TA ad.

FORSCHUNG - AUSBILDUNG - WEITERBILDUNG

BERICHT NR. 18

EINFLUSS EINES LATENTWÄRMESPEICHERS AUF DEN WÄRMEFLUSS DURCH EINE ZIEGELWAND

W. DIEDRICH

UNIVERSITÄT KAISERSLAUTERN
FACHBEREICH MATHEMATIK
ERWIN-SCHRÖDINGER-STRASSE
6750 KAISERSLAUTERN

DEZEMBER 1984

Inhalt
======

Grundlagen der Aufgabenstellung

Bei den bisherigen Untersuchungen auf dem Gebiet der passiven Solarnutzung mit speicherfähigen gemauerten Ziegelwänden stellte sich heraus, daß direkt im Strahlengang der Sonnenstrahlung liegende Wandbauteile keine ausreichende Speicherwirkung haben. Ausreichend heißt: Schnelle Wärmeabführung in eine Wärmesenke, ausreichende Speicherkapazität als Tagesspeicher. Abhilfe schaffen hier Latentspeicher oder großvolumige Wasserspeicher.

In unseren konkreten Fall stand ein Texxor-Latentspeicher mit 2 l Inhalt und 100 Wh Kapazität bei 27°C in einem Poroton-Formstein zur Verfügung.(Bild 1)

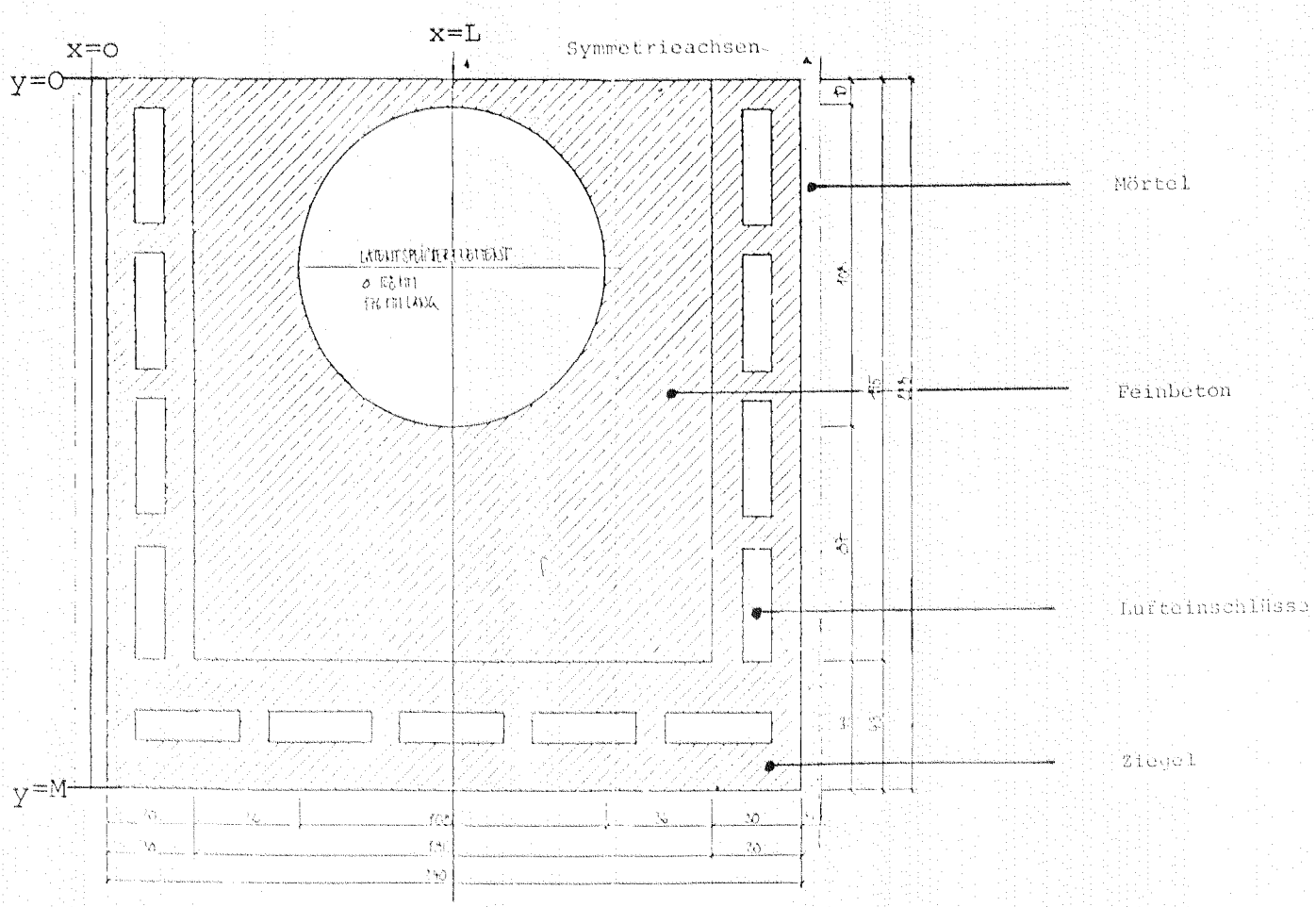


Bild 1: Aufbau des Ziegelsteins

Es soll ein Programm erstellt werden, mit dem die Gleichgewichtszustände bei verschiedenen Randtemperaturen, sowie die zeitliche Entwicklung der Temperaturverteilung bei Erwärmung des Solarspeicherziegel von außen (Sonneneinstrahlung), berechnet werden. Hierbei soll die Geometrie des Bauteils frei wählbar sein, um Konstruktionsvarianten in Bezug auf Latentspeichertemperatur und Querschnitt zu überwachen und überprüfen.

Die Aufheizung des Speicherbausteins efolgt an dem einen (dem äußeren) Rand. Der innere Rand stell die Wand eines Innenraumes dar. Innerhalb der Ziegelwand wird Periodizität angenommen; da überdies der Ziegel symmetrisch aufgebaut ist, kann man sich bei der Berechnung auf den halben Ziegel beschränken. Dieser Teil wird im folgenden mit Gebiet (G) bezeichnet.

Mathematische Grundlagen

Die zeitliche Entwicklung der Temperaturverteilung wird durch die Wärmeleitungsgleichung für die Temperatur $u(x,y;t)$ am Ort (x,y) zur Zeit t beschrieben:

$$(1) \quad \frac{\partial u}{\partial t} = \operatorname{div}(\sigma \operatorname{grad} u)$$

Dabei ist σ konstant in jedem, aus einheitlichem Material bestehenden Teilgebiet von G und berechnet sich dort aus:

$$\sigma = \frac{K}{\varrho \cdot c_p} \qquad \text{mit}$$

Wärmeleitfähigkeit	k	$[\frac{W}{m^2 \cdot K}]$,
spezifische Wärme	c_p	$[\frac{J}{Kg \cdot K}]$ und
Massendichte	ϱ	$[\frac{Kg}{m^3}]$.

Da in unserem Fall σ in G eine Treppenfunktion ist, kann $\sigma \operatorname{grad} u$ i.a. nicht überall differenziert werden. Man kann dieses Problem dadurch umgehen, daß man entweder eine schwache Lösung u von (1) in G sucht oder (1) nur in Teilgebieten mit konstantem σ fordert, wobei u dann noch an den Übergangsrändern der Teilgebiete bestimmte Randbedingungen erfüllen muß.

Beide Interpretationen des Problems ergeben die gleiche Lösung u und werden automatisch durch das Differenzenverfahren richtig approximiert.

Zur Lösung der Differentialgleichung werden noch folgende Anfangs- und Randbedingungen benötigt:

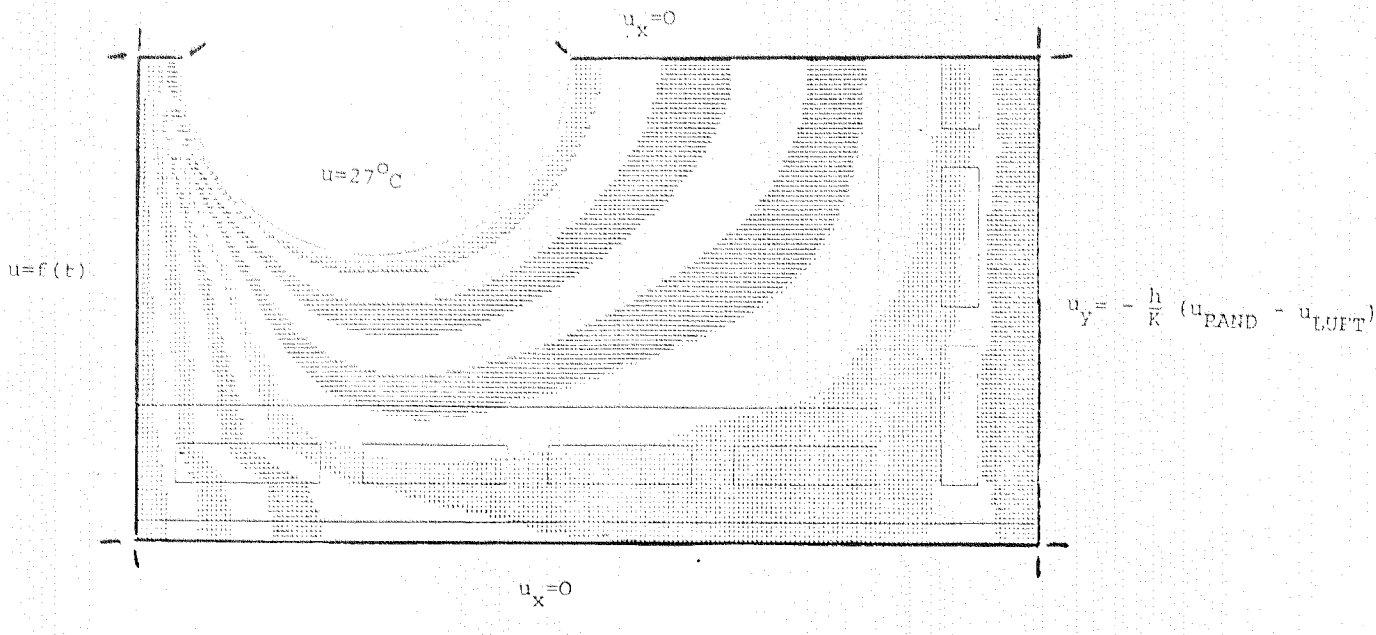


Bild 2: Randbedingungen

Der Anfangswert, d.h. die Temperaturverteilung zum Zeitpunkt T=0 sei:

(2) $u(x,y;0) = u_0(x,y)$

Zur Berechnung von u_0 wird das entsprechende stationäre Problem mit den stationären Randbedingungen gelöst (Bild 4).

Als instationäre Randbedingungen für G nehmen wir folgendes an:

Für die Ränder x=0 und x=L gilt:

(3) $\frac{\partial u}{\partial x}(x,y) = 0$ aus Symmetriegründen.

Am Innenrand y=M muß das sog. NEWTON'sche Kühlungsgesetz erfüllt sein:

(4) $(\frac{\partial u}{\partial y}) = -\frac{h}{K}(u-u_{Luft})$ [2] S.13 f

Dabei ist $h = 2{,}567 * |u_{Luft} - u_{Oberfläche}|^{1/4}$ (approximativ).

Am Rand y=0 wird der Temperaturverlauf (die tägliche Schwankung der Außentemperatur) vorgegeben:

$u(x,0;t)=f(t)$ für $0 \leqslant x \leqslant L$; $t \geqslant 0$.

Die Randbedingunng am Latentspeicher ist überall:

$u=\text{const}$ (=27°C) .

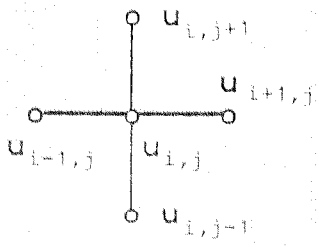
Um dieses Anfangsrandwertproblem numerisch lösen zu können, muß (1), (3) und (4) bezüglich der Zeit, als auch bezüglich des Ortes diskretisiert werden. Δx, Δy und Δt sind die Schrittweiten der Variablen x, y und t. Wir bezeichnen die Näherungslösung im Gitterpunkt $(i*\Delta x, j*\Delta y)$ zur Zeit $n*\Delta t$ mit $u_{i,j}$ wobei $0 \leqslant i \leqslant l=\frac{L}{\Delta x}$ und $0 \leqslant j \leqslant w=\frac{M}{\Delta y}$.
Zur Vereinfachung wird im Weiteren $\Delta x=\Delta y$ gewählt. Die diskretisierte Gleichung lautet dann für Gitterpunkte, die mit ihren vier Nachbarn ganz in Teilgebieten homogenen Materials liegen (Bild 3):

$$(5)\quad \frac{u_{i,j}^{(n+1)} - u_{i,j}^{(n)}}{\Delta t} = \frac{(\delta_x^2 u)_{i,j}^{(n+1)} + (\delta_y^2 u)_{i,j}^{(n+1)} + (\delta_x^2 u)_{i,j}^{(n)} + (\delta_y^2 u)_{i,j}^{(n)}}{2 \cdot (\Delta x)^2}$$

wobei $(\delta_x^2 u)_{i,j}$ die zentrale Differenz in Richtung x ist:

$(\delta_x^2 u)_{i,j} := u_{i-1,j} - 2u_{i,j} + u_{i+1,j}$; $(\delta_y^2 u)_{i,j}$ ist analog definiert.

Dieses implizite Schema wurde 1947 von CRANK und NICHOLSON zur Lösung der Wärmeleitungsgleichung eingeführt. Es ist unbedingt stabil (vergl. Richtmyer and Morton [1] S.185 ff)



Zu jedem Punkt $u_{i,j}$ in einer Gleichung gehen jeweils noch die vier benachbarten Werte ein.

Bild 3: 5-Punkte-Stern

Die Randbedingung $\frac{\partial u}{\partial x}=0$ wird mit $u_{1,j}=u_{-1,j}$ (bzw. $u_{l+1,j}=u_{l-1,j}$) diskretisiert. Man erhält dann:

$$(\delta_x^2 u)_{0,j}=-2u_{0,j}+2u_{1,j} \quad ; \quad (\delta_x^2 u)_{l,j}=-2u_{l,j}+2u_{l-1,j}$$

Für den äußeren Rand y=0 wird der vorgegebene Temperaturverlauf bei Sonneneinstrahlung diskretisiert:

$$u_{i,0}^{(n)}=f(n*\Delta t).$$

Das Newton'sche Kühlungsgesetz lautet (y=M):

$$\frac{u_{i,w}^{(n)}-u_{i,w-1}^{(n)}}{\Delta x}=-\frac{h}{k}(u_{i,w}^{(n)}-u_{Luft}) \qquad \text{h wie in (4)}$$

Als Anfangswert $u_0(x,y)$ verwenden wir eine Lösung der Zeitunabhängigen Gleichung:

(6) $\operatorname{div}(\sigma \operatorname{grad} u)=0$

dabei werden die Randwerte auf dem "äußeren" Rand y=0 durch f(0) und am "inneren" Rand y=M fest vorgegeben (Bild 4).

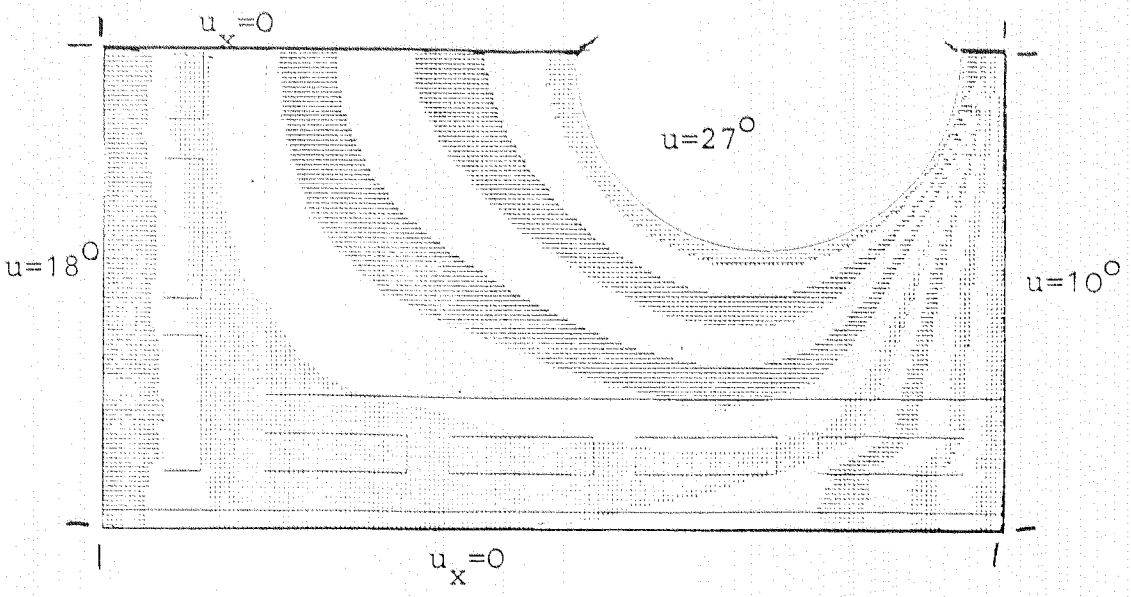


Bild 4: Randbedingungen der stationären Lösung

Die Ränder x=0 und x=L werden wie oben behandelt (Gleichung (3)).

