รายการอ้างอิง

ภาษาไทย

ดำรงค์ ทิพย์โยธา, เพ็ญพรรณ ยังคง, พีชคณิตเชิงเส้น. พิมพ์ครั้งที่ 3, กรุงเทพมหานคร, โรงพิมพ์จุฬาลงกรณ์มหาวิทยาลัย, 2540.

มงคล สีห์โสภณ, พีชคณิตเชิงเส้น สาขาคณิตศาสตร์-สถิติ คณวิทยาศาสตร์ และเทคโนโลยี มหาวิทยาลัยธรรมศาสตร์, สำนักพิมพ์ประกายพรึก, 2534.

ภาษาอังกฤษ

Chen, H.T., Parametric Pumping of Separation Techniques for chemical engineers. McGraw-Hill, New York, 1979

Chen, H.T.,T.K.Hsich, H,C.Lee and F.B.Hill, Separation of Proteins Via semicontinuous pH-Parametric Pumping. AIChe J.,23, 695, 1977.

Chen, H.T.,T.Pamchareon, W.T.Yang, C.O.Kerobo and R.J.Parisi, An Equilibrium Theory of the pH-Parametric Pump. paper presented at AICHE Nationnal Meeting Boston, Mass. (Aug., 1979) also Separation Sci. & Tech. 15, 1377, 1980.

Rice, R.G., Progress inParametric Pumping. Sep. Pur. Methods, 5, NO.1, 139, 1976.

Sabadell, J.E., and N.H.Sweed, Parametric Pumping with pH. Separation Sci., 5, 171, 1970

Shaffer, A.G., and C.E.Hamrin, Enzyme Separation byParametric Pumping. AIChE J.,21, 782, 1975.

Sweed, N.H., Parametric Pumping. Progress in Separation and Purification, Vol.4, John Wiley, New York, 1971.

Wankat, P.C., Cyclic Separation Processes. Separation Sci.,9, 85, 1974.

ภาคผนวก ก

การทำนายผลการทดลองโดยใช้กราฟ

ก.1 วิธีการทำนายผลการทดลองโดยใช้กราฟ

ในการศึกษาของ Chen การแสดงผลการทดลองที่ได้ สามารถแสดงได้ในรูปของกราฟ ซึ่งวิธีการหาค่าความเข้มข้นของการแยกสารโปรตีนดังกล่าว สามารถหาได้จากกราฟที่เขียนขึ้นระหว่างค่าความเข้มข้นของโปรตีนในวัฏภาคของแข็ง และค่าความเข้มข้นของโปรตีนในวัฏภาคของเหลว ซึ่งในการทดลองของ Chen ได้กำหนดให้สมดุลระหว่างวัฏภาคเป็นฟังก์ชั่นเชิงเส้น ในรูป

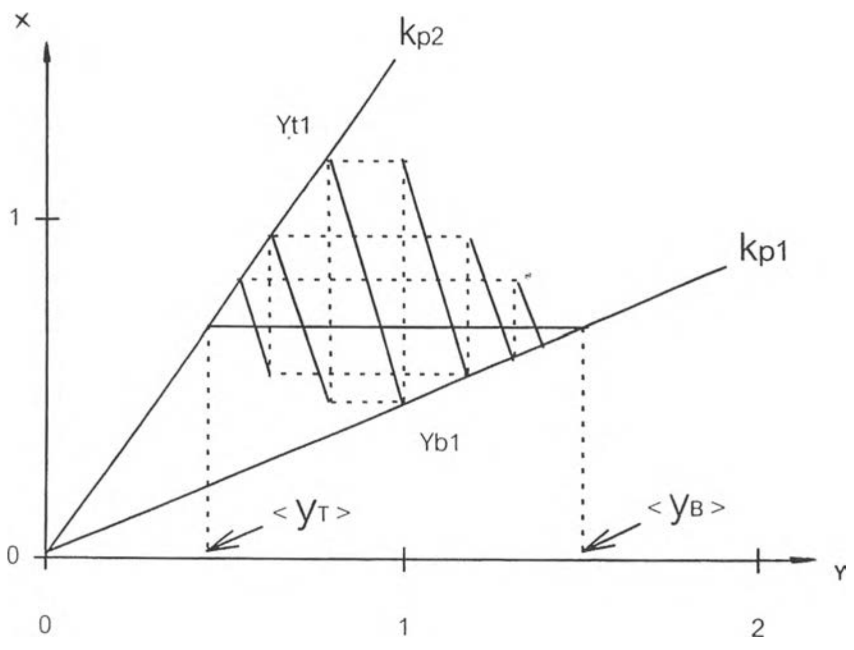
$$x = f(y) = ky \qquad \text{..................3.3}$$

กำหนดให้ x เป็นค่าความเข้มข้นของโปรตีนในวัฏภาคของแข็ง

y เป็นค่าความเข้มข้นของโปรตีนในวัฏภาคของเหลว

k เป็นค่าคงที่ของสมดุล

จากสมการที่ 3.3 สามารถนำไปเขียนเส้นสมดุลได้ 2 เส้น สำหรับในระบบ 1 คอลัมน์ระหว่างค่าพีเอชทั้ง 2 ค่า ที่ใช้ในการทดลอง คือ pH_1 และ pH_2 บนระนาบแกน X-Y แสดงได้ตามรูปที่ ก.1



รูปที่ ค.1 รูปแสดงวิธีการทำนายผลการทดลองโดยใช้กราฟ ซึ่งสมดุลระหว่างวัฏภาคเป็นฟังก์ชั่นเชิงเส้น

ขั้นตอนของการทำนายผลโดยใช้กราฟ สามารถอธิบายได้ดังนี้

1. เมื่อสารละลายในถังพักด้านบน (y_T) ถูกส่งผ่านเข้าไปในคอลัมน์ และสารละลายในคอลัมน์ถูกส่งผ่านไปยังถังพักด้านล่าง (y_B) จะทำให้ค่าความเข้มข้นของถังพักด้านล่างจะมีค่าเท่ากับ y_0
2. เป็นขั้นตอนการปรับสภาพค่าพีเอช ในคอลัมน์ ให้คงที่ที่ pH_2 และเมื่อระบบเข้าสู่สภาวะสมดุลองค์ประกอบใหม่ในคอลัมน์ได้เป็น (y_{T1} , x_{T1}) ซึ่งสามารถคำนวณได้โดยการใช้สมการสมดุลที่ 3.1 และสมการที่ 3.3 ถ้าเชื่อมจุด y_{B1} กับ y_{T1} จะได้เส้นตรงที่เรียกว่า เส้นดำเนินการทดลอง (operating line)
3. สารละลายในถังพักด้านล่าง (y_B) ถูกส่งผ่านเข้าไปในคอลัมน์ และสารละลายในคอลัมน์ถูกส่งผ่านไปยังถังพักด้านบน (y_T) เมื่อสิ้นสุดขั้นตอนนี้พบว่าองค์ประกอบในคอลัมน์แสดงได้เป็น

