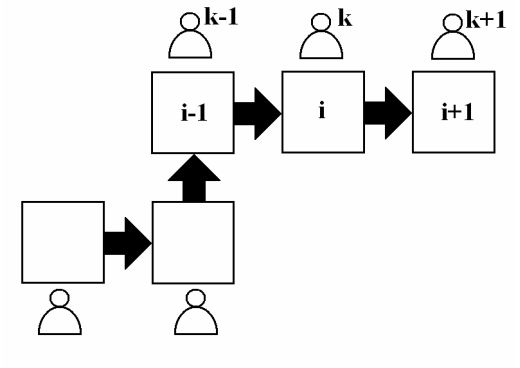

Abb. 8-3 Schema einer Fließbandarbeit

Ein Fließband hat n Stationen zur Bearbeitung. Der auf der Station i postierte Arbeiter k, übernimmt das Werkstück von der Station i-1 und übergibt sie nach der Bearbeitung an die Station i+1. Die eingesetzten Arbeiter haben an den Stationen ein unterschiedliches Arbeitsvermögen. Allgemein hat ein Arbeiter k an der Station i das Arbeitsvermögen a_{ik}. Dieses wird zum Beispiel in Stückzahl/Zeiteinheit ausgedrückt.

Das Problem besteht nun darin, die Arbeiter so den einzelnen Stationen zuzuordnen, dass der kleinste Ausstoß einer einzelnen Station maximiert wird. Dieser Wert entspricht dem Gesamtausstoß und kann auch durch erhöhte Beschickung des Fließbandes nicht überschritten werden.

Engpaßproblem

Gesucht ist also die Permutation $(p_1,\ldots,p_n)$ der Arbeitsvermögen der Arbeiter $(1,\ldots,n)$, so dass gilt

$$\min_{i=1,\ldots,n} a_{ip_i} \rightarrow Maximum\,. \tag{8.1.1}$$

Entsprechend müssen wir den vorherigen Algorithmus abändern.

Tab. 8-3 Dateneingabe zum Engpassproblem

Bestimmung der Anzahl n
i=1 (1) n
x(i)=i
k=1 (1) n
a(i,k)=Zelle(i,k)
Permutation(1)

Tab. 8-4 Auswertung des Engpassproblems

Permutation (k)	
y = x(k)	
i=k (1) n	
x(k)=x(i)	
x(i)=y	
Ist k < n	
Ja	Nein
Permutation (k+1)	Ausgabe von x(i)
x(i)=x(k)	
x(k)=y	

Harald Nahrstedt, Algorithmen für Ingenieure, ISBN 3-8348-0015-5.

Tab. 8-5 Ausgabe der berechneten Daten

∑=0	
i=i (1) n	
∑=∑+a(i,x(i))	
Ist ∑ > Max	
Ja	Nein
Max=∑	
i=1 (1) n	
Zelle(n+2,i)=x(i)	
Zelle(n+3,i)=a(i,x(i))	
Zelle(n+3,n+1)=∑	

In das neue Tabellenblatt Engpass können die Prozeduren zunächst übernommen und dann ergänzt werden.

Code 8-2 Prozeduren zum Engpassproblem

```
Option Explicit

Dim MyDoc As Object
Dim n, x() As Long
Dim a(), Max As Double

Sub Engpass_Leer()
   ThisWorkbook.Worksheets("Engpass").Cells.Clear
End Sub
Sub Engpass_Test()
   Cells(1, 1) = 7
   Cells(1, 2) = 5
   Cells(1, 3) = 11
   Cells(1, 4) = 13
   Cells(1, 5) = 14
   Cells(1, 6) = 9

   Cells(2, 1) = 3
   Cells(2, 2) = 6
```