In Gleichung (5) kommen auf beiden Seiten Werte von u zu den Zeitpunkten (n) und (n+1) vor. Ordnet man diese Gleichung bezüglich dieser Zeitschritte, so erhalt man:

$$(7)\quad u_{i,j}^{(n+1)} - c\cdot\sigma\cdot(\delta_x^2 u)_{i,j}^{(n+1)} - c\cdot\sigma\cdot(\delta_y^2 u)_{i,j}^{(n+1)} = u_{i,j}^{(n)} - c\cdot\sigma\cdot(\delta_x^2 u)_{i,j}^{(n)} - c\cdot\sigma\cdot(\delta_y^2 u)_{i,j}^{(n)}$$

$$\text{mit } c = \frac{\Delta t}{2\cdot(\Delta x)^2}.$$

Bisher haben wir nur Punkte im Inneren von homogenen Teilgebieten behandelt. Sitzen nicht alle Punkte eines Sterns (Bild 3) in einem Gebiet mit konstantem $\sigma = \sigma_1$, sondern gehören sie vielmehr zum Teil zu σ_2, so ersetzen wir den Faktor σ in den entsprechenden Summanden von $(\delta_x^2 u)_{ij} + (\delta_y^2 u)_{ij}$ durch σ_1 bzw. σ_2. Sizt ein Punkt genau auf der Grenze, so rechnen wir ihn halb zu σ_1 und σ_2 :

$$(8)\quad (\delta_x^2 u)_{i,j} = \sigma_1(u_{i-1,j} - u_{i,j}) + \sigma_2(u_{i+1,j} - u_{i,j})$$

und $(\delta_x^2 u)_{ij}$ entsprechend, falls u_{ij} auf der Grenze.

Gleichung (7) unter Berücksichtigung von (8) führt zu einem rekursiv linearen Gleichungssystem der Form:

$$(9.1)\quad A * u^{(n+1)} = B * u^{(n)} \qquad \text{mit u Vektor aller Punkte } u_{ij}.$$

Die rechte Seite B*u ist bekannt und wird zur Vereinfachung gleich b gestzt:

$$(9.2)\quad A * u^{(n+1)} = b\ .$$

In der Matrix A stehen die von σ abhängigen Koeffizienten gemäß Gleichung (7) und (8).

Numerische Lösung

Im Fall der Wärmeleitungsgleichung hat die Matrix A eine besondere Form. Der größte Teil der Matrix ist mit Nullen aufgefüllt und nur 4 Nebendiagonalen und die Diagonale selbst enthalten von Null verschiedene Elemente (Bild 5):

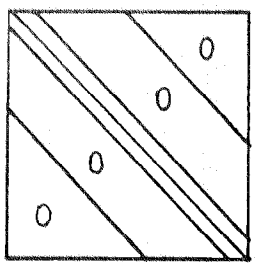


Bild 5: Schematischer Aufbau der Matrix A [3] S.150 ff

Die Größe der Matrix ist (NxN), wobei N die Anzahl der zu berechnenden Gitterpunkte des Gebietes ist: $N=l*w$.
Zur Lösung deises Gleichungssystems bietet sich das YOUNG'sche Überrelaxationsverfahren an. Dies ist ein Iterationsverfahren, bei dem der neue Wert $u^{(n+1)}$ sukzessiv durch eine Vektorfolge $^{(k)}u$ approximiert wird. Ist $\|{}^{(k)}u-{}^{(k-1)}u\|_\infty \leq \varepsilon$, wird abgebrochen und $u^{(n+1)} := {}^{(k)}u$ gesetzt. Als Startwert $^{(0)}u$ wird der Vektor $u^{(n)}$ angenommen. In Matrizenform lautet die Rekursionsformel:

$$(10)\quad ({}^{(k)}u) = (1-\omega)({}^{(k-1)}u) - \omega \cdot D^{-1} \cdot (L \cdot ({}^{(k)}u) + R \cdot ({}^{(k-1)}u) - b)$$

In Gleichung (10) ist D eine Diagonalmatrix, die aus der Hauptdiagonalen der Matrix A besteht, L eine strenge untere und R eine strenge obere Dreiecksmatrix. Es gilt daher:

$$(11)\quad A = D + L + R \ .$$

Der Parameter ω ist der Relaxationsparameter, der so gewählt wird, daß eine schnellere Konvergenz erreicht wird. Für relle positive Matrizen konvergiert das Verfahren, wenn $0<\omega<2$ gilt. Der optimale Wert liegt in der Regel zwischen 1 und 1,5 .

Realisierung durch ein Programm

Als erster Schritt wird das Gebiet mit einem Gitternetz der Schrittweite $\Delta := \Delta x = \Delta y$ überzogen. Jedem Punkt $(i*\Delta, j*\Delta)$ wird neben seinen Koordinaten i und j auch eine fortlaufende Nummer k zugeordnet. Die Randpunkte, die nicht berechnet werden, sind hiervon ausgenommen. Sie erhalten zur besonderen Kennzeichnung eine negative Zahl zugeordnet. Dies ist insbesondere wichtig für Punkte, die nahe am Rand des Latentspeichers und für Punkte auf dem oberen (dem äußeren) Rand liegen. So kann anhand der Nummer k jede Koordinate i und j wiedergefunden werden. Man braucht daher für den Laufindex keine Doppelindizierung.

Als nächstes werden die Koeffizienten (die Materialkonstanen) errechnet. Es sind nach Gleichung (7) (s.a. Bild 3) diejenigen Materialkonstanten, die über, unter, rechts und links des Punktes anzutreffen sind. Für Punkte auf einer Materialgrenze wird nach Gleichung (8) vorgegangen. Die Konstanten, multipliziert mit $c=\frac{\Delta t}{2\cdot\Delta^2}$ ergeben die Koeffizienten der Matrix A.
Als Programmiersprache für dieses Problem wurde PASCAL gewählt.

Programmbeschreibungen

1) Programm GEBIET

Dieses Programm ist, im Gegensatz zu den beiden andern Programmen, für das spezielle Gebiet geschrieben. Änderungen im geometrischen Aufbau, insbesondere die Form des Latentspeicherelements, wirken sich auf das Programm aus.
Die Aufgabe dieses Programms ist die Zuordnung der fortlaufenden Nummer k und der Koordinaten i und j der Gitterpunkte. Außerdem werden die Randpunkte des Latentspeichers errechnet. Diese Berechnungen erfolgen in Abhängigkeit von der Schrittweite (DELTA).
Parallel zur Berechnung der Nummer k wird eine Vorbesetzung der Temperatur der Gitterpunkte vorgenommen. Diese erfolgt linear von den Randpunkten aus.
Die Ausgabe der Nummer k erfolgt in die Datei POSITION, die Temperatur in die Datei WÄRME. Diese Dateien werden von dem folgenden Programm benötigt.

2) Progamm MATRIX

Aufgabe dieses Programms ist die Berechnung des Anfangswertes $u_0(x,y)$ nach Gleichung (6). Das Programm liest aus der Datei POSITION die Nummer k des Gitterpunktes bzw. die Kennzeichnung des Randpunktes. Aufgrund der Stellung der Nummer k in der Datei wird zu jedem k eine i- und j-Koordinate erzeugt. Gleizeitig werden die Temperaturen aus der Datei WÄRME gelesen und dem Punkt k zugeordnet.

Mit diesen Daten können jetzt die Materialkonstanten der Gitterpunkte berechnet werden. Dies geschieht mit der Prozedur MATERIAL. Sie beinhaltet die Geometrie des Gebietes und ist bei einer Formänderung zu überarbeiten.
Dies sind alle Daten, die zur Berechnung benötigt werden. Das Verfahren ist eine fünf-Punkt-GAUSS-SEIDEL-Iteration. Als Abbruchkriterien sind eine maximale Anzahl von Iterationen und $\|u^{(l-1)} - u^{(l)}\|_{\infty} \leq \varepsilon$ vorgesehen. Die so gewonnenen Werte $u_o(x,y)$ werden in die Datei WÄRME zurückgespeichert.

3) Programm CRNV

Die Bereitstellung der Daten (Gleichgewichtslösung aus Datei WÄRME, Materialkonstanten, Koeffizienten i und j) erfolgt analog dem Programm Matrix. Die Berechnung eines Zeitschrittes wird gemäß Gleichung (10) ausgeführt. Besondere Beachtung erfordert der "innere" Rand. Er muß gesondert für jeden Iterationschritt gerechnet werden. Der einzelne Zeitschritt wird $T/\Delta t$ mal wiederholt. Der zugehörige Temperaturverlauf f(t) am "äußeren" Rand wird als Funktion vereinbart. Die Ausgabe der Temperaturverteilung erfolgt zu festgelegten Zeitpunkten $(a*\Delta t)*n$ $a \in \mathbb{N}_0$ fest; $n=\{0,1,2,..,T/\Delta t*a\}$. Dabei wird die Temperaturskala in Bereiche eingeteilt, die einen festen Kennbuchstaben besitzen.

Möglichkeiten des Programms

Das Programm ist zur Zeit in der Lage für einen symmetrischen Ziegel, bei variablem inneren Aufbau (Dicke der Schichten, Lage und größe der Lufteischlüsse, Lage und Form des Latentspeichers, kein Latentspeicher) die zeitliche Entwicklung der räumlichen Temperaturverteilung zu berechnen. Da sich die Temperaturverteilung im inneren eines Ziegels schlecht messen läßt, ergeben sich jetzt neue Möglichkeiten der wärmetechnischen Beurteilung von Ziegeln verschiedener innerer Struktur.

Ergebnisse

Unter folgenden Rand und Anfangsbedingungen wurden für verschiedene Geometrien der zeitliche Verlauf der Temperaturverteilung berechnet.

Die Anfangsbedingung:

$u_o(x,y)$: Gleichgewichtslösung zu den Randwerten:

außen y=0 +10°C

innen y=M +18°C

Latentspeicher +27°C

Siehe hierzu Bild 2.

Die Randbedingungen:

f(t): Temperaturverlauf an der Außenwand:

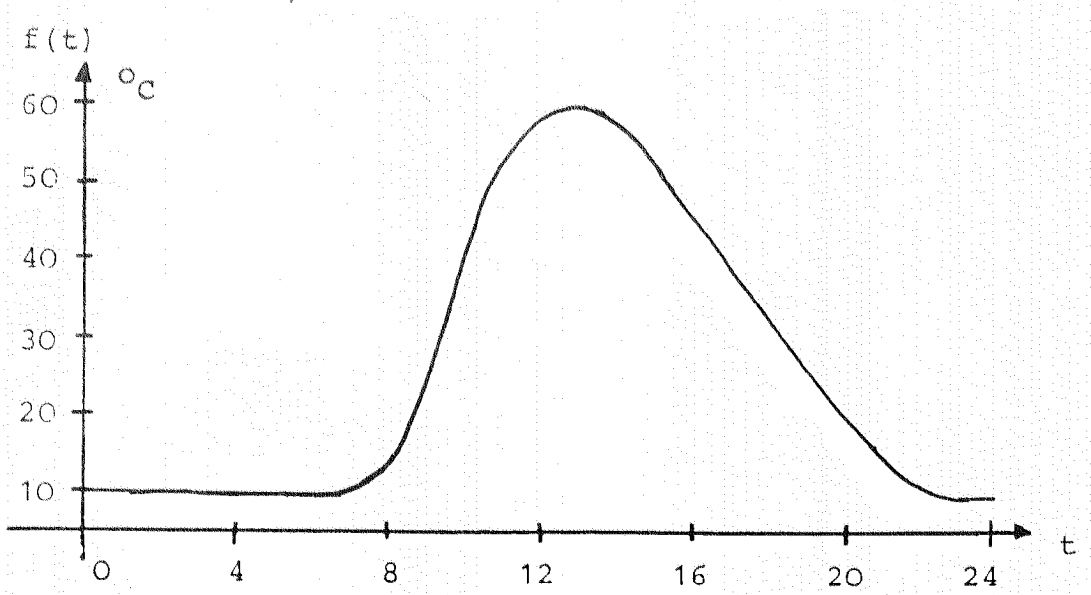


Bild 6 Temperaturverlauf

Temperatur der Innenluft: +18°C .

Für x=0 und x=L wie in Bild 2, Seite 5.

Zu den folgenden Bildern

Jeder Punkt des Bildes gibt die Temperatur entsprechend der folgenden Codierung an. Zusammenhängende Bereiche entsprechen gleichen Temperatubereichen.

Es bedeutet:

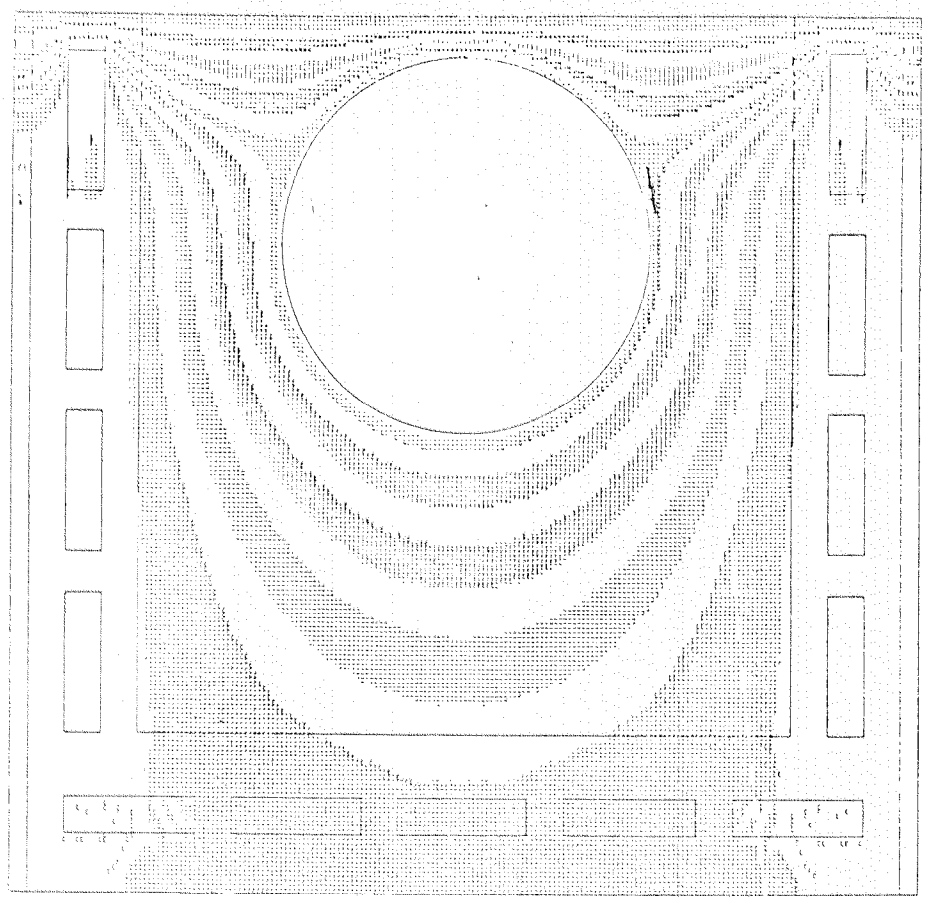
A	10,0°C	-	12,0°C
B	14,0°C	-	16,0°C
C	18,0°C	-	19,5°C
D	20,5°C	-	21,5°C
E	22,5°C	-	23,5°C
F	24,5°C	-	25,5°C
G	26,5°C	-	27,5°C
H	29,5°C	-	31,5°C
I	33,0°C	-	36,0°C
K	39,0°C	-	42,0°C
L	45,0°C	-	48,0°C
M	51,0°C	-	54,0°C
O	57,0°C	-	60,0°C

in den Bildfolgen ist als Startuhrzeit T=0 7 Uhr morgens angenommen. Die obere Bildfolge stellt den Versuch mit den eingezeichneten Luftlöchern dar. In der unteren Bildfolge wird ohne Luft gerechnet, d.h. der Ziegelkörper wird als homogen angenommen. In beiden Fällen wird eine getrennte Gleichgewichtslösung berechnet.