4. เป็นขั้นตอนการปรับสภาพค่าพีเอช ในคอลัมน์ ให้คงที่ที่ pH_1 และเมื่อระบบเข้าสู่สภาวะสมดุลองค์ประกอบใหม่ในคอลัมน์ได้เป็น (y_{B1} , x_{B1}) ซึ่งสามารถคำนวณได้โดยการใช้สมการสมดุลที่ 3.1 และสมการที่ 3.3 จะสามารถสร้างเส้นดำเนินการทดลองได้อีกเส้นหนึ่ง ซึ่งพบว่าเส้นดำเนินการเส้นใหม่ขนานกับเส้นดำเนินการเดิม

การทำนายผลการทดลองในลำดับขั้นต่อไป จะเป็นการทำซ้ำในขั้นตอนที่ 1 ถึงขั้นตอนที่ 4 ข้างต้น โดยที่ค่าความเข้มข้นในถังพักการทดลองด้านล่าง จะเปลี่ยนจาก y_{B1} เป็น y_{B2} และเมื่อได้ทำการทำนายผลการทดลองจนกระทั่งจำนวนรอบของการทำนายมีค่ามากพอ ค่าความเข้มข้นของสารละลายในถังพักด้านบน (y_T) และถังพักทางด้านล่าง (y_B) ที่ต้องการจะมีค่าสู่สภาวะคงตัว (เส้นทึบ) ดังที่แสดงไว้ในรูปที่ ค.1

ในลักษณะเดียวกัน การศึกษาของผู้วิจัยในครั้งนี้ ทางผู้วิจัยได้ตั้งสมมติฐาน ให้สมดุลระหว่างวัฏภาคไม่เป็นฟังก์ชั่นเชิงเส้น ซึ่งมีความสัมพันธ์ตามสมการที่ 3.5 และสมการที่ 3.6 คือ ในคอลัมน์ที่บรรจุตัวแลกเปลี่ยนประจุบวก มีความสัมพันธ์ดังนี้

$$x = f(y) = Ay^3 + By^2 + Cy + D \qquad \text{......................3.5}$$

ในคอลัมน์ที่บรรจุตัวแลกเปลี่ยนประจุลบ มีความสัมพันธ์ดังนี้

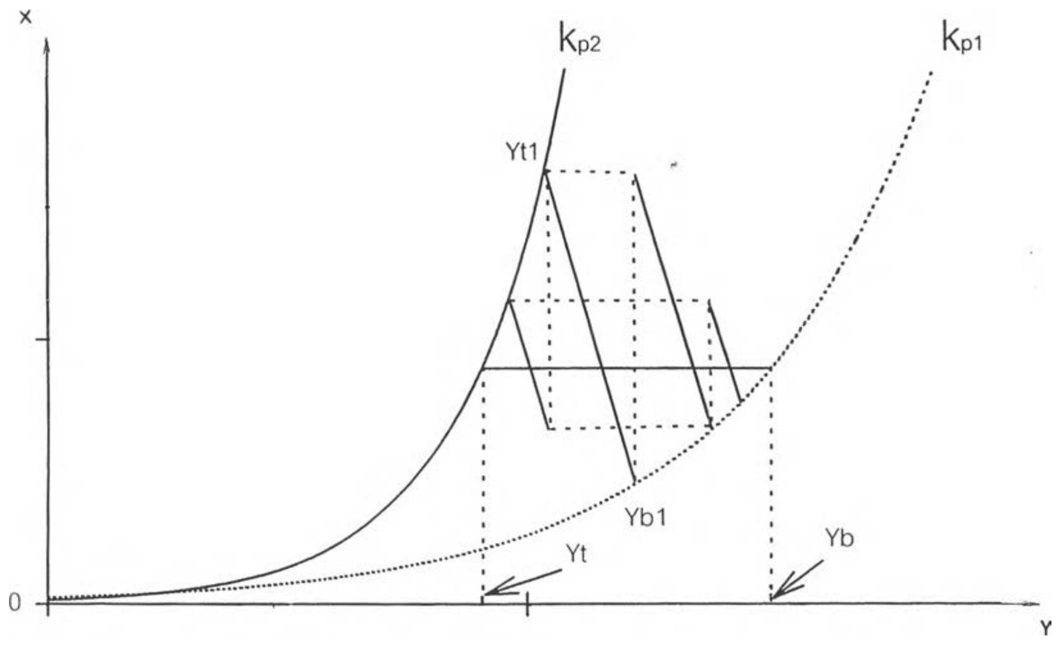
$$x = f(y) = Ay^2 + By + C + D/y \qquad \text{......................3.6}$$

กำหนดให้ x เป็นค่าความเข้มข้นของโปรตีนในวัฏภาคของแข็ง

y เป็นค่าความเข้มข้นของโปรตีนในวัฏภาคของเหลว

A,B,C,D เป็นค่าคงที่

การทำนายผลการทดลองตามสมมติฐานข้างต้น สามารถใช้วิธีการทำนายแบบเดียวกันกับวิธีการที่กำหนดให้สมดุลระหว่างวัฏภาคเป็นฟังก์ชั่นเชิงเส้น จากสมการที่ 3.5 และสมการที่ 3.6 สามารถเขียนเส้นสมดุลได้ 2 เส้น สำหรับระบบ 1 คอลัมน์ บนระนาบ X-Y ขั้นตอนการทำนายผลให้ดำเนินการตามขั้นตอนข้างต้น เหมือนกับการที่กำหนดให้ สมดุลระหว่างวัฏภาค เป็นฟังก์ชั่นเชิงเส้นซึ่งแสดงได้ตามรูปที่ ค.2