Harald Nahrstedt, Algorithmen für Ingenieure, ISBN 3-8348-0015-5.

```
   Cells(2, 3) = 15
   Cells(2, 4) = 6
   Cells(2, 5) = 3
   Cells(2, 6) = 11

   Cells(3, 1) = 10
   Cells(3, 2) = 6
   Cells(3, 3) = 3
   Cells(3, 4) = 1
   Cells(3, 5) = 2
   Cells(3, 6) = 7

   Cells(4, 1) = 4
   Cells(4, 2) = 7
   Cells(4, 3) = 5
   Cells(4, 4) = 1
   Cells(4, 5) = 6
   Cells(4, 6) = 8

   Cells(5, 1) = 5
   Cells(5, 2) = 7
   Cells(5, 3) = 8
   Cells(5, 4) = 3
   Cells(5, 5) = 10
   Cells(5, 6) = 6

   Cells(6, 1) = 4
   Cells(6, 2) = 5
   Cells(6, 3) = 3
   Cells(6, 4) = 6
   Cells(6, 5) = 12
   Cells(6, 6) = 9
End Sub
```

```
Sub Enpass_Start()
   Dim i, k As Long
   Set MyDoc = ThisWorkbook.Worksheets("Engpass")
   n = MyDoc.UsedRange.Rows.Count
   Max = 0

   ReDim x(n), a(n, n)

   For i = 1 To n
      x(i) = i
```

```
      For k = 1 To n
         a(i, k) = Cells(i, k)
      Next k
   Next i

   Call Permut(1)
End Sub

Sub Permut(k As Integer)
   Dim i, y As Long

   y = x(k)
   For i = k To n
      x(k) = x(i)
      x(i) = y
      If k < n Then
         Call Permut(k + 1)
      Else
         Call Permut_Ausgabe
      End If
      x(i) = x(k)
   Next i
   x(k) = y
End Sub

Sub Permut_Ausgabe()
   Dim i As Integer
   Dim z As Double

   z = 0
   For i = 1 To n
      z = z + a(i, x(i))
   Next i

   If z > Max Then
      Max = z
      For i = 1 To n
         Cells(n + 2, i) = x(i)
         Cells(n + 3, i) = a(i, x(i))
      Next i
      Cells(n + 3, n + 1) = z
   End If
End Sub
```

Damit freie Daten ins Formblatt eingetragen werden können, muss zunächst der alte Inhalt gelöscht werden. Dafür sorgt die

Harald Nahrstedt, Algorithmen für Ingenieure, ISBN 3-8348-0015-5.

Prozedur Engpass_Leer. Testdaten und Auswertung sind nach dem bisherigen Schema aufrufbar.

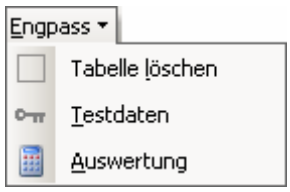

Abb. 8-4 Menü zur Fließbandarbeit

Als Beispieldaten sind sechs Arbeitsstationen mit sechs Arbeitern besetzt, deren Arbeitsvermögen an den einzelnen Stationen durch die Testdaten ausgedrückt werden.

	A	B	C	D	E	F	G
1	7	5	11	13	14	9	
2	3	6	15	6	3	11	
3	10	6	3	1	2	7	
4	4	7	5	1	6	8	
5	5	7	8	3	10	6	
6	4	5	3	6	12	9	
7							
8	4	3	1	6	2	5	
9	13	15	10	8	7	12	65

Abb. 8-5 Auswertung der Testdaten

Der Maximalwert von 65 wird erreicht, wenn in der ersten Zeile der 4. Spaltenwert, nämlich 13 genommen wird. Dann in der zweiten Zeile und der 3. Spalte der Wert 15, usw. Von allen Werten 13, 15, 10, 8, 7 und 12 ist 7 der Minimalwert. Folglich ist der Arbeitswert $a_{5,2}$ der Engpass.

Übungen

Ist die Reihenfolge der auszuführenden Arbeiten nicht fest vorgegeben und kann damit parallel erfolgen, so ist der Engpass derjenige Auftrag, der die längste Arbeitszeit erfordert. Mathematisch bedeutet dies

$$\max_{i=1,\ldots,n} a_{ip_i} \rightarrow Minimum\,. \qquad (8.1.2)$$

Das Problem ist mit dem gleichen Programm lösbar, wenn die Werte negativ eingegeben werden. Prüfen Sie dies nach.

Harald Nahrstedt, Algorithmen für Ingenieure, ISBN 3-8348-0015-5.