Temperatur am oberen Rand: . °C

Temperatur am unteren Rand: 18.0°C

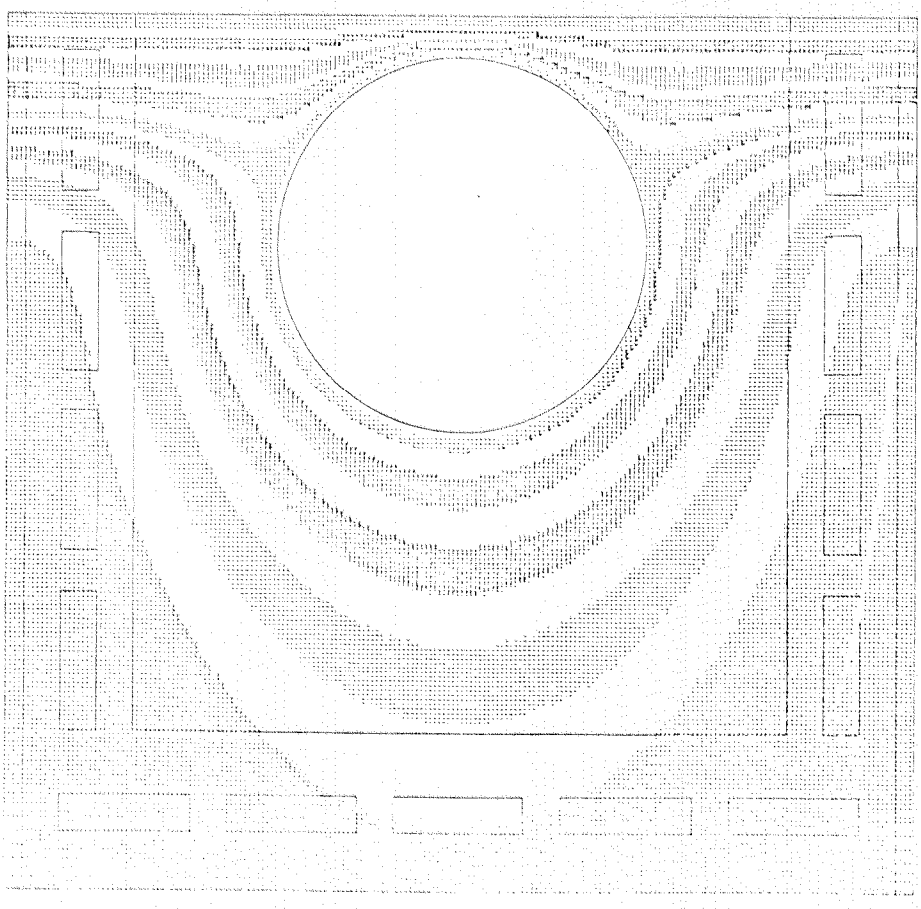
Zeitpunkt T= 4h mit Luft



Temperatur am oberen Rand: [illegible]°C

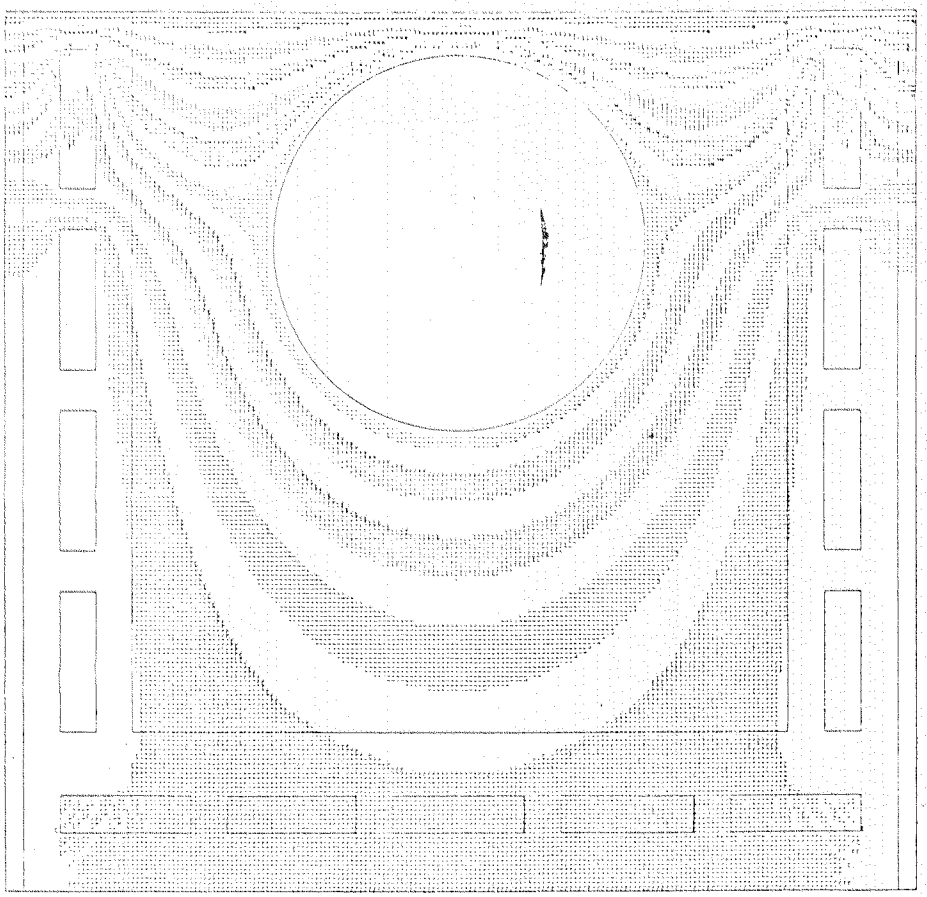
Temperatur am unteren Rand: 18.0°C

Zeitpunkt T= 4h ohne Luft



Temperatur am unteren Rand: 18.0°C

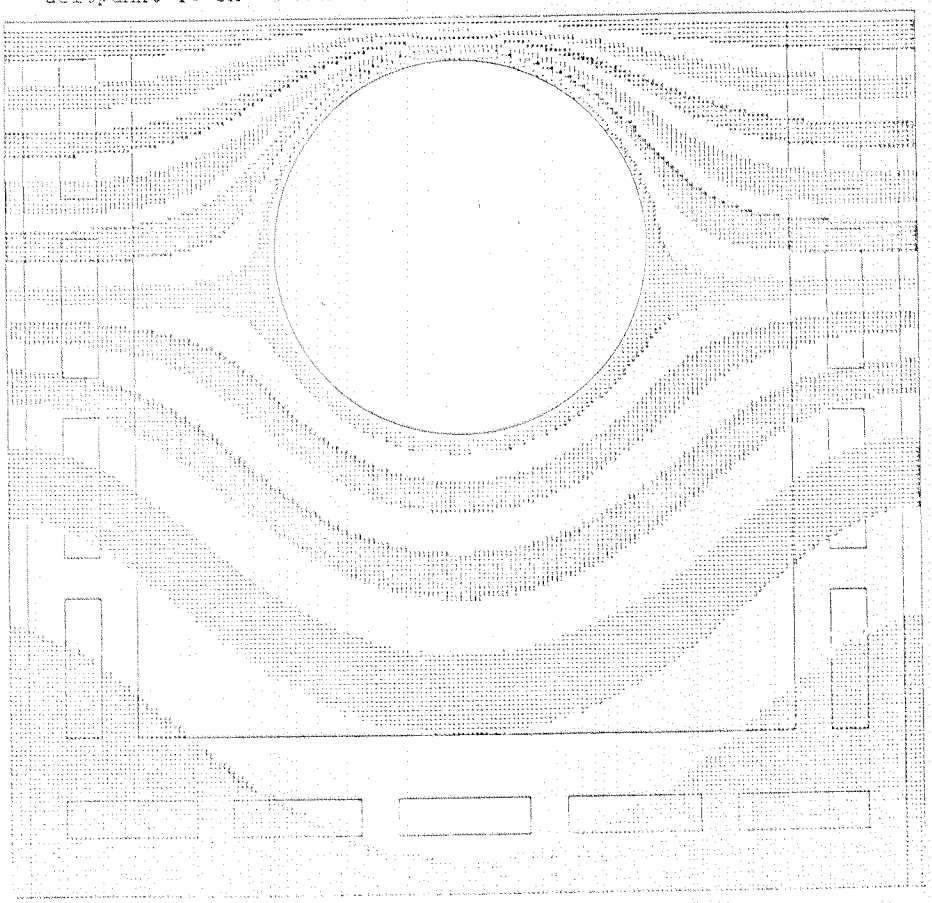
Zeitpunkt T= 8h mit Luft



Temperatur am oberen Rand: 55.2°C

Temperatur am unteren Rand: 18.5°C

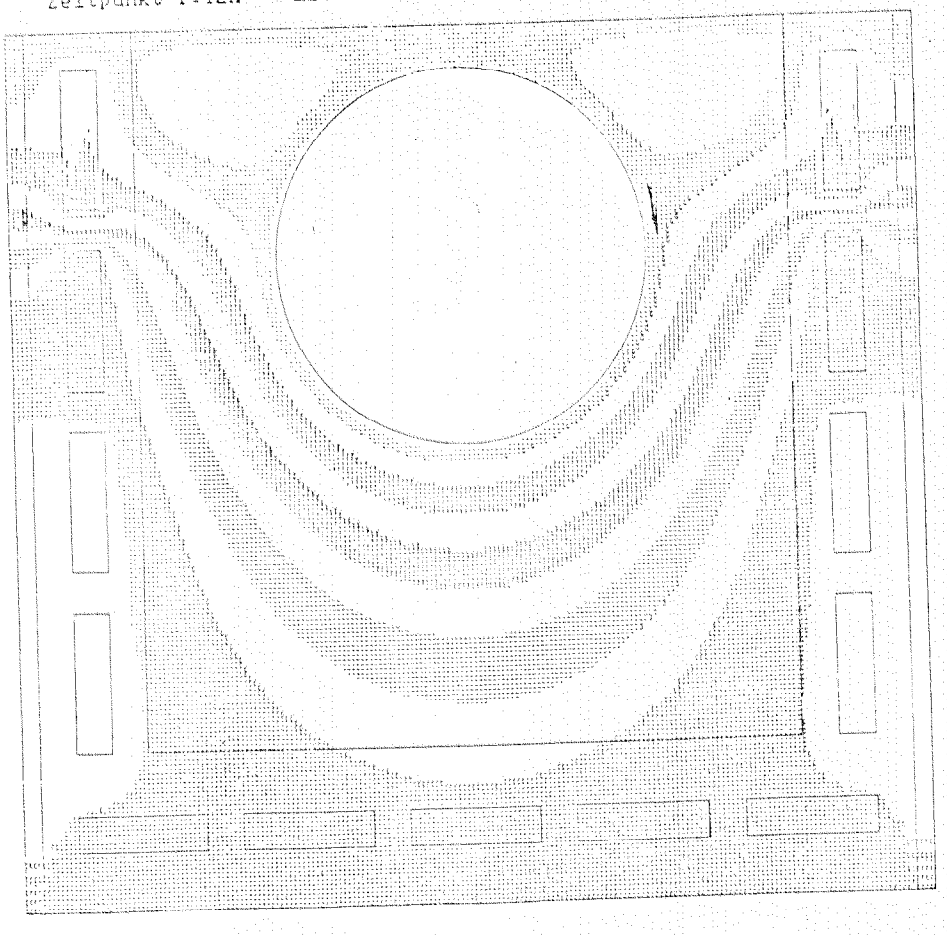
Zeitpunkt T= 8h ohne Luft



Temperatur am oberen Rand: 27.3°C

Temperatur am unteren Rand: 18.0°C

Zeitpunkt T=12h mit Luft



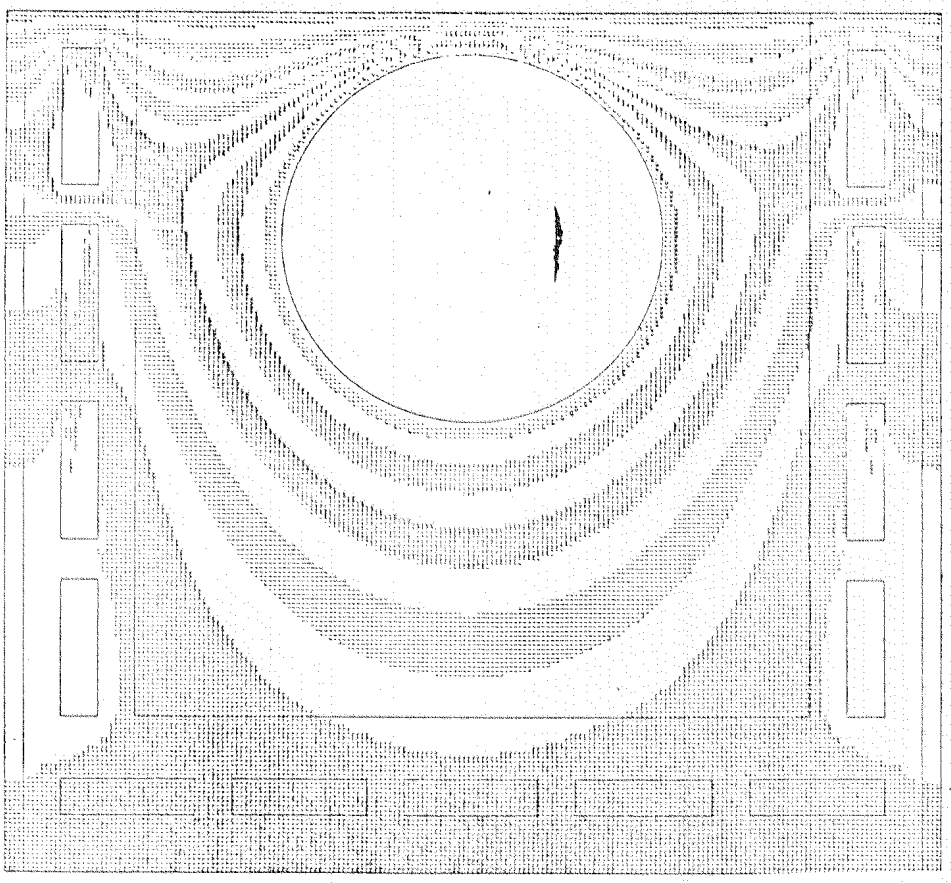
Temperatur am oberen Rand: 27.3°C

Temperatur am unteren Rand: 19.0°C

Zeitpunkt T=12h ohne Luft

Temperatur am unteren Rand: 18.1°C

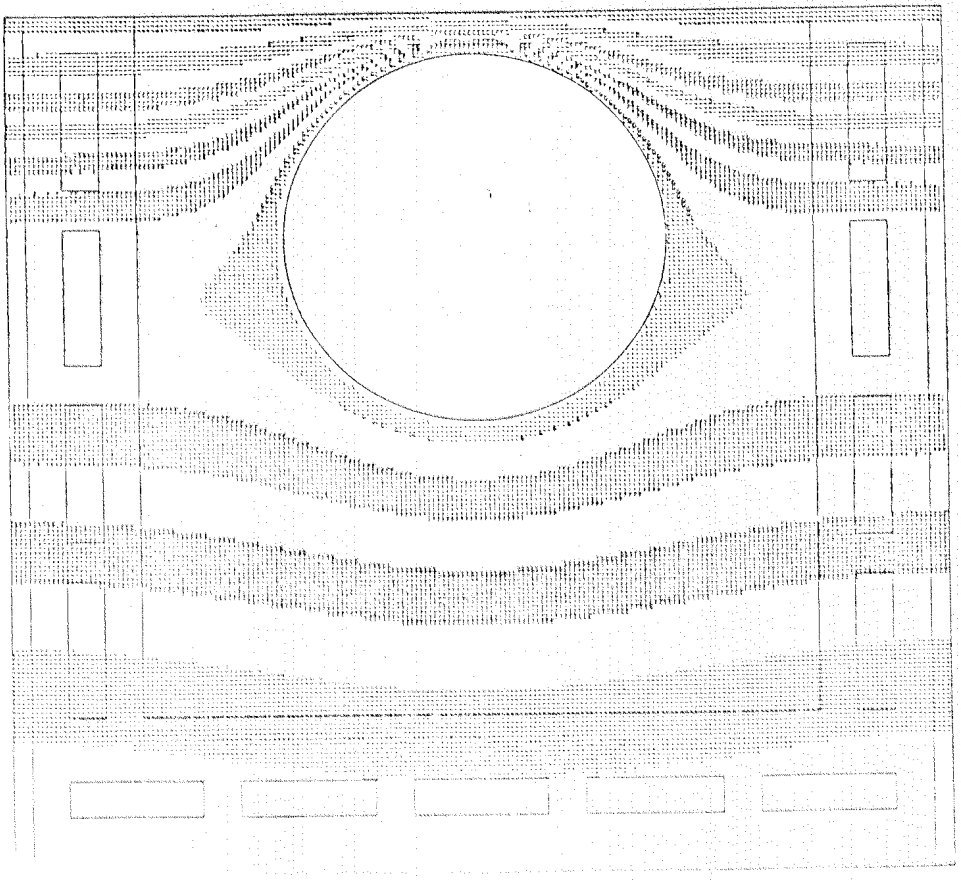
Zeitpunkt T=16h mit Luft



Temperatur am oberen Rand: 10.0°C

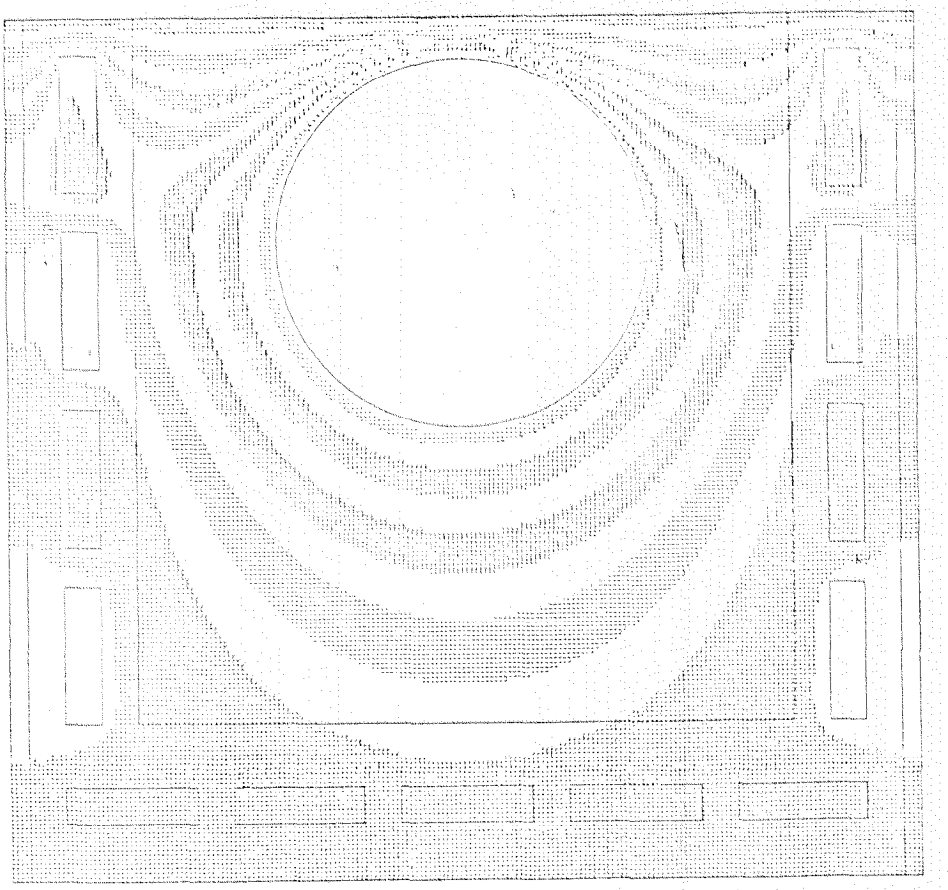
Temperatur am unteren Rand: 19.5°C

Zeitpunkt T=16h ohne Luft



Temperatur am unteren Rand: 18.1°C

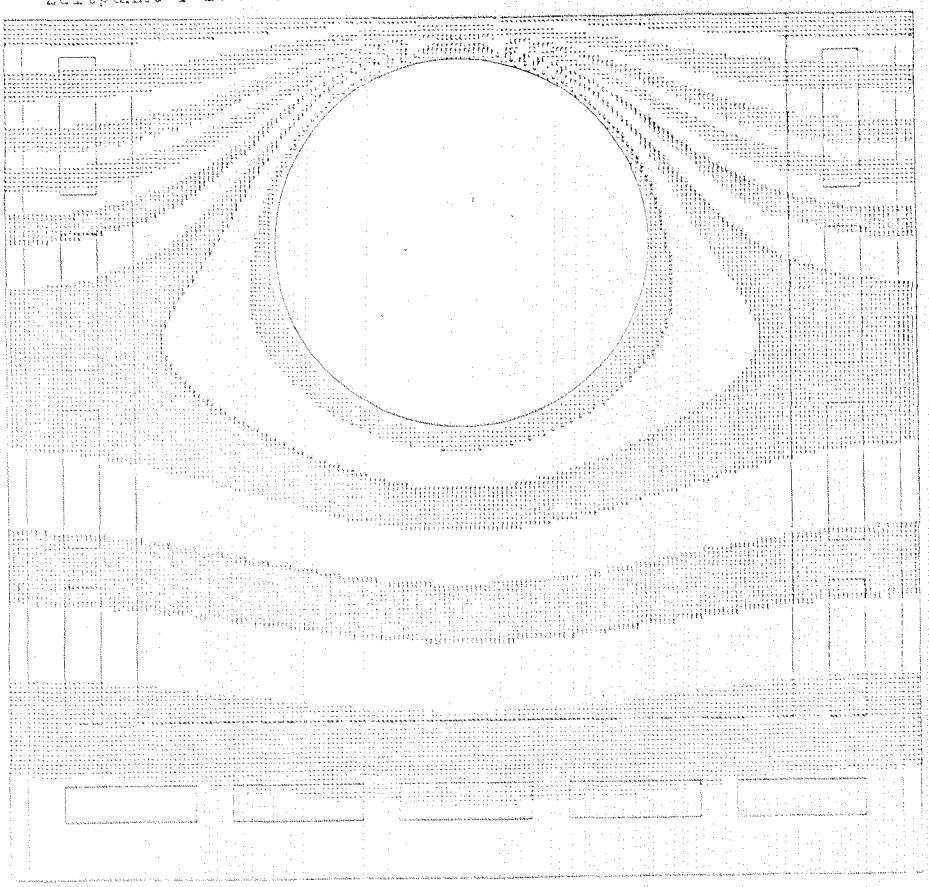
Zeitpunkt T=20h mit Luft



Temperatur am oberen Rand: 19.6°C

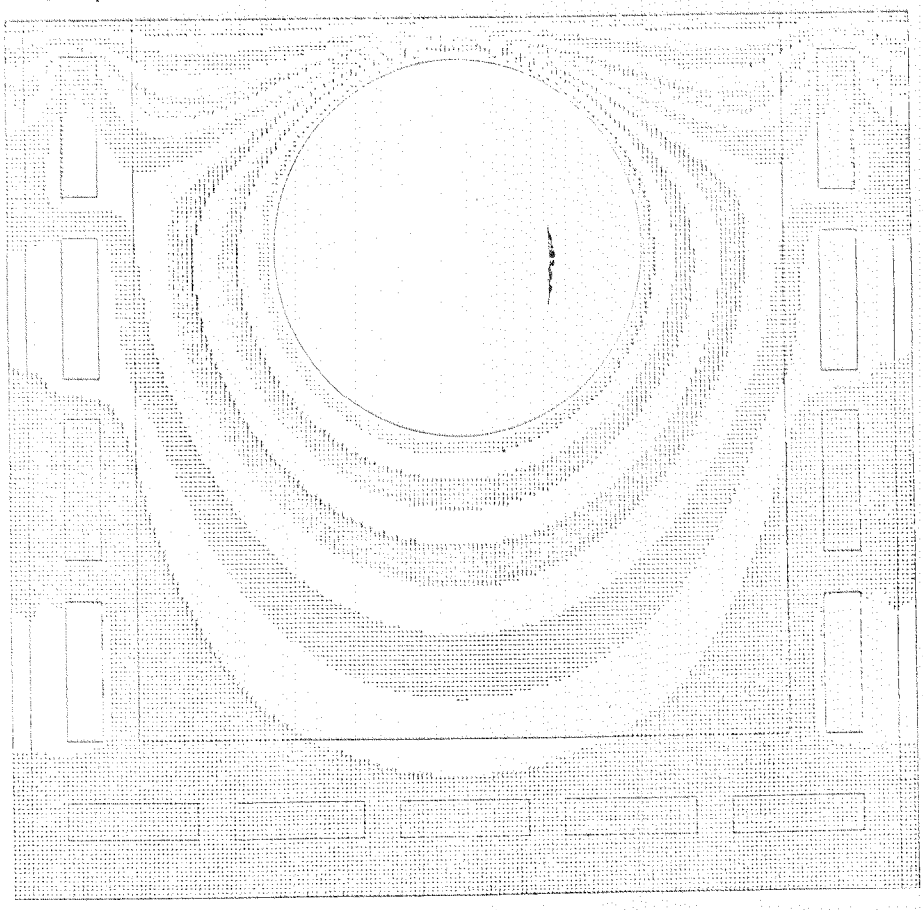
Temperatur am unteren Rand: 19.6°C

Zeitpunkt T=20h ohne Luft



Temperatur am unteren Rand: 18.1°C

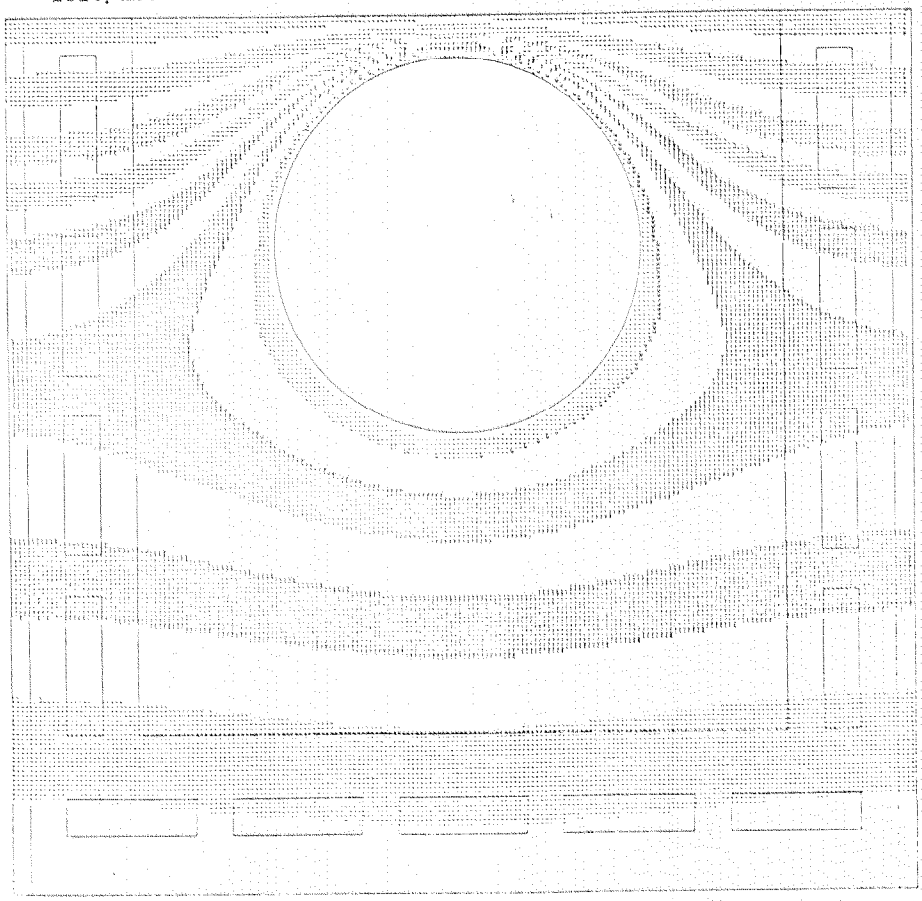
Zeitpunkt T=24h mit Luft



Temperatur am oberen Rand: 10.0 C

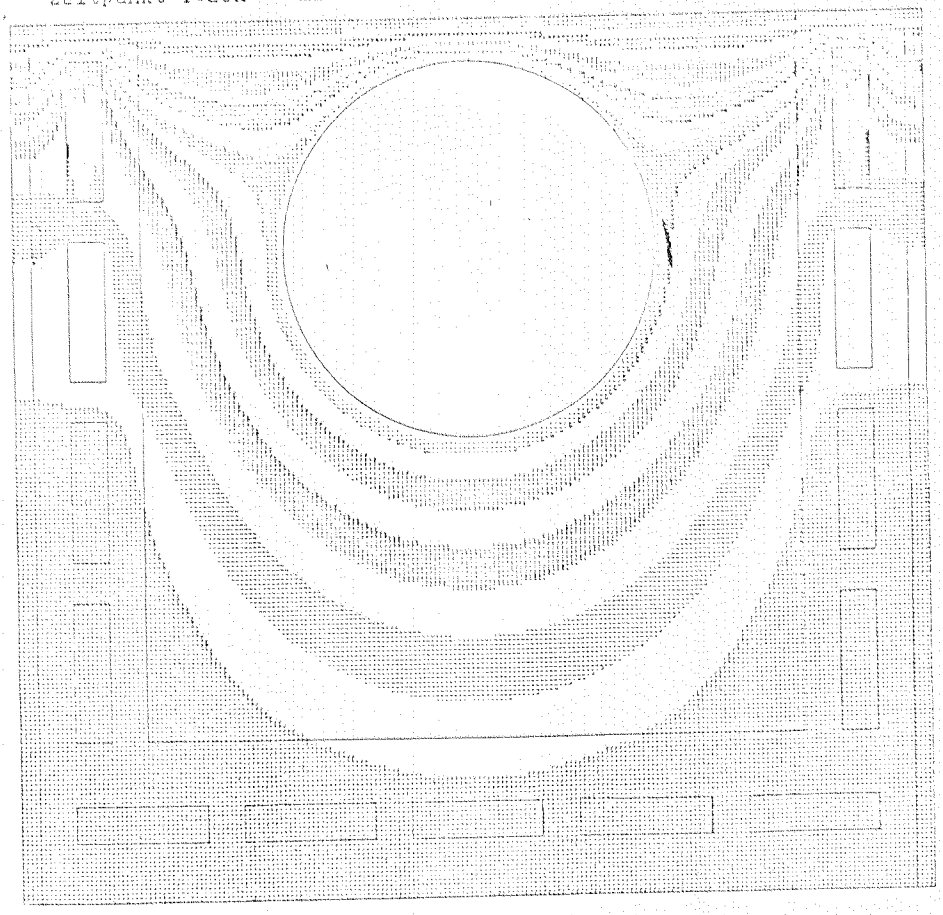
Temperatur am unteren Rand: 19.7°C

Zeitpunkt T=24h ohne Luft



Temperatur am unteren Rand: 18.2°C

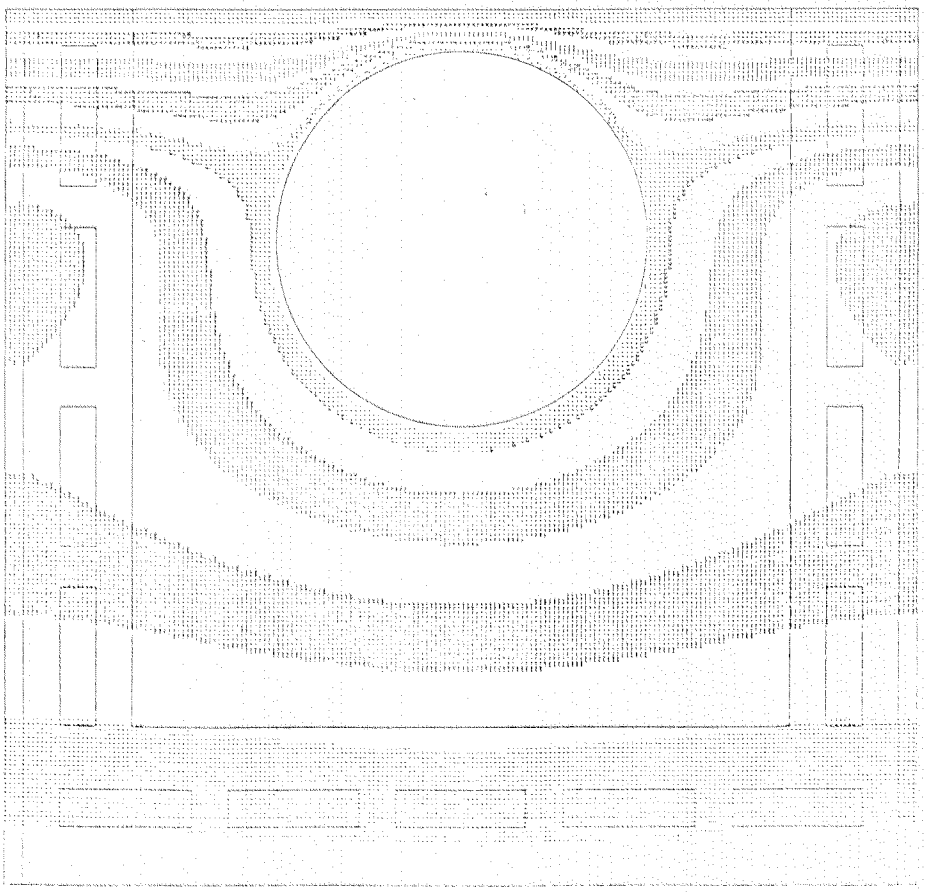
Zeitpunkt T=28h mit Luft



Temperatur am oberen Rand: 47.5°C

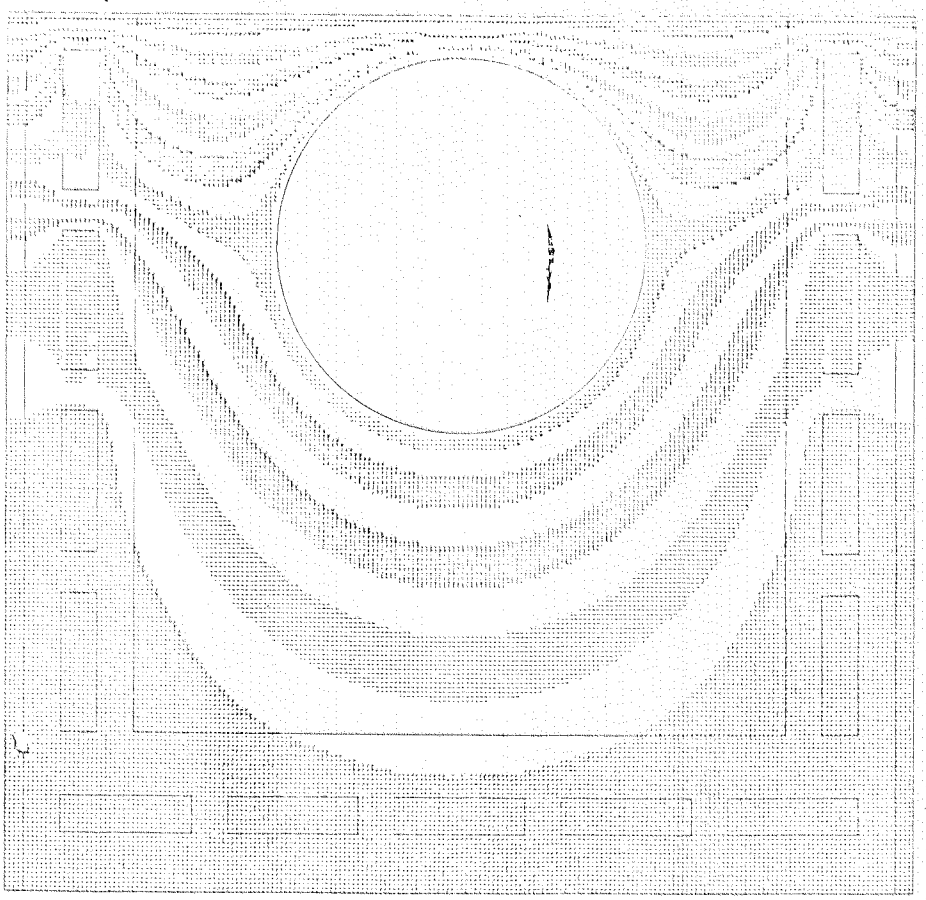
Temperatur am unteren Rand: 20.0°C

Zeitpunkt T=28h ohne Luft



Temperatur am unteren Rand: 20.1°C

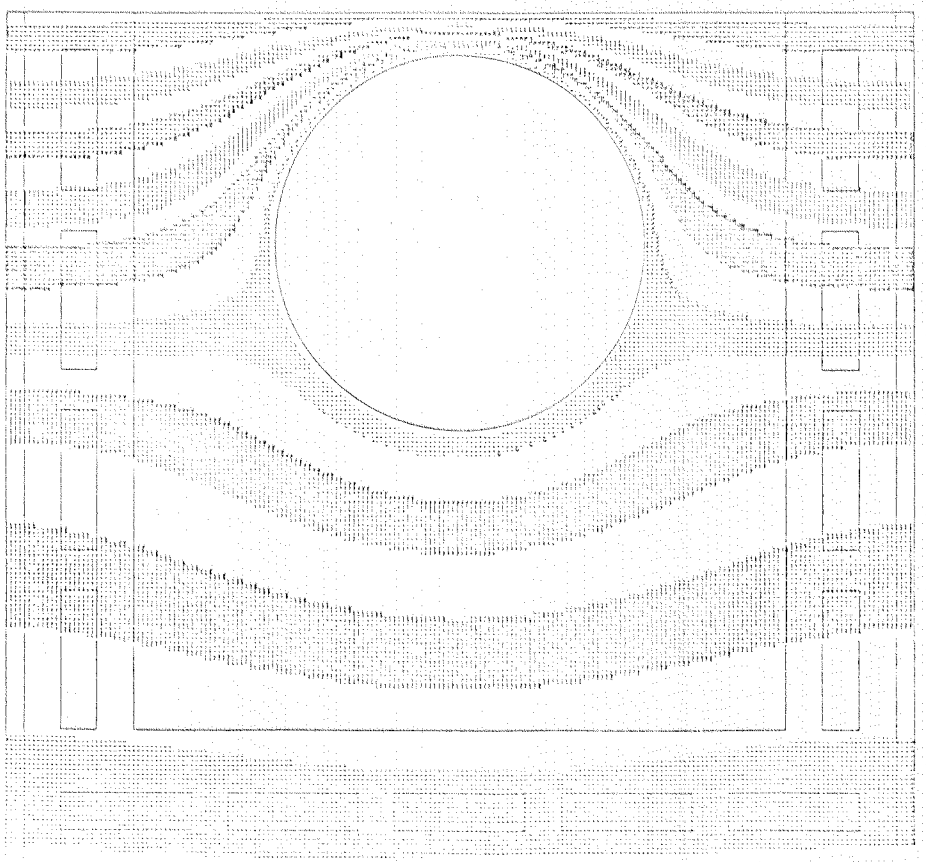
Zeitpunkt T=32h mit Luft



Temperatur am oberen Rand: 55.2°C

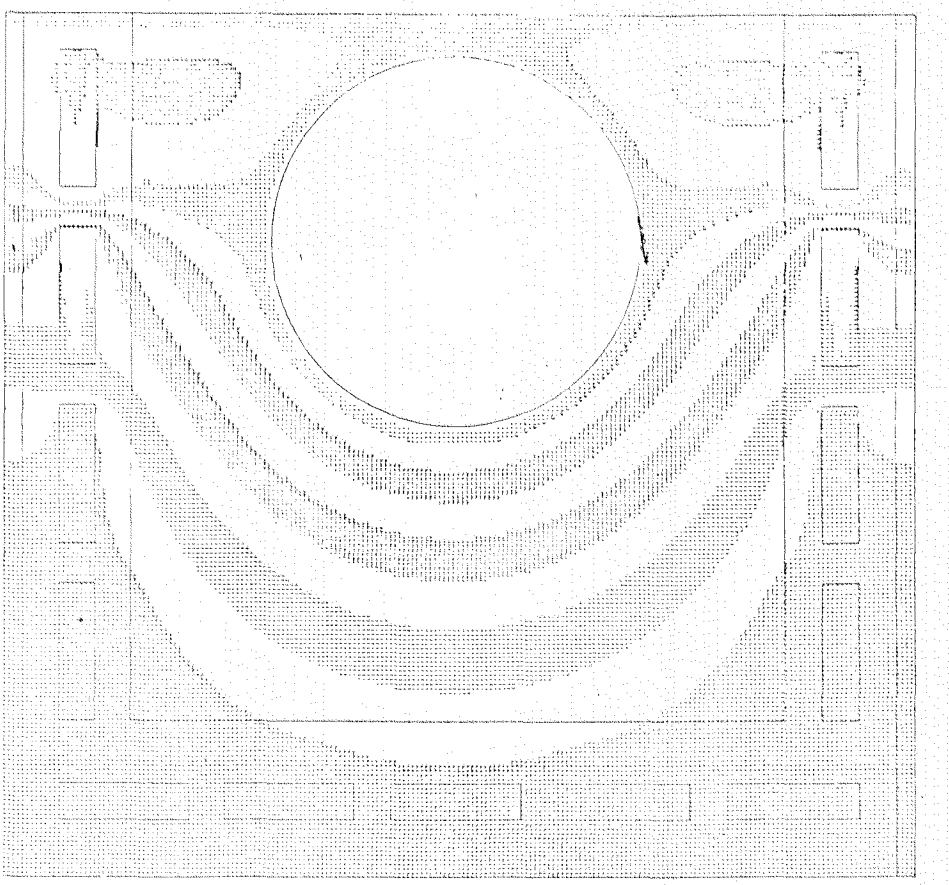
Temperatur am unteren Rand: 20.1°C

Zeitpunkt T=32h ohne Luft



Temperatur am unteren Rand: 18.3°C

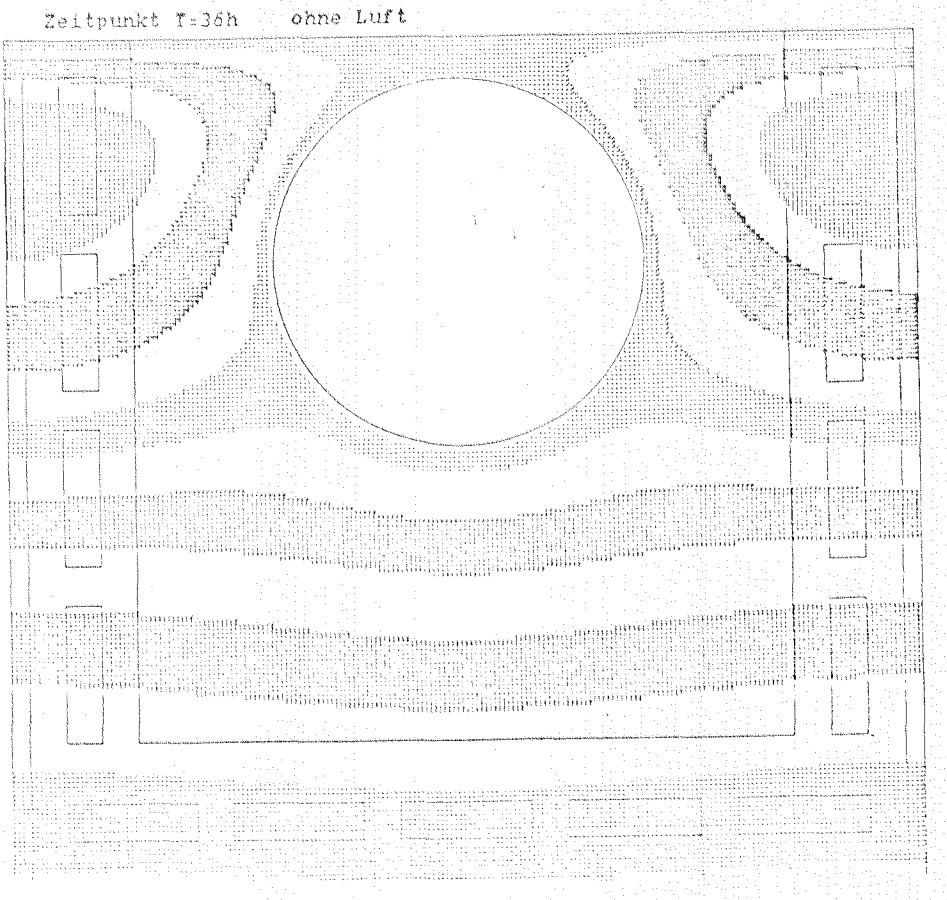
Zeitpunkt T=36h mit Luft



Temperatur am oberen Rand: 27.3°C

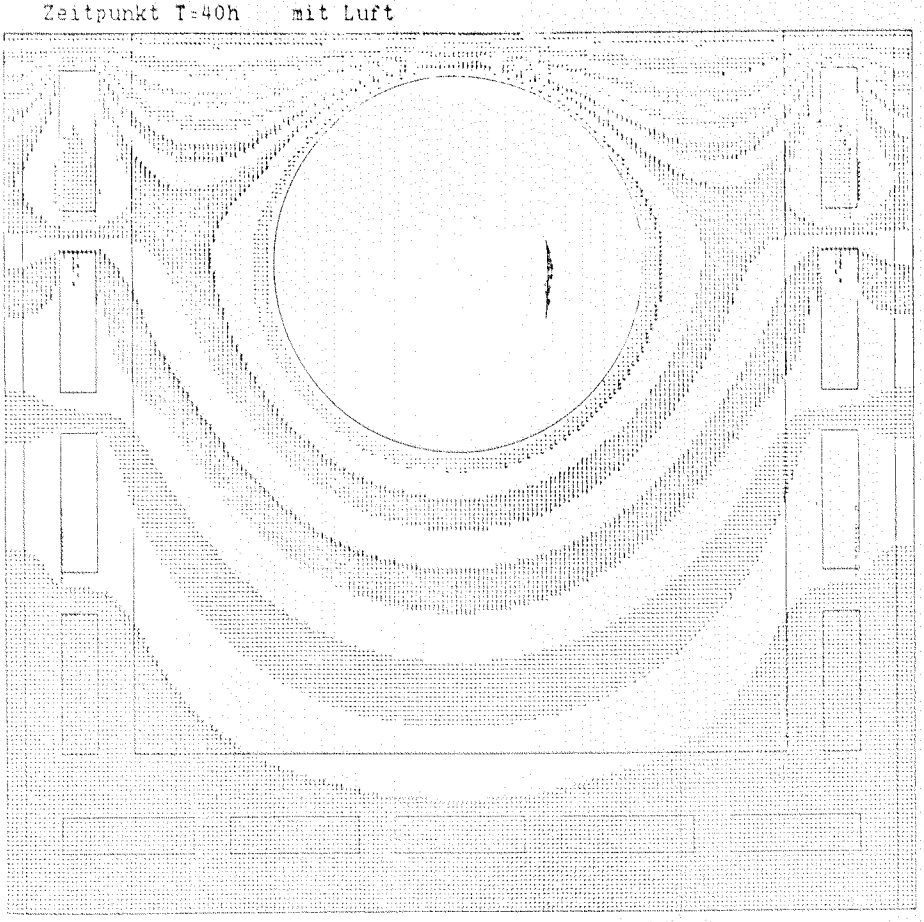
Temperatur am unteren Rand: 20.2°C

Zeitpunkt T=36h ohne Luft



Temperatur am unteren Rand: 18.4°C

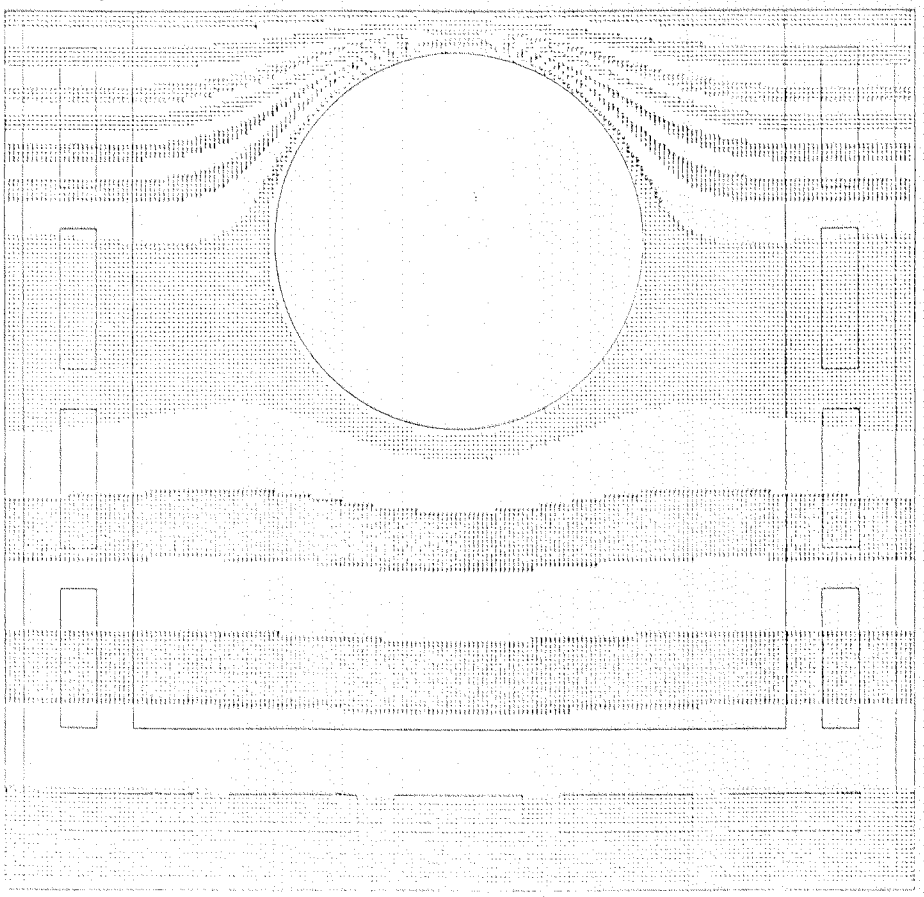
Zeitpunkt T=40h mit Luft



Temperatur am oberen Rand: 10.0°C

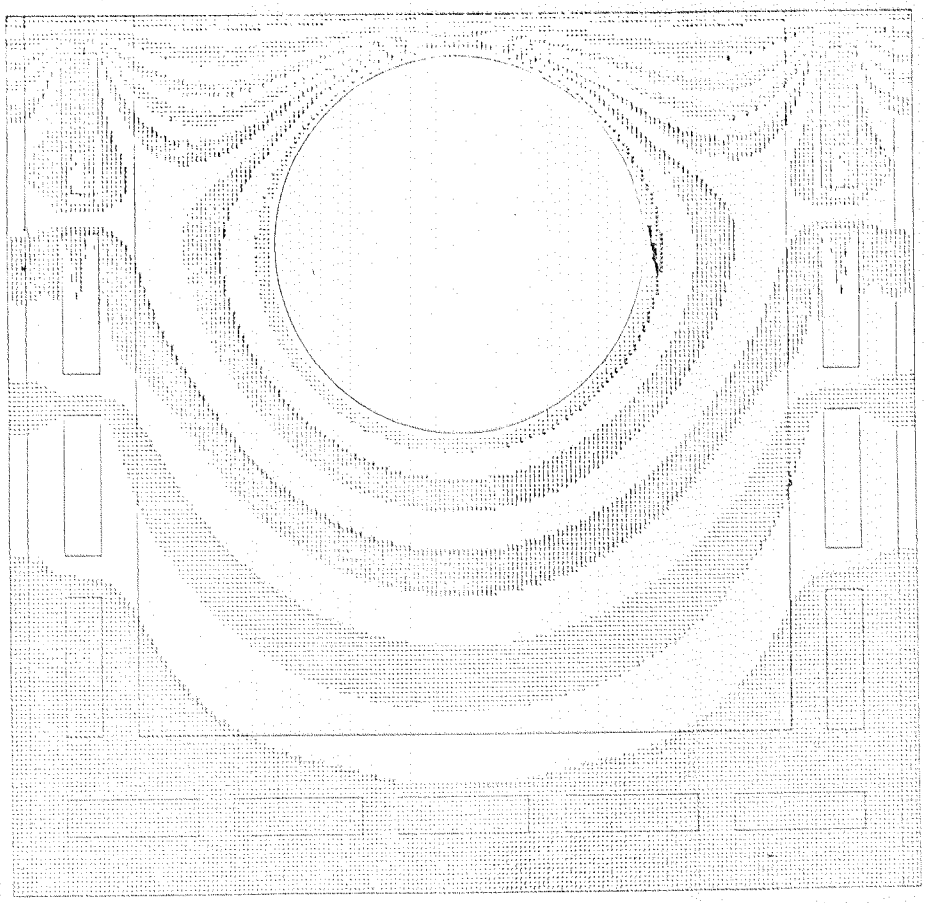
Temperatur am unteren Rand: 20.4°C

Zeitpunkt T=40h ohne Luft



Temperatur am unteren Rand: 18.4°C

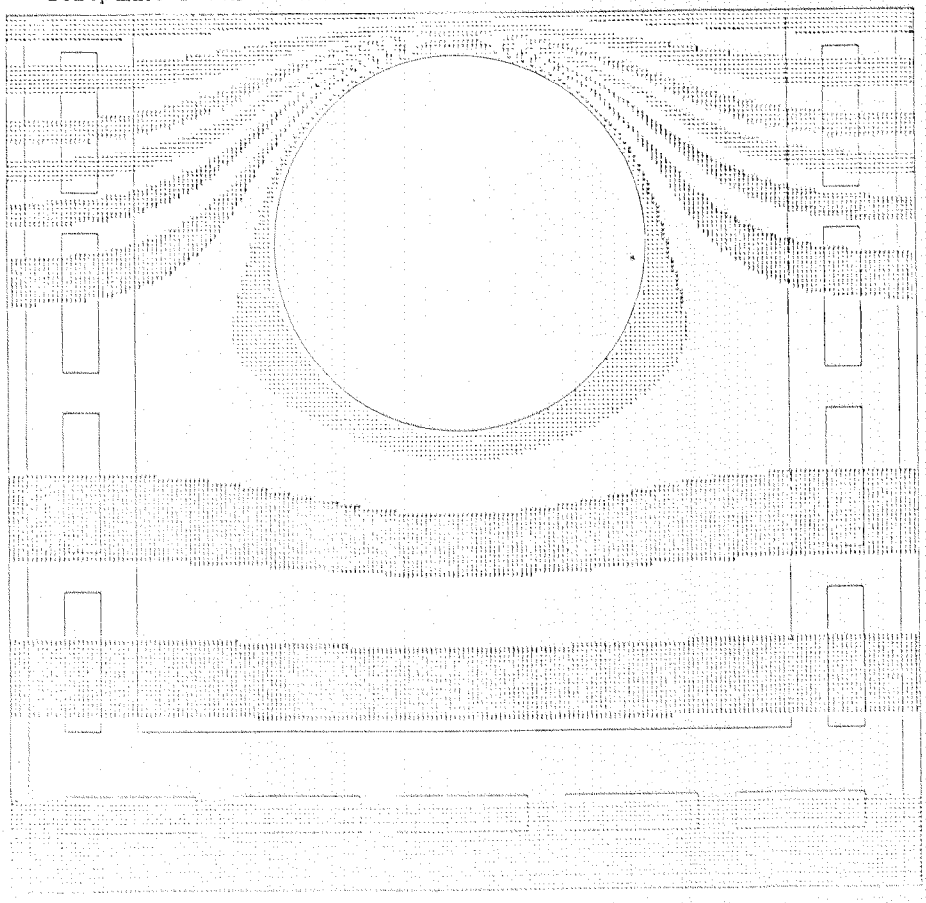
Zeitpunkt T=44h mit Luft



Temperatur am oberen Rand: 10.0°C

Temperatur am unteren Rand: 20.4°C

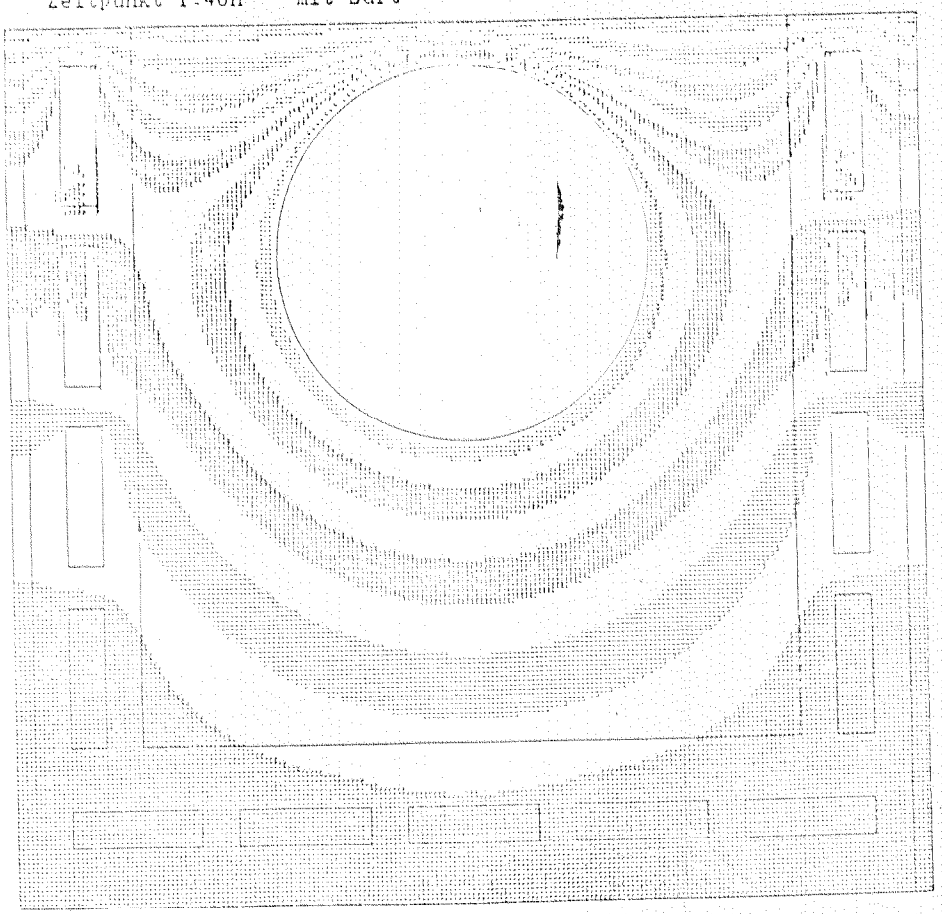
Zeitpunkt T=44h ohne Luft



Temperatur am oberen Rand: 10.0°C

Temperatur am unteren Rand: 18.5°C

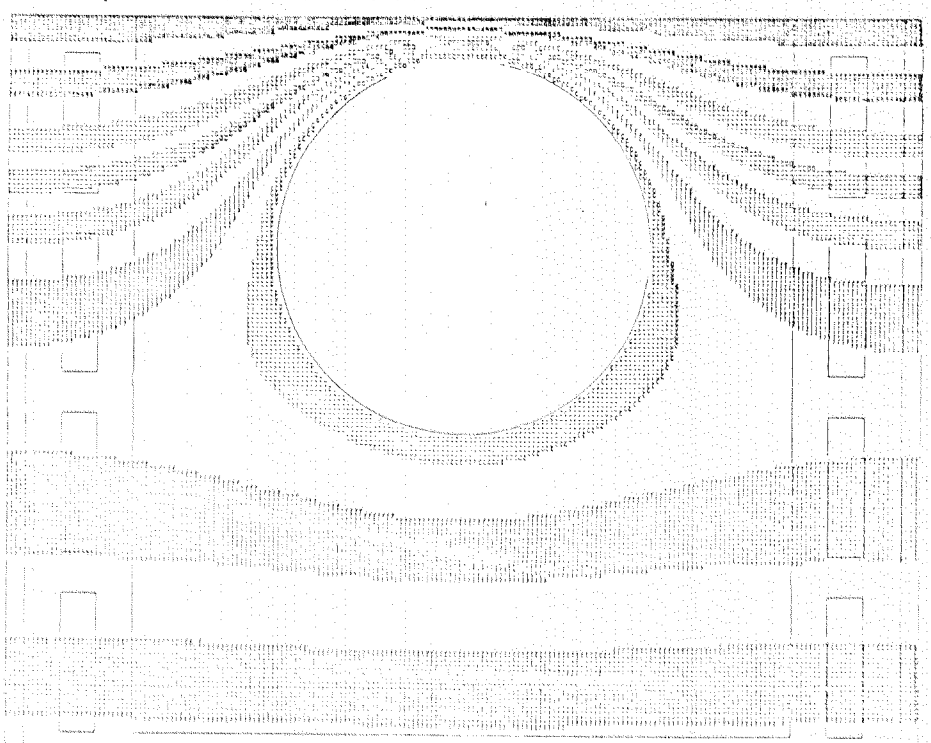
Zeitpunkt T=48h mit Luft



Temperatur am oberen Rand: 10.0°C

Temperatur am unteren Rand: 20.5°C

Zeitpunkt T=48h ohne Luft



Anhang

Programme

1) Programm **GEBIET**

2) Programm MATRIX

3) Programm CRNV

```
  1  1  0    (*$n+,d++++-,y+,h---,w--*)
  2 (* Datum: 26.05.1983*)
  3
  4   program  GEBIET (Input,Output,Position,Waerme );
  5   (* Berechnung der Zeilenlaenge in Abhaengigkeit von Delta *)
  6   (* und Vorbesetzen der Temperatur der Punkte C(I,J).      *)
  7
  8
  9
 10   const  r1=-1;                (* Oberer Rand     *)
 11          r2=-2;  trw2=27.0;    (* Kreis-Rand      *)
 12          r3=-3;                (* Unterer Rand    *)
 13          delta =0.25;          (* Schrittweite    *)
 14          w1=101;               (* (25 / delta)+1  *)
 15          w2=192;               (*  48 / delta     *)
 16
 17   type Breite = 0..w1;
 18        Hoehe  = 0..w2;
 19        Wert   = array [Breite,Hoehe] of real;
 20        Vektor = array [Hoehe]        of integer;
 21
 22   var  Position,Waerme: text;
 23        d,k,i,j,r,m,l,n: integer;
 24        z,b,y,trw1,trw3: real;
 25        v:               Vektor;
 26        t:               Wert;
 27
 28 procedure Zeile(var n:integer);
 29
 30 begin
 31   writeln (Position);
 32   writeln (Waerme);
 33   n:=1;
 34 end;

END ZEILE    L=    88

 35
 36
 37 begin
 38   rewrite (Position);
 39   rewrite (Waerme);
 40   d:=round(1/delta);
 41   writeln (' Bitte Eingabe der Temperatur am oberen Rand: ');