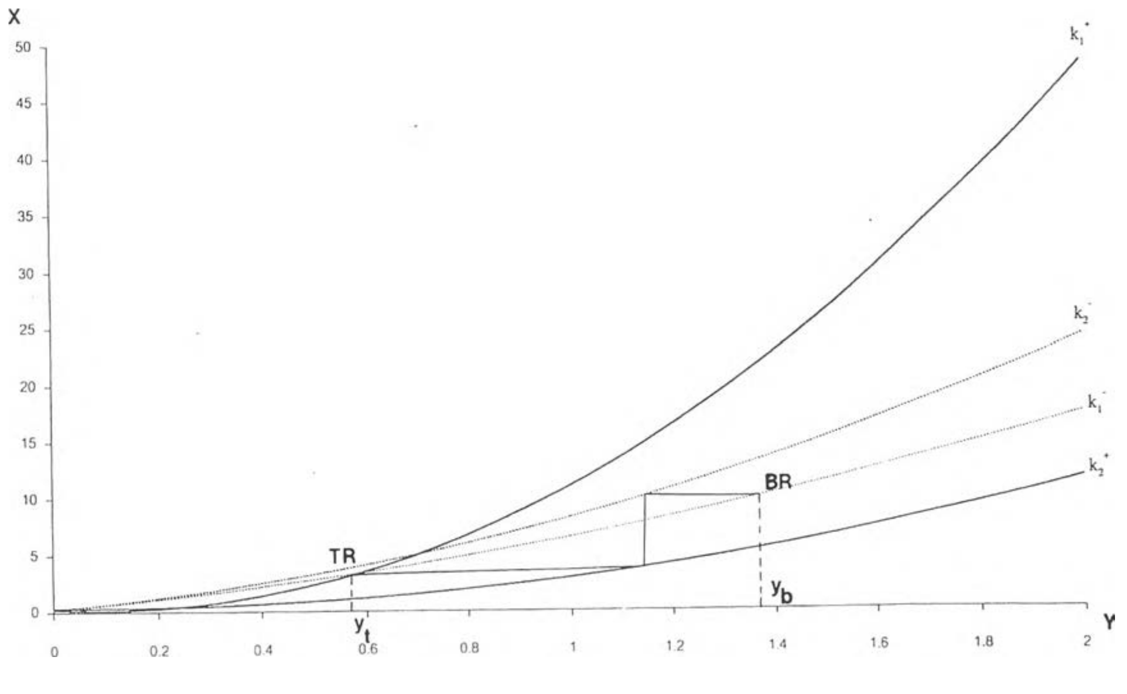
รูปที่ ก.2 รูปแสดงวิธีการทำนายผลการทดลองโดยใช้กราฟ ซึ่งสมดุลระหว่างวัฏภาคไม่เป็นฟังก์ชั่นเชิงเส้น

ค.2 การแสดงผลการทดลองโดยวิธีการใช้กราฟ

การทำนายผลการทดลองโดยวิธีการใช้กราฟ ในระบบที่ประกอบด้วยคอลัมน์ 2 คอลัมน์ สามารถใช้วิธีการทำนายแบบเดียวกันกับวิธีการที่กำหนดให้ สมดุลระหว่างวัฏภาคเป็นฟังก์ชั่นเชิงเส้น จากสมการที่ 3.5 และสมการที่ 3.6 สามารถเขียนเส้นสมดุลได้ 4 เส้น สำหรับระบบ 2 คอลัมน์ บนระนาบ X-Y ขั้นตอนการทำนายผลให้ดำเนินการตามขั้นตอนข้างต้น เหมือนกับการที่กำหนดให้ สมดุลระหว่างวัฏภาค เป็นฟังก์ชั่นเชิงเส้น ซึ่งแสดงได้ตามรูปที่ค.2

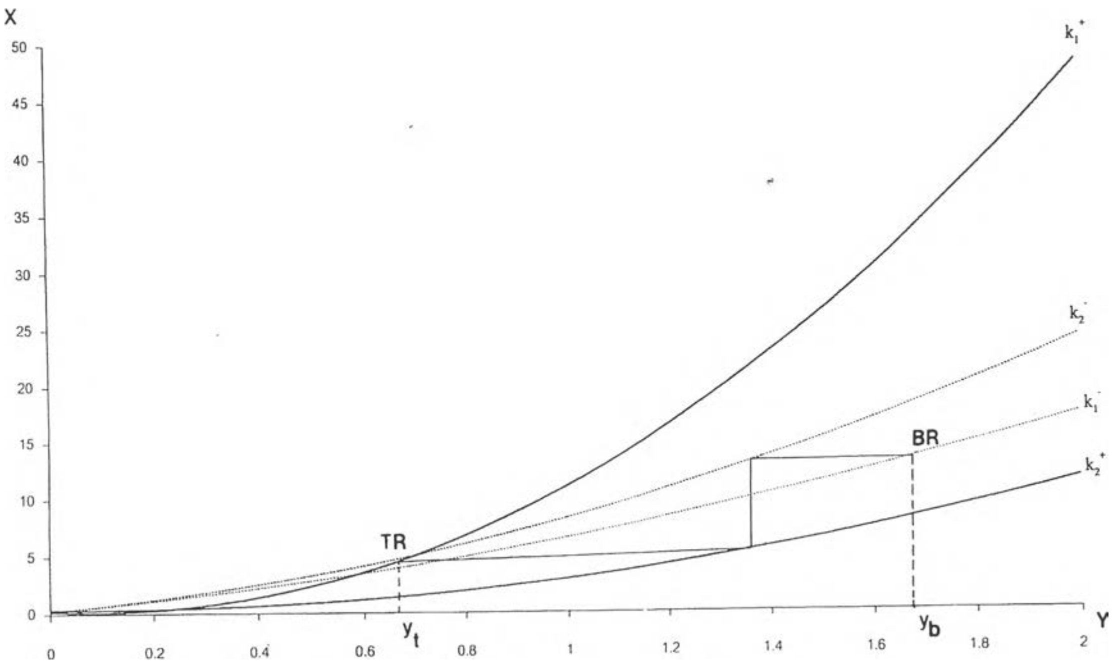
การแสดงผลการทำนายผลการทดลอง สามารถแสดงได้ในรูปของกราฟเช่นเดียวกัน ซึ่งผลการทดลองในแต่ละรูปแบบของการทดลอง สามารถแสดงได้ดังนี้

รูปแบบการทดลองที่ 1

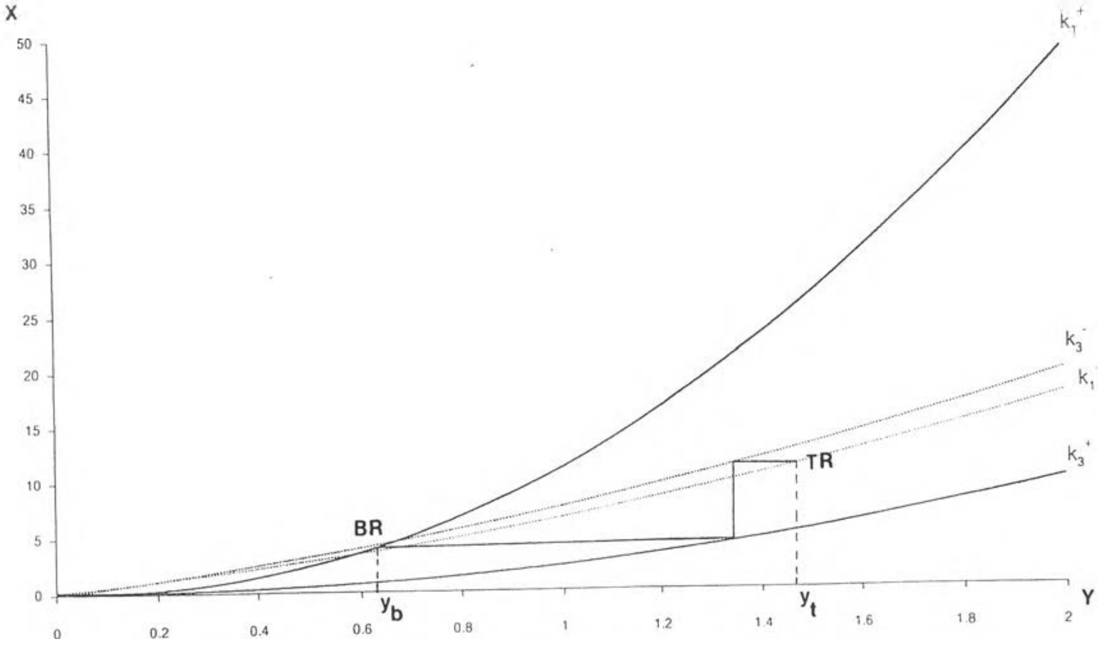


รูปที่ ค.3 กราฟแสดงผลการทดลองของรูปแบบการทดลองที่ 1 (ฮีโมโกลบิน)