 42   read(trw1);
 43   writeln (' Bitte Eingabe der Temperatur am unteren Rand: ');
 44   read(trw3);
 45
 46
 47 (*Berechnung der Zeilenlaenge im Kreisbereich *)
 48
 49   l:=(25*d)+1;
 50   for j:=2*d to (trunc(12.5*d)) do
 51   begin
 52     y:=j*delta;
 53     b:=-(sqrt(sqr(10.5)-sqr(y+delta-12.5))-26);
 54     m:=trunc(b*d)-(d-1);
 55     if l>m then
 56     begin
```

```
57        v(.j.):=m;
58        v(.(25*d)-j.):=m;
59      end;
60      if l=m then
61      begin
62        v(.j.):=m-1;
63        v(.(25*d)-j.):=m-1;
64      end;
65      l:=m;
66    end;
67    for j:=0         to (2*d)-1 do v(.j.):=(25*d)+1;
68    for j:=(23*d)+1 to  w2      do v(.j.):=(25*d)+1;
69
70
71  (* Bestimmen der Startwerte *)
72
73    z:=(trw3-trw1)/w2;
74    for j:=1 to w2 do t(.1,j.):=trw1+(j*z);
75    for j:=2*d to (23*d) do
76    begin
77      z:=(trw2-t(.1,j.))/v(.j.);
78      for i:= 2 to v(.j.) do
79         t(.i,j.):=t(.1,j.)+((i-1)*z);
80    end;
81    for i:=2 to (25*d)+1 do
82    begin
83      for j:=0 to((2*d)-1) do
84      begin
85        if i>v(.2*d.) then t(.i,2*d.):=trw2;
86        z:=(trw1-t(.i,2*d.))/(2*d);
87        t(.i,j.):=trw1-(j*z);
88      end;
89      for j:=w2 downto ((23*d)+1) do
90      begin
91        if i>v(.23*d.) then t(.i,23*d.):=trw2;
92        z:=(trw3-t(.i,23*d.))/(w2-23*d);
93        t(.i,j.):=trw3-(w2-j)*z;
94      end;
95    end;
96
97
98  (* Berechnung oberhalb des Kreises *)
99
100   k:=1;
101   for j:=0 to (2*d)-1 do
102   begin
103     n:=0;
104     for i:=1 to (25*d)+1 do
105     begin
106       n:=n+1;
107       if n=27 then Zeile (n);    (*30*)
108       if j=0 then
109       begin
110         write (Position,r1:6);
111         write (Waerme,trw1:5:1);
112         t(.i,j.):=trw1;
113       end
114       else
```

```
115       begin
116         write (Position,k:6);
117         write (Waerme,t(.i,j.):5:1);
118         k:=k+1;
119       end;
120     end;
121     writeln (Position,-10:6);
122     writeln (Waerme,-10:5);
123   end;
124
125 (* Berechnung im Kreisbereich *)
126
127   for j:=(2*d) to (23*d) do
128   begin
129   n:=0;
130   if (v(.j-1.) > v(.j.)) then
131    begin
132      for i:=1 to v(.j-1.) do
133      begin
134        n:=n+1;
135        if n=27 then Zeile (n);   (*30*)
136        if i<=v(.j.) then
137        begin
138          write (Position,k:6);
139          write (Waerme,t(.i,j.):5:1);
140          k:=k+1;
141        end
142        else
143        begin
144          write (Position,r2:6);
145          write (Waerme,trw2:5:1);
146          t(.i,j.):=trw2;
147        end;
148      end;
149      writeln (Position,-10:6);
150      writeln (Waerme,-10:5);
151    end
152    else
153   If ((v(.j.)=v(.j-1.)) or (v(.j.)=v(.j+1.))) then
154    begin
155      for i:=1 to v(.j.)+1 do
156      begin
157        n:=n+1;
158        if n=27 then Zeile (n);   (*30*)
159        if i<=v(.j.) then
160        begin
161          write (Position,k:6);
162          write (Waerme,t(.i,j.):5:1);
163          k:=k+1;
164        end
165        else
166        begin
167          write (Position,r2:6);
168          write (Waerme,trw2:5:1);
169          t(.i,j.):=trw2;
170        end;
171      end;
172      writeln (Position,-10:6);
```

```
173       writeln (Waerme,-10:5);
174     end
175     else
176    if (v(.j+1.)>v(.j.)) then
177     begin
178       for i:=1 to v(.j+1.) do
179       begin
180         n:=n+1;
181         if n=27 then Zeile (n);    (*30*)
182         if i<=v(.j.) then
183         begin
184           write (Position,k:6);
185           write (Waerme,t(.i,j.):5:1);
186           k:=k+1;
187         end
188         else
189         begin
190           write (Position,r2:6);
191           write (Waerme,trw2:5:1);
192           t(.i,j.):=trw2;
193         end;
194       end;
195       writeln (Position,-10:6) ;
196       writeln (Waerme,-10:5);
197     end;
198   end;
199
200 (* Unter dem Kreisbereich *)
201
202   for j:= (23*d)+1 to w2 do
203   begin
204     n:=0;
205     for i:=1 to (25*d)+1 do
206     begin
207       n:=n+1;
208       if n=27 then Zeile (n);    (*30*)
209       if j=w2 then
210       begin
211         write (Position,r3:6);
212         write (Waerme,trw3:5:1);
213         t(.i,j.):=trw3;
214       end
215       else
216       begin
217         write (Position,k:6);
218         write (Waerme,t(.i,j.):5:1);
219         k:=k+1;
220       end;
221     end;
222     writeln (Position,-10:6);
223     writeln (Waerme,-10:5);
224   end;
225   writeln('  Die Anzahl der Matrinpunkte ist : ',k-1);
226 end.

END GEBIET    L=  7476
```

```
  1  1  0    (*$N+,D++++-,H---,L+-+,W--*)
  2
  3 Program MATRIXML (Input,Output,Position,Waerme,Resultat);
  4 (* Programm zur Erzeugung der Matrixpunkte *)
  5
  6 const k=16368;      (* Anzahl der Matrixpunkte                    *)
  7       delta=0.25;   (* Schrittweite in Zweierpotenzen             *)
  8                     (* in der Form Delta:=1/(2 hoch t)  t e N     *)
  9       l=101;        (* (25 / delta)+1                             *)
 10       w=192;        (*  48 / delta                                *)
 11       m=102;        (* l+1                                        *)
 12       eps=0.0009;   (* Genauigkeit                                *)
 13       na=99;        (* Anzahl der maximalen Iterationen           *)
 14       trw2=27;      (* Temperatur im Kreisbereich                 *)
 15
 16 type Anzahl = 1..k;
 17      Breite = 0..m;
 18      Hoehe  = 0..w;
 19      Wert   = array [Breite,Hoehe] of real;
 20      Vektor = array [Anzahl] of real;
 21      Punkt  = array [1..2,Anzahl] of integer;
 22
 23 var x,y,s,h,z,trw1,trw3,tt,zt,gs:    real;
 24     x1,y1,i,j,p,itn,n,d,t1,t2,t3,t4: integer;
 25     t:                               Wert;
 26     v1,v2,v3,v4,v6,b:                Vektor;
 27     v5:                              Punkt;
 28     Position, Waerme, Resultat:      Text;
 29
 30   (*   V1: Punkt ueber C(I,J)        *)
 31   (*   V2: Punkt links neben C(I,J)  *)
 32   (*   V3: Punkt rechs neben C(I,J)  *)
 33   (*   V4: Punkt unter C(I,J)        *)
 34   (*   V5,1: I                       *)
 35   (*   V5,2: J                       *)
 36   (*   V6: Summe V1 bis V4           *)
 37
 38
 39 procedure Zeile(var n: integer);
 40
 41   begin
 42     writeln (Waerme);
 43     n:=1;
 44   end;

END ZEILE     L=    74

 45
 46
 47 (* Prozedur zur Berechnung der Materialkonstanten *)
 48
 49 function Material (x,y: real): real;
 50
 51 const  D1=0.87;  (* Materialkonstante Moertel   *)
 52        D2=0.36;  (* Materialkonstante Ziegel    *)
 53        D3=0.10;  (* Materialkonstante Luft      *)
 54        D4=2.10;  (* Materialkonstante Feinbeton *)
 55
```

```
 56 (*%COPY MATERIAL,FMS.ZIEGEL,FMS*)
 57 ¢  begin
 58 ¢   if (x>0) and (x<2) and (y>0) and (y<48)
 59 ¢     then  Material:=D1                (*A:=1*)    else
 60 ¢   if (x=2) and (y>0) and (y<48)
 61 ¢     then  Material:=(D1+D2) / 2       (*A:=2*)    else
 62 ¢   if (x>2) and (x<4) and (y>0) and (y<48)
 63 ¢     then  Material:=D2                (*A:=3*)    else
 64 ¢   if ((x=4) or (x=6)) and ((( y>0) and  (y<2)) or
 65 ¢                            ((y>10) and (y<12)) or
 66 ¢                            ((y>20) and (y<22)) or
 67 ¢                            ((y>30) and (y<32)))
 68 ¢     then  Material:=D2                (*A:=4*)    else
 69 ¢   if ((x=4) or (x=6)) and ((( y>2) and (y<10)) or
 70 ¢                            ((y>12) and (y<20)) or
 71 ¢                            ((y>22) and (y<30)) or
 72 ¢                            ((y>32) and (y<39)))
 73 ¢     then  Material:=(D2+D3) / 2       (*A:=5*)    else
 74 ¢   if ((x>4) and (x<6) and ((y=2)  or
 75 ¢                           (y=10) or
 76 ¢                           (y=12) or
 77 ¢                           (y=20) or
 78 ¢                           (y=22) or
 79 ¢                           (y=30) or
 80 ¢                           (y=32) or
 81 ¢                           (y=39)))
 82 ¢     then  Material:=(D2+D3) / 2       (*A:=6*)    else
 83 ¢   if ((x>4) and (x<6) and ((( y>0) and (y<2) ) or
 84 ¢                            ((y>10) and (y<12)) or
 85 ¢                            ((y>20) and (y<22)) or
 86 ¢                            ((y>30) and (y<32))))
 87 ¢     then  Material:=D2                (*A:=7*)    else
 88 ¢   if ((x>4) and (x<6)) and (((y> 2) and (y<10)) or
 89 ¢                             ((y>12) and (y<20)) or
 90 ¢                             ((y>22) and (y<30)) or
 91 ¢                             ((y>32) and (y<39)))
 92 ¢     then  Material:=D3                (*A:=8*)    else