<u>รูปแบบการทดลองที่ 2</u>

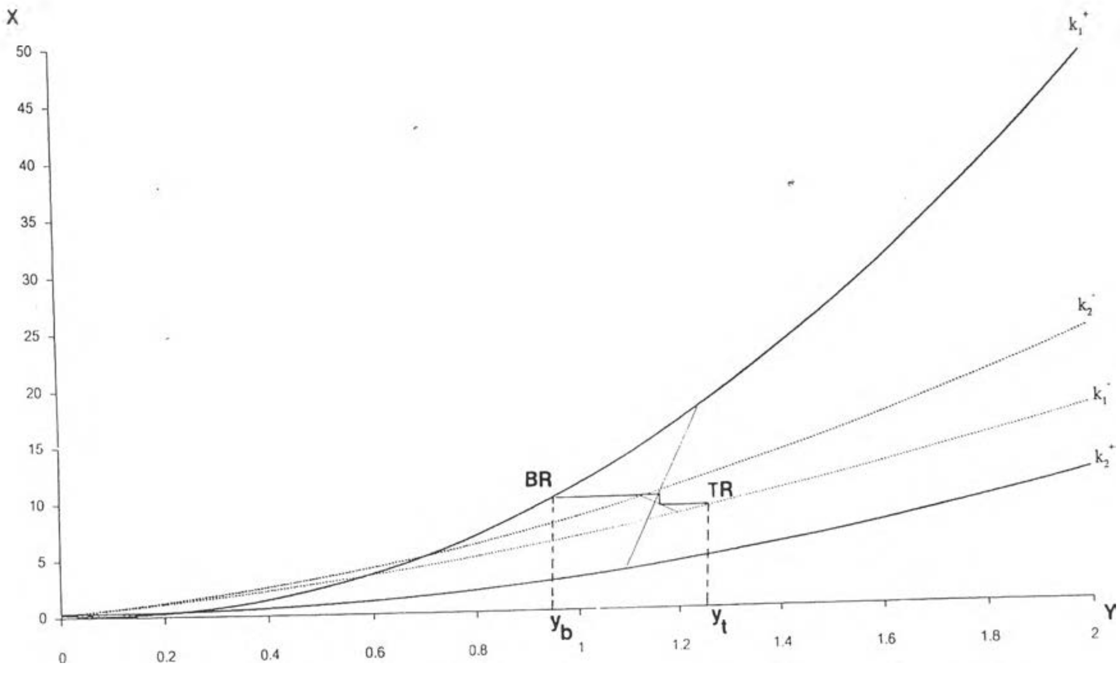


รูปที่ ก.4 กราฟแสดงผลการทดลองของรูปแบบการทดลองที่ 2 (ฮีโมโกลบิน)

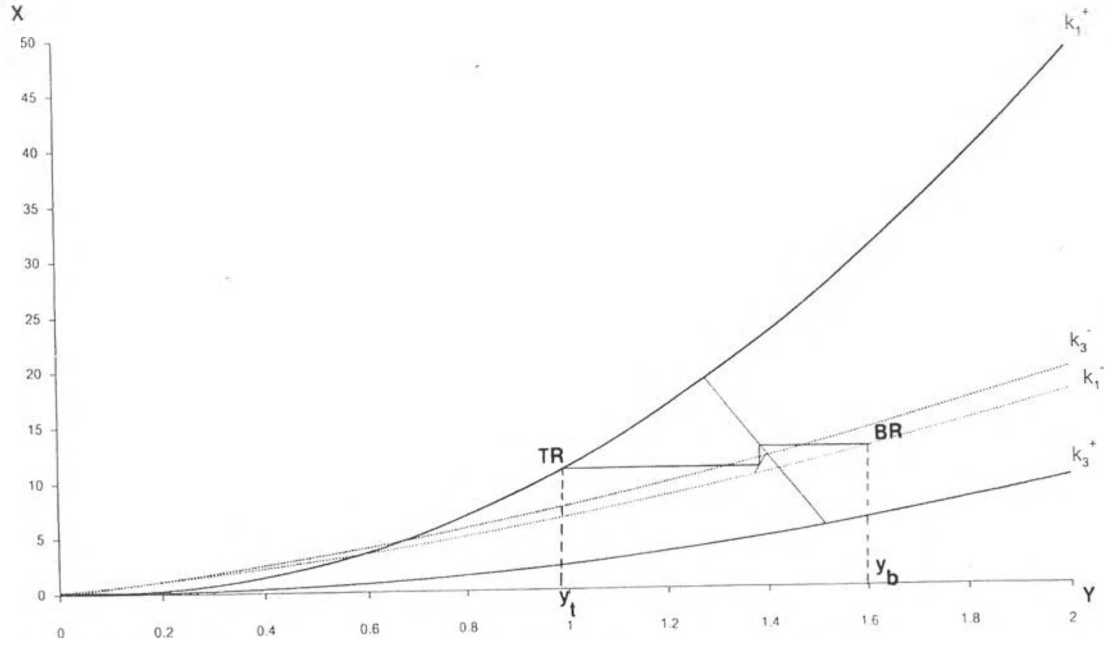


รูปที่ ก.5 กราฟแสดงผลการทดลองของรูปแบบการทดลองที่ 2 (อัลบูบิน)