 93 ¢   if ((x>6) and (x<28)) and
 94 ¢      ((y>0) and (y<39))
 95 ¢     then  Material:=D2                (*A:=9*)    else
 96 ¢   if ((x=8) and (( 0<y) and (y<39)))
 97 ¢     then  Material:=(D2+D4) / 2      (*A:=10*)    else
 98 ¢   if ((x>8) and (x<27) and ( y>0) and (y<39))
 99 ¢     then  Material:=D4               (*A:=11*)    else
100 ¢   if ((x>8 )and (x<27) and (y=39))
101 ¢     then  Material:=(D2+D4) / 2      (*A:=12*)    else
102 ¢   if (((x>6) and (x<8)) or ((x>2) and (x<4))) and
103 ¢         (y=39)
104 ¢     then  Material:=D2               (*A:=13*)    else
105 ¢   if (((x= 4) or
106 ¢        (x=11) or
107 ¢        (x=13) or
108 ¢        (x=20) or
109 ¢        (x=22)) and ((y>43) and (y<45)))
110 ¢     then  Material:=(D2+D3) / 2      (*A:=14*)    else
111 ¢   if (((y>39) and (y<43)) or
112 ¢       ((y>45) and (y<48))) and
113 ¢       ((x> 2) and (x<27))
```

```
  114 ¢      then  Material:=D2                (*A:=15*)     else
  115 ¢    if  ((y>43) and (y<45)) and
  116 ¢       (((x> 4) and (x<11)) or
  117 ¢        ((x>13) and (x<20)) or
  118 ¢        ((x>22) and (x<27)))
  119 ¢      then  Material:=D3                (*A:=16*)     else
  120 ¢    if  ((y>43) and (y<45)) and
  121 ¢       (((x> 2) and (x< 4)) or
  122 ¢        ((x>11) and (x<13)) or
  123 ¢        ((x>20) and (x<22)))
  124 ¢      then  Material:=D2                (*A:=17*)     else
  125 ¢    if  ((y=43) or (y=45)) and
  126 ¢       (((x> 4) and (x<11)) or
  127 ¢        ((x>13) and (x<20)) or
  128 ¢        ((x>22) and (x<27)))
  129 ¢      then  Material:=(D2+D3) /2        (*A:=18*)     else
  130 ¢    if  ((y=43) or (y=45)) and
  131 ¢       (((x> 2) and (x< 4)) or
  132 ¢        ((x>11) and (x<13)) or
  133 ¢        ((x>20) and (x<22)))
  134 ¢      then  Material:=D2                (*A:=19*)     else
  135 ¢    if (y>48) and ((x> 0) and (x<27))
  136 ¢      then  Material:=D3                (*A:=20*)     else
  137 ¢    if (y=48) and ((x> 0) and (x< 2))
  138 ¢      then  Material:=(D1+D3) / 2       (*A:=21*)     else
  139 ¢    if (y=48) and ((x>2) and (x<27))
  140 ¢      then  Material:=(D2+D3) / 2       (*A:=22*)     else
  141 ¢      begin
  142 ¢       writeln(' ACHTUNG: Koeffizient falsch berechnet!!  ');
  143 ¢       writeln(' Koordinaten des Punktes: X:',x:5:2,'  Y:',y:5:2);
  144 ¢      end;
  145 ¢ end;

END MATERIAL L=  4214

  146 ¢
  147 (*%COPY AUSGABE,FMS.ZIEGEL,FMS*)
  148 ¢ (* Ausgabeprozedur *)
  149 ¢
  150 ¢ function Ausgabe (x:real):char;
  151 ¢
  152 ¢ var y:integer;
  153 ¢
  154 ¢ begin
  155 ¢   y:=round(x*100);
  156 ¢   if (y>=1000) and (y<1200) then Ausgabe:='A' else
  157 ¢   if (y>=1400) and (y<1600) then Ausgabe:='B' else
  158 ¢   if (y>=1800) and (y<1950) then Ausgabe:='C' else
  159 ¢   if (y>=2050) and (y<2150) then Ausgabe:='D' else
  160 ¢   if (y>=2250) and (y<2350) then Ausgabe:='E' else
  161 ¢   if (y>=2450) and (y<2550) then Ausgabe:='F' else
  162 ¢   if (y>=2650) and (y<2750) then Ausgabe:='G' else
  163 ¢   if (y>=2900) and (y<3150) then Ausgabe:='H' else
  164 ¢   if (y>=3300) and (y<3610) then Ausgabe:='I' else
  165 ¢   if (y>=3900) and (y<4210) then Ausgabe:='K' else
  166 ¢   if (y>=4500) and (y<4810) then Ausgabe:='L' else
  167 ¢   if (y>=5100) and (y<5410) then Ausgabe:='M' else
  168 ¢   if (y>=5700) and (y<6010) then Ausgabe:='O' else
```

```
  169 ¢                                        Ausgabe:=' ';
  170 ¢ end;

END AUSGABE  L=  1352

  171 ¢
  172
  173 (* Maximum *)
  174
  175 function Max (x,y:real):real;
  176
  177 begin
  178   if x>=y then Max:=x
  179           else Max:=y;
  180 end;

END MAX      L=   100

  181
  182
  183
  184 (* Bestimmen der C[I,J] aus K *)
  185
  186 begin
  187   d:=round(1/delta);
  188   writeln (' D:',d:1);
  189   writeln (' Bestimmung der C[I,J]  ');
  190   reset (Position);
  191   j:=0;
  192   i:=1;
  193   while not eof(Position) do
  194   begin
  195     while not eoln(Position) do
  196     begin
  197       read (Position,p);
  198       if p=-10 then
  199       begin
  200         j:=j+1;
  201         i:=1;
  202       end
  203       else
  204       begin
  205         if p>0 then
  206         begin
  207           v5[1,p]:=i;
  208           v5[2,p]:=j;
  209           i:=i+1;
  210         end;
  211       end;
  212     end;
  213     readln (Position);
  214   end;
  215
  216
```

```
217 (* Einlesen der Temperatur der C[I,J] *)
218
219   writeln (' Einlesen der Temperatur  ');
220   reset (Waerme);
221   j:=0;
222   i:=1;
223   while not eof(Waerme) do
224   begin
225     while not eoln(Waerme) do
226     begin
227       read (Waerme,z);
228       if z=-10 then
229       begin
230         i:=1;
231         j:=j+1;
232       end
233       else
234       begin
235         t[i,j]:=z;
236         i:=i+1;
237       end;
238     end;
239     readln(Waerme);
240   end;
241
242
243 (* Berechnung der Koeffizienten der C[I,J] *)
244
245   time (t1);
246   writeln (' Bestimmung der Koeffizienten  ');
247   for p:=1 to k do
248   begin
249     i:=v5[1,p];
250     j:=v5[2,p];
251     x:=(i-1)*delta + 1;
252     y:=j*delta;
253     v1[p]:=Material(x,y-delta/2);   (*57*)
254     if i=1 then
255       begin
256         v2[p]:=0;
257         v3[p]:=2*Material(x+delta/2,y);   (*57*)
258       end;
259     if i=l then
260       begin
261         v2[p]:=2*Material(x-delta/2,y);   (*57*)
262         v3[p]:=0;
263       end;
264     if (i<>1) and (i<>l) then
265       begin
266         v2[p]:=Material(x-delta/2,y);   (*57*)
267         v3[p]:=Material(x+delta/2,y);   (*57*)
268       end;
269     v4[p]:=Material(x+delta/2,y);   (*57*)
270     v6[p]:=v1[p]+v2[p]+v3[p]+v4[p];
271     end;
272   time (t2);
273   zt:=(t2-t1)/10000;
274   writeln (' Berechnung abgeschlossen, Zeit:',zt:7:4,'sec');
```

```
275
276
277 (* Randtemperatur setzen *)
278
279   writeln (' Bitte Eingabe der Temperatur am oberen Rand: ');
280   read (tt);
281   writeln (' Bitte Eingabe der Temperatur am unteren Rand: ');
282   read (trw3);
283   rewrite (Resultat);
284   for i:=1 to m do t[i,w]:=trw3;
285
286 (* Lauf für verscheidene Temperaturen *)
287
288   repeat
289 (*  for i:=1 to m do t[i,w]:=t[i,w-1];     Unterer Rand frei  *)
290     trw1:=tt;
291     time (t1);
292     for i:=0 to m do t[i,0]:=trw1;
293
294 (* Iteration *)
295
296     writeln (' Beginn der Iteration, Randtemperatur:',trw1:4:1);
297     itn:=0;
298     repeat
299       time (t3);
300       itn:=itn+1;
301       s:=0;
302       for p:=1 to k do
303       begin
304         i:=v5[1,p];
305         j:=v5[2,p];
306         h:=(v1[p]*t[i,j-1]+
307             v2[p]*t[i-1,j]+
308             v3[p]*t[i+1,j]+
309             v4[p]*t[i,j+1])/v6[p];
310         s:=Max(s,abs(h-t[i,j]));   (*177*)
311         t[i,j]:=h;
312       end;
313       time (t4);
314       zt:=(t4-t3)/10000;
315       writeln ('  ',itn:2,'.Iteration  S:',s:5:3,' Zeit:',zt:7:4,'sec
316     until (s<=eps) or (itn>=na);
317     time (t2);
318     gs:=(t2-t1)/10000;
319     writeln (' Ende der Iteration, Gesamtzeit:',gs:7:4,'sec');
320
321
322
323
324 (* Ausgabe der Temperatur *)
325
326     writeln (' Ausgabe der Temperatur ');
327     writeln (Resultat,'A');
328     for i:=0 to 6 do writeln (Resultat,'  ');
329     writeln (Resultat,'   TEMPERATUR AM OBEREN RAND: ',trw1:3:1,
330                       ' GRAD                                    ');
331     writeln (Resultat,'   TEMPERATUR AM UNTEREN RAND: ',trw3:3:1,
332                       ' GRAD                                    ');
```

```
333      write   (Resultat,'   GLEICHGEWICHTSLOESUNG ');
334      writeln (Resultat,'                                          ');
335      for i:=0 to 3 do writeln (Resultat,'  ');
336      writeln (Resultat,'  *');
337      for j:=0 to w do
338      begin
339        write (Resultat,'  ');
340        for i:=1 to l do
341          write (Resultat,Ausgabe(t[i,j]));   (*154*)
342      end;
343      writeln (Resultat,'  *');
344
345
346      writeln (' Bitte Eingabe der Temperatur am oberen Rand, ');
347      writeln (' 0 wenn Programm beendet werden soll:         ');
348      read (tt)
349    until tt=0;
350
351    (* Ausgabe auf Datei Wärme *)
352
353    rewrite (Waerme);
354    writeln (' Neue Startwerte ausgeben ');
355    for j:=0 to w do
356    begin
357      n:=0;
358      for i:=1 to l do
359      begin
360        n:=n+1;
361        if n=27 then Zeile(n);   (*41*)
362        write (Waerme,t[i,j]:5:1);
363      end;
364      writeln (Waerme,-10:5);
365    end;
366
367
368    writeln (' Programmende ');
369
370  end.
```