<u>รูปแบบการทดลองที่ 3</u>

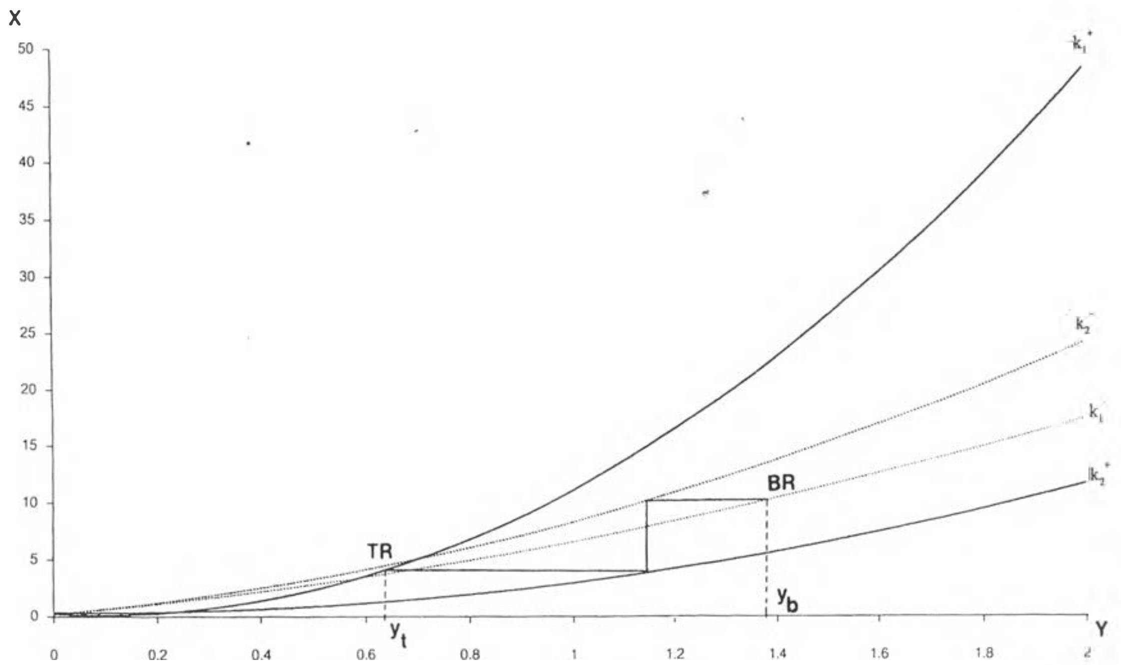


รูปที่ค.6 กราฟแสดงผลการทดลองของรูปแบบการทดลองที่ 3 (ฮีโมโกลบิน)

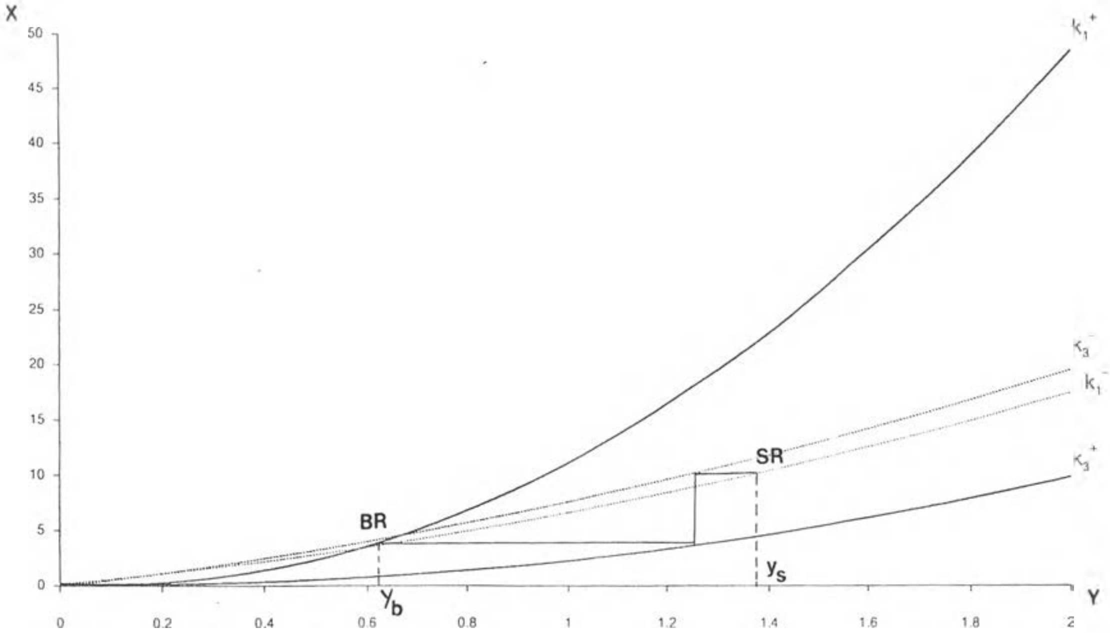


รูปที่ค.7 กราฟแสดงผลการทดลองของรูปแบบการทดลองที่ 3 (อัลบูบิน)

<u>รูปแบบการทดลองที่ 4</u>

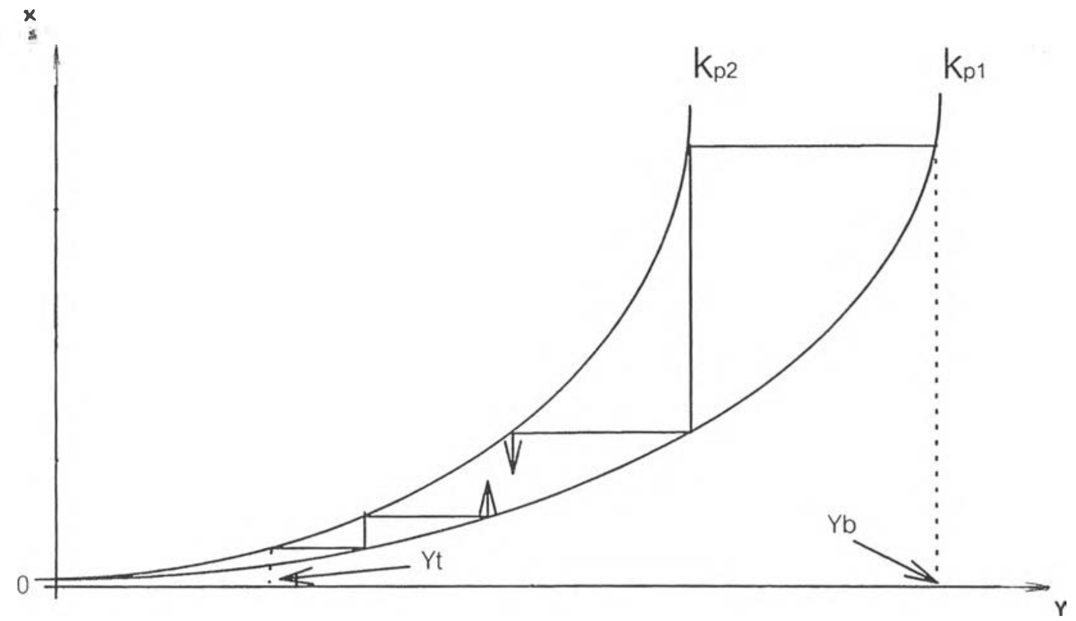


รูปที่ค.8 กราฟแสดงผลการทดลองของรูปแบบการทดลองที่ 4 (ฮีโมโกลบิน)