END MATRIXML L= 6254

```
 1  1  0   (*$N+,D++++-,H---,L+-+,W--*)
 2
 3 Program CRNVML (Input,Output,Position,Waerme,Resultat);
 4 (* Programm zur Berechnung der Temperaturverteilung *)
 5
 6 const k=16368;      (* Anzahl der zu berechnenden Punkte       *)
 7       kk=16469;     (* k+l                                     *)
 8       delta=0.25;   (* Schrittweite in Zweierpotenzen          *)
 9                     (* in der Form Delta:=1/(2 hoch t)  t e N  *)
10       l=101;        (* (25 / delta)+1                          *)
11       m=102;        (* l+1                                     *)
12       w=192;        (* 48 / delta                              *)
13       ww=193;       (* w+1                                     *)
14       trw2=27;      (* Temperatur im Kreisbereich in °Celsius  *)
15       eps=0.005;    (* Abbruchschranke der Iteration           *)
16       omega=0.90;   (* Überrelaxationsparameter                *)
17       pi=3.1416;    (*                                         *)
18
19 type Anzahl = 1..kk;
20      Breite = 0..m;
21      Hoehe  = 0..ww;
22      Zeit   = array [1..15]         of char;
23      Wert   = array [Breite,Hoehe] of real;
24      Punkt  = array [1..2,Anzahl]  of integer;
25      Zeile  = array [Breite]        of real;
26      Nummer = array [Breite,Hoehe] of integer;
27      Vektor = array [Anzahl]        of real;
28
29
30 var x,y,z,b,h,st1,trw1,trw3,deltaX,zt,s: real;
31     i,j,p,itn,d,hh,t1,t2,t3,t4,lauf,na1,na2,na3,deltaT: integer;
32     t:                                Wert;
33     uhr:                              Zeit;
34     c:                                Nummer;
35     nc:                               Zeile;
36     v1,v2,v3,v4,v6,v7:                Vektor;
37     v5:                               Punkt;
38     Position, Waerme, Resultat:       Text;
39
40   (*   V1: Materialkonstante über C(I,J)          *)
41   (*   V2: Materialkonstante links neben C(I,J)   *)
42   (*   V3: Materialkonstante rechs neben C(I,J)   *)
43   (*   V4: Materialkonstante unter C(I,J)         *)
44   (*   V5,1: I                                    *)
45   (*   V5,2: J                                    *)
46   (*   V6: V1+V2+V3+V4                            *)
47   (*   V7: Rechte Seite B                         *)
48   (*    b: deltaT / 2*(deltaX**2)                 *)
49   (*   deltaX: delta*0.005 in m                   *)
50
51 (* Prozedur zur Berechnung der Materialkonstanten *)
52
53 function Material (x,y: real): real;
54
55 const  D1=0.543E-06;(* Materialkonstante Mörtel     *)
56        D2=0.285E-06;(* Materialkonstante Ziegel     *)
57        D3=1.783E-05;(* Materialkonstante Luft       *)
58        D4=0.875E-06;(* Materialkonstante Feinbeton  *)
59                     (* Einheit  (m**2) / sec        *)
```

```
(*%COPY MATERIAL,FMS.ZIEGEL,FMS*)
 begin
  if (x>0) and (x<2) and (y>0) and (y<48)
    then  Material:=D1                (*A:=1*)     else
  if (x=2) and (y>0) and (y<48)
    then  Material:=(D1+D2) / 2       (*A:=2*)     else
  if (x>2) and (x<4) and (y>0) and (y<48)
    then  Material:=D2                (*A:=3*)     else
  if ((x=4) or (x=6)) and ((( y>0) and  (y<2)) or
                           ((y>10) and (y<12)) or
                           ((y>20) and (y<22)) or
                           ((y>30) and (y<32)))
    then  Material:=D2                (*A:=4*)     else
  if ((x=4) or (x=6)) and ((( y>2) and (y<10)) or
                           ((y>12) and (y<20)) or
                           ((y>22) and (y<30)) or
                           ((y>32) and (y<39)))
    then  Material:=(D2+D3) / 2       (*A:=5*)     else
  if ((x>4) and (x<6) and ((y=2)  or
                          (y=10) or
                          (y=12) or
                          (y=20) or
                          (y=22) or
                          (y=30) or
                          (y=32) or
                          (y=39)))
    then  Material:=(D2+D3) / 2       (*A:=6*)     else
  if ((x>4) and (x<6) and ((( y>0) and (y<2) ) or
                           ((y>10) and (y<12)) or
                           ((y>20) and (y<22)) or
                           ((y>30) and (y<32))))
    then  Material:=D2                (*A:=7*)     else
  if ((x>4) and (x<6)) and (((y> 2) and (y<10)) or
                            ((y>12) and (y<20)) or
                            ((y>22) and (y<30)) or
                            ((y>32) and (y<39)))
    then  Material:=D3                (*A:=8*)     else
  if ((x>6) and (x<28)) and
     ((y>0) and (y<39))
    then  Material:=D2                (*A:=9*)     else
  if ((x=8) and (( 0<y) and (y<39)))
    then  Material:=(D2+D4) / 2       (*A:=10*)    else
  if ((x>8) and (x<27) and ( y>0) and (y<39))
    then  Material:=D4                (*A:=11*)    else
  if ((x>8 )and (x<27) and (y=39))
    then  Material:=(D2+D4) / 2       (*A:=12*)    else
  if (((x>6) and (x<8)) or ((x>2) and (x<4))) and
        (y=39)
    then  Material:=D2                (*A:=13*)    else
  if (((x= 4) or
       (x=11) or
       (x=13) or
       (x=20) or
       (x=22)) and ((y>43) and (y<45)))
    then  Material:=(D2+D3) / 2       (*A:=14*)    else
  if (((y>39) and (y<43)) or
      ((y>45) and (y<48))) and
```

```
 118 ¢        ((x> 2) and (x<27))
 119 ¢      then  Material:=D2                (*A:=15*)    else
 120 ¢    if  ((y>43) and (y<45)) and
 121 ¢       (((x> 4) and (x<11)) or
 122 ¢        ((x>13) and (x<20)) or
 123 ¢        ((x>22) and (x<27)))
 124 ¢      then  Material:=D3                (*A:=16*)    else
 125 ¢    if  ((y>43) and (y<45)) and
 126 ¢       (((x> 2) and (x< 4)) or
 127 ¢        ((x>11) and (x<13)) or
 128 ¢        ((x>20) and (x<22)))
 129 ¢      then  Material:=D2                (*A:=17*)    else
 130 ¢    if  ((y=43) or (y=45)) and
 131 ¢       (((x> 4) and (x<11)) or
 132 ¢        ((x>13) and (x<20)) or
 133 ¢        ((x>22) and (x<27)))
 134 ¢      then  Material:=(D2+D3) /2        (*A:=18*)    else
 135 ¢    if  ((y=43) or (y=45)) and
 136 ¢       (((x> 2) and (x< 4)) or
 137 ¢        ((x>11) and (x<13)) or
 138 ¢        ((x>20) and (x<22)))
 139 ¢      then  Material:=D2                (*A:=19*)    else
 140 ¢    if (y>48) and ((x> 0) and (x<27))
 141 ¢      then  Material:=D3                (*A:=20*)    else
 142 ¢    if (y=48) and ((x> 0) and (x< 2))
 143 ¢      then  Material:=(D1+D3) / 2       (*A:=21*)    else
 144 ¢    if (y=48) and ((x>2) and (x<27))
 145 ¢      then  Material:=(D2+D3) / 2       (*A:=22*)    else
 146 ¢      begin
 147 ¢       writeln(' ACHTUNG: Koeffizient falsch berechnet!!  ');
 148 ¢       writeln(' Koordinaten des Punktes: X:',x:5:2,'  Y:',y:5:2);
 149 ¢      end;
 150 ¢ end;

END MATERIAL L=  4214

 151 ¢
 152 (*%COPY AUSGABE,FMS.ZIEGEL,FMS*)
 153 ¢ (* Ausgabeprozedur *)
 154 ¢
 155 ¢ function Ausgabe (x:real):char;
 156 ¢
 157 ¢ var y:integer;
 158 ¢
 159 ¢ begin
 160 ¢   y:=round(x*100);
 161 ¢   if (y>=1000) and (y<1200) then Ausgabe:='A' else
 162 ¢   if (y>=1400) and (y<1600) then Ausgabe:='B' else
 163 ¢   if (y>=1800) and (y<1950) then Ausgabe:='C' else
 164 ¢   if (y>=2050) and (y<2150) then Ausgabe:='D' else
 165 ¢   if (y>=2250) and (y<2350) then Ausgabe:='E' else
 166 ¢   if (y>=2450) and (y<2550) then Ausgabe:='F' else
 167 ¢   if (y>=2650) and (y<2750) then Ausgabe:='G' else
 168 ¢   if (y>=2900) and (y<3150) then Ausgabe:='H' else
 169 ¢   if (y>=3300) and (y<3610) then Ausgabe:='I' else
 170 ¢   if (y>=3900) and (y<4210) then Ausgabe:='K' else
 171 ¢   if (y>=4500) and (y<4810) then Ausgabe:='L' else
 172 ¢   if (y>=5100) and (y<5410) then Ausgabe:='M' else
```

```
  173 ¢    if (y>=5700) and (y<6010) then Ausgabe:='0' else
  174 ¢                                   Ausgabe:=' ';
  175 ¢ end;

END AUSGABE  L=  1352

  176 ¢
  177
  178 (* Maximum *)
  179
  180 function Max (x,y:real):real;
  181
  182 begin
  183   if x>=y then Max:=x
  184           else Max:=y;
  185 end;

END MAX      L=   100

  186
  187
  188 begin
  189   d:=round(1/delta);
  190   writeln (' Bitte Eingabe der Temperatur am oberen Rand: ');
  191   read (st1);
  192   writeln (' Bitte Eingabe der Lufttemperatur im Raum: ');
  193   read (trw3);
  194   writeln (' Bitte Eingabe der maximalen Iterationsschritte: ');
  195   read (na1);
  196   writeln (' Bitte Eingabe der Zeitschrittweite: ');
  197   read (deltaT);
  198   writeln (' Ueber wieviele Stunden soll der Versuch laufen? ');
  199   read (na2);
  200   deltaX:=delta*0.005;
  201   na2:=round((na2*3600) / deltaT);
  202   na3:=round(3600/deltaT);
  203   writeln (' DELTA T:',deltaT:4,' sec ');
  204   writeln (' DELTA X:',deltaX:7:5,' m   ');
  205   b:=deltaT / (2*deltaX*deltaX);
  206   writeln (' B: ',b);
  207   writeln (' Genauigkeit EPS: ',eps:6:4);
  208   for i:=0 to m do
  209   begin
  210     for j:=0 to ww do
  211     begin
  212       c[i,j]:=0;
  213       t[i,j]:=0;
  214     end;
  215   end;
  216
  217 (* Bestimmen der C[I,J] aus der Nummer K *)
  218
  219   writeln (' Bestimmung der C[I,J]  ');
  220   reset (Position);
  221   j:=0;
  222   i:=1;
  223   while not eof(Position) do
  224   begin
```

```
225      while not eoln(Position) do
226      begin
227        read (Position,p);
228        if p=-10 then
229        begin
230          j:=j+1;
231          i:=1;
232        end
233        else
234        begin
235          if p>0 then
236          begin
237            v5[1,p]:=i;
238            v5[2,p]:=j;
239          end;
240          c[i,j]:=p;
241          i:=i+1;
242        end;
243      end;
244      readln (Position);
245    end;
246    for i:=1 to l do
247    begin
248      p:=k+i;
249      c[i,w]:=p;
250      v5[1,p]:=i;
251      v5[2,p]:=w;
252    end;
253    for i:=1 to l do c[i,ww]:=-4;
254
255
256
257  (* Berechnung der Koeffizienten der C[I,J] *)
258
259    time (t1);
260    writeln (' Bestimmung der Koeffizienten  ');
261    for p:=1 to k do
262    begin
263      i:=v5[1,p];
264      j:=v5[2,p];
265      x:=(i-1)*delta + 1;
266      y:=j*delta;
267      v1[p]:=b*Material(x,y-delta/2);    (*62*)
268      if i=1 then
269        begin
270          v2[p]:=0;
271          v3[p]:=2*b*Material(x+delta/2,y);    (*62*)
272        end;
273      if i=l then
274        begin
275          v2[p]:=2*b*Material(x-delta/2,y);    (*62*)
276          v3[p]:=0;
277        end;
278      if (i<>1) and (i<>l) then
279        begin
280          v2[p]:=b*Material(x-delta/2,y);    (*62*)
281          v3[p]:=b*Material(x+delta/2,y);    (*62*)
282        end;
```

```
283       v4[p]:=b*Material(x,y+delta/2);   (*62*)
284       v6[p]:=v1[p]+v2[p]+v3[p]+v4[p];
285
286       (*%COPY EQUILIB,FMS.ZIEGEL,FMS*)
287 ¢     v1[p]:=v1[p] / (1+v6[p]);
288 ¢     v2[p]:=v2[p] / (1+v6[p]);
289 ¢     v3[p]:=v3[p] / (1+v6[p]);
290 ¢     v4[p]:=v4[p] / (1+v6[p]);
291 ¢
292     end;
293     for p:=(k+1) to kk do
294     begin
295       v1[p]:=0;
296       v2[p]:=0;
297       v3[p]:=0;
298       v4[p]:=0;
299       v6[p]:=0;
300       v7[p]:=0;
301     end;
302
303     time (t2);
304     zt:=(t2-t1)/10000;
305     writeln (' Berechnung abgeschlossen, Zeit:',zt:7:4,'sec');
306
307
308 (* Einlesen der Temperatur der C[I,J] *)
309
310     writeln (' Einlesen der Temperatur  ');
311     reset (Waerme);
312     j:=0;
313     i:=1;
314     while not eof(Waerme) do
315     begin
316       while not eoln(Waerme) do
317       begin
318         read (Waerme,z);
319         if z=-10 then
320         begin
321           i:=1;
322           j:=j+1;
323         end
324         else
325         begin
326           t[i,j]:=z;
327           i:=i+1;
328         end;
329       end;
330       readln(Waerme);
331     end;
332
333 (* Ausgabe vorbereiten *)
334
335     rewrite (Resultat);
336     date (uhr);
337     writeln (Resultat,'A');
338     for p:=1 to 10 do writeln (Resultat,'  ');
339     writeln (Resultat,'                    Programm  C R N V.EQ ML ');
340     writeln (Resultat,'  ');
```

```
341   writeln (Resultat,'                    Lauf: ',uhr[4],uhr[5],'.',uhr[1],
342   uhr[2],'.1983       Uhrzeit: ',uhr[11],uhr[12],':',uhr[14],uhr[15]
343   writeln (Resultat,'  ');
344   writeln (Resultat,'                    Lufttemperatur:',trw3:4:1,'°C');
345   writeln (Resultat,'  ');
346   writeln (Resultat,'                    EPS: ',eps:6:4,'   Omega: ',omega:
347   lauf:=0;
348
349   REPEAT
350
351   time (t1);
352   lauf:=lauf+1;
353
354   (*%COPY TAGL,FMS.ZIEGEL,FMS*)
355 ¢   hh:=(lauf*deltaT) mod 86400;
356 ¢   if ((hh>=0) and (hh<21600))
357 ¢   then begin
358 ¢          h:=(pi/21600) * hh;
359 ¢          trw1:=(((60-st1)/2) * (1-cos(h))) + st1;
360 ¢        end;
361 ¢   if ((hh>=21600) and (hh<57600))
362 ¢   then begin
363 ¢          h:=(pi/36000) * (hh-21600);
364 ¢          trw1:=(((60-st1)/2) * (1+cos(h))) + st1;
365 ¢        end;
366 ¢   if ((hh>=57600) and (hh<86400))
367 ¢   then trw1:=st1;
368 ¢
369   For i:=0 to m do t[i,0]:=trw1;
370   writeln (' Beginn der Iteration, Randtemperatur:',trw1:4:1);
371   itn:=0;
372   for p:=1 to l do
373     nc[p]:=2.567*(sqrt(sqrt(abs(trw3-t[p,w])))) * deltaX * 5;
374   for p:=1 to k do
375   begin
376     i:=v5[1,p];
377     j:=v5[2,p];
378     h:=(1-v6[p]) / (1+v6[p]);
379     v7[p]:=h*t[i,j] + v1[p]*t[i,j-1] + v4[p]*t[i,j+1]
380                     + v2[p]*t[i-1,j] + v3[p]*t[i+1,j];
381   end;
382
383   repeat
384     time (t3);
385     itn:=itn+1;
386     s:=0;
387     for p:=1 to k do
388     begin
389       i:=v5[1,p];
390       j:=v5[2,p];
391       h:=(1-omega)*t[i,j] + omega *
392          (v1[p]*t[i,j-1] + v2[p]*t[i-1,j] +
393           v3[p]*t[i+1,j] + v4[p]*t[i,j+1] +v7[p]);
394       s:=max(s,abs(h-t[i,j]));    (*182*)
395       t[i,j]:=h;
396     end;
397     (* Gleichung für den unteren Rand *)
398     for i:=1 to l do
```

```
399      begin
400        h:=(1-omega)*t[i,w] + (omega/(1+nc[i])) *
401             (t[i,w-1] + nc[i] * trw3);
402        s:=max(s,abs(h-t[i,w]));    (*182*)
403        t[i,w]:=h;
404      end;
405      time (t4);
406      zt:=(t4-t3) / 10000;
407      writeln ('  ',itn:2,'.Iteration  s:',s:5:3,'  Zeit:',zt:7:4,' sec
408    until (s<eps) or (itn>na1);
409
410
411    time (t2);
412    zt:=(t2-t1) / 10000;
413    writeln (' Gesamtzeit: ',zt:7:4,' sec');
414
415
416
417  (* Ausgabe der Temperatur *)
418
419     if lauf mod na3 =0 then
420     begin
421      h:=(lauf * deltaT) / 3600;
422      hh:=round(h);
423      writeln (' Ausgabe der Temperatur ');
424      writeln (Resultat,'A');
425      for i:=0 to 6 do writeln (Resultat,'  ');
426      writeln (Resultat,'   Temperatur am oberen Rand:  ',trw1:3:1,
427                        ' Grad          ');
428      writeln (Resultat,'   Temperatur am unteren Rand: ',t[8,w]:3:1,
429                        ' Grad          ');
430      write   (Resultat,'   Zeitpunkt: T=',hh:2,' h');
431      writeln (resultat,'   DELTAT: ',deltaT:4,' sec          ');
432      for i:=0 to 3 do writeln (Resultat,'  ');
433      writeln (Resultat,'  *');
434      for j:=0 to w do
435      begin
436        write (Resultat,'  ');
437        for i:=1 to l do
438           write (Resultat,Ausgabe(t i,j ));   (*159*)
439        writeln (Resultat);
440      end;
441      writeln (Resultat,'  *')
442    end;
443
444    until lauf=na2;
445
446    writeln (' Programmende ');
447  end.

END CRNVML    L= 10744
```

Literaturverzeichnis

[1] Robert D. Richtmyer; K. W. Morton

Difference Methods For Initial-Value Problems

[2] Alan J. Chapman

Heat Transfer

[3] A. Bjorck; G. Dahlquist

Numerische Methoden