รูปที่ค.9 กราฟแสดงผลการทดลองของรูปแบบการทดลองที่ 4 (อัลบูบิน)

จากวิธีการทำนายผลการทดลองโดยวิธีการใช้กราฟนี้ สามารถนำไปประยุกต์ใช้ทำนายผลการทดลอง สำหรับการทดลองในรูปแบบที่มีจำนวนของคอลัมน์มากกว่า 2 คอลัมน์ (n ชุด) ได้อีกด้วย ซึ่งในการแสดงวิธีการทำนายผลการทดลองในระบบที่ประกอบไปด้วยคอลัมน์ n ชุด สามารถแสดงได้ในรูปที่ค.10



รูปที่ค.10 รูปแสดงวิธีการทำนายผลการแยกโปรตีนในระบบที่ประกอบด้วยคอลัมน์ n ชุด

การทำนายผลการทดลองในระบบดังกล่าว สามารถใช้วิธีการทำนายแบบเดียวกันกับวิธีการที่กำหนดให้สมดุลระหว่างวัฏภาคไม่เป็นฟังก์ชั่นเชิงเส้น จากสมการที่ 3.5 และสมการที่ 3.6 สามารถเขียนเส้นสมดุลได้ สำหรับระบบหลายคอลัมน์ ขั้นตอนการทำนายผลให้ดำเนินการตามขั้นตอนข้างต้น

ภาคผนวก ข

โปรแกรมคอมพิวเตอร์

```
//  Project File             ( PMP.DPR )
program PMP;
uses
  Forms,
  Unit1 in 'Unit1.pas' {W_Main},
  TypeI in 'TypeI.pas' {W_Mode1},
  Graph in 'Graph.pas' {W_Graph},
  TypeII in 'TypeII.pas' {W_Mode2},
  Monitor in 'Monitor.pas' {W_Monitor},
  TypeIII in 'TypeIII.pas' {W_Mode3},
  TypeIV in 'TypeIV.pas' {W_Mode4},
  Printing in 'Printing.pas' {W_Report};
{$R *.RES}
begin
  Application.Initialize;
  Application.CreateForm(TW_Main, W_Main);
  Application.CreateForm(TW_Mode1, W_Mode1);
  Application.CreateForm(TW_Graph, W_Graph);
  Application.CreateForm(TW_Mode2, W_Mode2);
  Application.CreateForm(TW_Monitor, W_Monitor);
  Application.CreateForm(TW_Mode3, W_Mode3);
  Application.CreateForm(TW_Mode4, W_Mode4);
  Application.CreateForm(TW_Report, W_Report);
  Application.Run;
end.
```

```
// Program for Main form    ( Main.Pas )
unit Main;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons, ExtCtrls;

type
  TW_Main = class(TForm)
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    BitBtn5: TBitBtn;
    Bevel1: TBevel;
    procedure BitBtn5Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  W_Main: TW_Main;
implementation
```

```
uses TypeI, TypeII, TypeIII, TypeIV;
{$R *.DFM}
procedure TW_Main.BitBtn5Click(Sender: TObject);
begin
 Close;
end;

procedure TW_Main.BitBtn1Click(Sender: TObject);
begin
  W_Mode1.Showmodal;
end;

procedure TW_Main.BitBtn2Click(Sender: TObject);
begin
  W_Mode2.Showmodal;
end;

procedure TW_Main.BitBtn3Click(Sender: TObject);
begin
  W_Mode3.ShowModal;
end;

procedure TW_Main.BitBtn4Click(Sender: TObject);
begin
  W_Mode4.Showmodal;
end;
end.
```

// Program for TypeI Form (TypeI.Pas)

```
unit TypeI;interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls,
ExtCtrls, Buttons, ComCtrls,Math;
type TW_Mode1 = class(TForm)   Button1: TButton;   BitBtn3: TBitBtn;   BitBtn2:
TBitBtn;
    BitBtn1: TBitBtn;
    Bevel1: TBevel;
    Bevel2: TBevel;
    Label4: TLabel;
    Label7: TLabel;
    Label1: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label2: TLabel;
    Label11: TLabel;
    Edit4: TEdit;
    Edit5: TEdit;
    ListBox4: TListBox;
    ListBox5: TListBox;
    Edit1: TEdit;
    ListBox1: TListBox;
    Edit6: TEdit;
    Edit7: TEdit;
    Edit8: TEdit;
    ListBox2: TListBox;
    Edit11: TEdit;
```

```
Label12: TLabel;
Edit12: TEdit;
ComboBox1: TComboBox;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label14: TLabel;
Label15: TLabel;
Label16: TLabel;
Label17: TLabel;
Label18: TLabel;
Label19: TLabel;
Label3: TLabel;
Label5: TLabel;
EAp1: TEdit;
EBp1: TEdit;
ECp1: TEdit;
EAp2: TEdit;
EBp2: TEdit;
ECp2: TEdit;
EDp1: TEdit;
EDp2: TEdit;
TabSheet2: TTabSheet;
Label6: TLabel;
Label10: TLabel;
Label20: TLabel;
Label21: TLabel;
Label22: TLabel;
Label23: TLabel;
Label24: TLabel;
```

```
Label25: TLabel;
EAn1: TEdit;
EBn1: TEdit;
ECn1: TEdit;
EAn2: TEdit;
EBn2: TEdit;
ECn2: TEdit;
EDn1: TEdit;
EDn2: TEdit;
KCheckBox: TCheckBox;

procedure SetMatrixDown;
procedure SetMatrixUp;
procedure CalMatrix;
procedure ShiftUp;
procedure ClearMatrix;
procedure ShiftDown;
procedure DownFlow;
procedure InitAllvalue;
procedure UpFlow;
procedure Calculate;
procedure BitBtn2Click(Sender: TObject);
procedure SetRightStep;
procedure ListBox4Click(Sender: TObject);
procedure ListBox5Click(Sender: TObject);
procedure ListBox1Click(Sender: TObject);
procedure Edit6Change(Sender: TObject);
procedure BitBtn3Click(Sender: TObject);
procedure BitBtn1Click(Sender: TObject);
```

```
  procedure FormShow(Sender: TObject);
  procedure ListBox2Click(Sender: TObject);
  procedure Button2Click(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  function  AddText(str : string;length : integer) : string;
  function  strleng(str : string) : integer;
  function  Kfunction(y : double):double;

 private
   { Private declarations }
 public
  { Public declarations }
 end;

var
 W_Mode1: TW_Mode1;
 MatrixV : Array [0..1,0..100,0..100] of double;
 Ap,Bp,Cp,Dp,An,Bn,Cn,Dn : Array [1..2] of double;
 MatrixY,MatrixSum,MatrixDev : Array [0..11,0..100] of double;
 YmAvg,OldYmAvg : Array [0..11] of double;
 V,VB,Y0,YtAvg,YbAvg : double;
 Step,Round,ColumnIndex,ColumnCount,Kindex,StepCount:integer;

implementation

uses Graph, Monitor, Printing;

{$R *.DFM}
```

```
procedure TW_Mode1.CalMatrix;
var i,j: integer;



begin
 for j:=0 to Step do
 begin
   MatrixSum[ColumnIndex,j]:=0;
   for i:=0 to Step do
        MatrixSum[ColumnIndex,j]:=MatrixSum[ColumnIndex,j]+MatrixY[ColumnIndex,i]
               *MatrixV[ColumnIndex mod 2,i,j];
   MatrixSum[ColumnIndex,j]:=MatrixSum[ColumnIndex,j]/MatrixDev[ColumnIndex mod
2,j];
  end;
end;

procedure TW_Mode1.ClearMatrix;
var i,j :integer;
begin

 if ColumnIndex<2 then
 begin
  for i:=0 to Step do
  for j:=0 to Step do MatrixV[ColumnIndex,i,j]:=0;
 end;
end;

function TW_Mode1.Kfunction(y : double):double;
 var i : integer;
```

```
    YY : double;

Begin
    i:=Kindex;
    if KCheckBox.Checked then YY:=1 else YY:=y;
    if ColumnIndex mod 2 = 0 then Kfunction := Ap[i]*Power(y,-2)+Bp[i]*Power(y,-1)
       +Cp[i]+Dp[i]/Power(YY,-1)
        else    Kfunction := An[i]*Power(y,2)+Bn[i]*Power(y,1)+Cn[i]+Dn[i]/Power(YY,1);
end;

procedure TW_Mode1.SetMatrixDown;
 var i,j: integer;
 begin
   ClearMatrix;
   for i:=0 to Step-1 do
    begin
     MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i];
    end;
   if ColumnIndex=0 then  MatrixY[ColumnIndex,Step]:=YtAvg
    else MatrixY[ColumnIndex,Step]:=YmAvg[ColumnIndex-1];
   i:=0;
   if ColumnIndex mod 2=0 then Kindex:=2 else Kindex:=1;
   if ColumnIndex<2 then
   begin
     for j:= 1 to Step do
      begin
       MatrixV[ColumnIndex,i,j]:= VB*Kfunction(MatrixY[ColumnIndex,j-1]);
       MatrixV[ColumnIndex,i+1,j]:= V;
       i:=i+1;
      end;
     MatrixV[ColumnIndex,0,0]:=VB*Kfunction(MatrixY[ColumnIndex,0])+V;
   end;

   for i:=1 to Step-1 do
     if ColumnIndex<2 then MatrixDev[ColumnIndex,i]:=
V+VB*Kfunction(MatrixY[ColumnIndex,i-1]);

   if ColumnIndex mod 2=0 then Kindex:=1 else Kindex:=2;
     if ColumnIndex<2 then
       Begin
         MatrixDev[ColumnIndex,0]:=MatrixV[ColumnIndex,0,0];
         MatrixDev[ColumnIndex,Step]:=V+VB*Kfunction(MatrixY[ColumnIndex,Step-1]);
       End;
end;

procedure TW_Mode1.SetMatrixUp;
 var i,j: integer;
 begin
   ClearMatrix;
   for i:=1 to Step do
    begin
     MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i];
    end;
    if ColumnIndex=ColumnCount-1 then   MatrixY[ColumnIndex,0]:=YbAvg
      else MatrixY[ColumnIndex,0]:=YmAvg[ColumnIndex];
    if ColumnIndex mod 2=0 then Kindex:=1 else Kindex:=2;
    if ColumnIndex<2 then
   begin
```

```
    i:=0;
   for j:= 0 to Step-1 do
    begin
     MatrixV[ColumnIndex,i+1,j]:= VB*Kfunction(MatrixY[ColumnIndex,j+1]);
     MatrixV[ColumnIndex,i,j]:= V;
     i:=i+1;
    end;
    MatrixV[ColumnIndex,Step,Step]:=VB*Kfunction(MatrixY[ColumnIndex,Step])+V;
  end;
  for i:=1 to Step-1 do
   begin
     if ColumnIndex<2 then MatrixDev[ColumnIndex,i]:=
            VB*Kfunction(MatrixY[ColumnIndex,i+1])+V;
   end;

    if ColumnIndex mod 2=0 then Kindex:=2 else Kindex:=1;
  if ColumnIndex<2 then
    begin
     MatrixDev[ColumnIndex,Step]:=MatrixV[ColumnIndex,Step,Step];
     MatrixDev[ColumnIndex,0]:=VB*Kfunction(MatrixY[ColumnIndex,1])+V;
    end;
 end;

procedure TW_Mode1.ShiftDown;
var i,j : integer;
begin
  for i:=Step downto 0 do
  begin
    MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i-1];
  end;

  if ColumnIndex=ColumnCount-1 then MatrixY[ColumnIndex,0]:=YbAvg else
    MatrixY[ColumnIndex,0]:=YmAvg[ColumnIndex];
   if ColumnIndex<2 then
  begin
     i:=0;
    for j:= 0 to Step-1 do
     begin
      if ColumnIndex mod 2=0 then
         begin
           if j<StepCount-1 then  Kindex:=2 else Kindex:=1;
         end else
         begin
           if j<StepCount-1 then  Kindex:=1 else Kindex:=2;
         end;

      MatrixV[ColumnIndex,i+1,j]:= VB*Kfunction(MatrixY[ColumnIndex,j+1]);
      MatrixV[ColumnIndex,i,j]:= V;
      i:=i+1;
     end;
     if ColumnIndex mod 2=0 then Kindex:=1 else Kindex:=2;
MatrixV[ColumnIndex,Step,Step]:=VB*Kfunction(MatrixY[ColumnIndex,Step])+V;
  end;

  for i:=0 to Step-1 do
   begin
     if ColumnIndex mod 2=0 then
         begin
```

```
          if i<StepCount then  Kindex:=2 else Kindex:=1;
        end else
        begin
          if i<StepCount then  Kindex:=1 else Kindex:=2;
        end;
      if ColumnIndex<2 then
        MatrixDev[ColumnIndex,i]:= VB*Kfunction(MatrixY[ColumnIndex,i+1])+V;

   end;

   if ColumnIndex<2 then
    begin
     MatrixDev[ColumnIndex,Step]:=MatrixV[ColumnIndex,Step,Step];
    end;

end;

procedure TW_Mode1.ShiftUp;
var i,j : integer;
begin
  for i:=0 to Step-1 do
   begin
    MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i+1];
   end;

  if ColumnIndex=0 then MatrixY[ColumnIndex,Step]:=YtAvg else
    MatrixY[ColumnIndex,Step]:=YmAvg[ColumnIndex-1];
    i:=0;
   if ColumnIndex mod 2=0 then Kindex:=2 else Kindex:=1;
if ColumnIndex<2 then
begin
    for j:= 1 to Step do
  begin
     if ColumnIndex mod 2=0 then
      begin
        if j>Step-StepCount+1 then  Kindex:=1 else Kindex:=2;
      end else
      begin
        if j>Step-StepCount+1 then  Kindex:=2 else Kindex:=1;
      end;
    MatrixV[ColumnIndex,i,j]:= VB*Kfunction(MatrixY[ColumnIndex,j-1]);
    MatrixV[ColumnIndex,i+1,j]:= V;
    i:=i+1;
  end;

   if ColumnIndex mod 2=0 then Kindex:=2 else Kindex:=1;
   MatrixV[ColumnIndex,0,0]:=VB*Kfunction(MatrixY[ColumnIndex,0])+V;
end;

for i:=1 to Step do
 begin
  if ColumnIndex mod 2=0 then
      begin
        if i>Step-StepCount then  Kindex:=1 else Kindex:=2;
      end else
      begin
        if i>Step-StepCount then  Kindex:=2 else Kindex:=1;
      end;
    if ColumnIndex<2 then MatrixDev[ColumnIndex,i]:=
         V+VB*Kfunction(MatrixY[ColumnIndex,i-1]);
   end;

     if ColumnIndex<2 then
      Begin
        MatrixDev[ColumnIndex,0]:=MatrixV[ColumnIndex,0,0];
      End;

end;

procedure TW_Mode1.DownFlow;
var i,j: integer;
   averg1: double;
begin
 StepCount:=1;
  for i:=0 to ColumnCount-1 do
   begin
     ColumnIndex:=i ;
     SetMatrixDown;
   end;
  averg1:=0;
  for i:=0 to Step-1 do
   begin
    Inc(StepCount);
   for j:=0 to ColumnCount-2 do
    begin
      ColumnIndex:= j;
      if ColumnIndex>0 then MatrixY[ColumnIndex,Step]:=YmAvg[ColumnIndex-1];
     CalMatrix;
     YmAvg[ColumnIndex]:=OldYmAvg[ColumnIndex];
     OldYmAvg[ColumnIndex]:=(OldYmAvg[ColumnIndex]*(Step*2-1)
            +MatrixSum[ColumnIndex,0])/(2*Step);

    end;
   ColumnIndex:=ColumnCount-1;
   MatrixY[ColumnIndex,Step]:=YmAvg[ColumnIndex-1];

   CalMatrix;
   averg1:=averg1+MatrixSum[ColumnIndex,0];
   for i:=0 to ColumnCount-1 do
    begin
     ColumnIndex:=i;
     ShiftUp;
    end;
   end;
   YbAvg:=((YbAvg*Step)+averg1)/(2*Step);

end;

procedure TW_Mode1.UpFlow;
var i,j: integer;
   averg0: double;
begin
 StepCount:=1;
 for i:=0 to ColumnCount-1 do
 begin
   ColumnIndex:=i;
```

```
  SetMatrixUp;
 end;

 averg0:=0;
 for i:=0 to Step-1 do
  begin
  Inc(StepCount);
  for j:=ColumnCount-1 downto 1 do
  begin
  ColumnIndex:=j;
  if ColumnIndex < ColumnCount-1 then
MatrixY[ColumnIndex,0]:=YmAvg[ColumnIndex];

  CalMatrix;
  YmAvg[ColumnIndex-1]:=OldYmAvg[ColumnIndex-1];
  OldYmAvg[ColumnIndex-1]:=(OldYmAvg[ColumnIndex-1]*(Step*2-
1)+MatrixSum[ColumnIndex,Step])/(2*Step);
  end;

  ColumnIndex:=0;
  MatrixY[ColumnIndex,0]:=YmAvg[ColumnIndex];

  CalMatrix;
  averg0:=averg0+MatrixSum[ColumnIndex,Step];
   for i:=0 to ColumnCount-1 do
     begin
      ColumnIndex:=i;
      ShiftDown;
     end;
  end;
  YtAvg:=((YtAvg*Step)+averg0)/(2*Step);
end;

procedure TW_Mode1.InitAllValue;
var i,j: integer;
begin
  ColumnCount:=Strtoint(Combobox1.text);
  BitBtn2.Enabled:=True;

  V:=strtofloat(Edit4.Text);
  Y0:=Strtofloat(Edit12.Text);

  Ap[1]:=strtofloat(EAp1.Text);
  Bp[1]:=strtofloat(EBp1.Text);
  Cp[1]:=strtofloat(ECp1.Text);
  Dp[1]:=strtofloat(EDp1.Text);
  Ap[2]:=strtofloat(EAp2.Text);
  Bp[2]:=strtofloat(EBp2.Text);
  Cp[2]:=Strtofloat(ECp2.Text);
  Dp[2]:=strtofloat(EDp2.Text);
  An[1]:=strtofloat(EAn1.Text);
  Bn[1]:=strtofloat(EBn1.Text);
  Cn[1]:=strtofloat(ECn1.Text);
  Dn[1]:=strtofloat(EDn1.Text);
  An[2]:=strtofloat(EAn2.Text);
  Bn[2]:=strtofloat(EBn2.Text);
  Cn[2]:=Strtofloat(ECn2.Text);
  Dn[2]:=strtofloat(EDn2.Text);
```

```
for i:=0 to ColumnCount-2 do
 begin
   YmAvg[i]:=Y0;
   OldYmAvg[i]:=Y0;
 end;
YtAvg:=Y0;
YbAvg:=Y0;
if strlen(Pchar(Edit5.text))<1 then SetRightStep else
 begin
   if strtoint(Edit5.Text)<2 then Edit5.Text:='2';
   Step:=Strtoint(Edit5.Text);
 end;

for i:=0 to ColumnCount-2 do
 begin
   YmAvg[i]:=Y0;
   OldYmAvg[i]:=Y0;
 end;
YtAvg:=Y0;
YbAvg:=Y0;

V:=strtofloat(Edit4.text)/(2*Step);
VB:=V/3;

for j:=0 to ColumnCount-1 do
begin
  for i:=0 to Step do
  begin
    MatrixY[j,i]:=Y0;
```

```
      MatrixSum[j,i]:=Y0;
    end;
  end;

end;

procedure TW_Mode1.SetRightStep;
var i,j: integer;
begin

  Step:=4;
  repeat
    Step:=Step+1;
    V:=strtofloat(Edit4.text)/(2*Step);
    VB:=V/3;
      for j:=0 to ColumnCount-1 do
      begin
          for i:=0 to Step do
              begin
                MatrixY[j,i]:=Y0;
                MatrixSum[j,i]:=Y0;
              end;
      end;
    DownFlow;
  until MatrixSum[0,0]>0.94;

  Edit5.Text:=inttostr(Step);
end;
```

```
procedure TW_Mode1.Calculate;
var i :integer;
  last0, last1 : double;
  stop:boolean;
begin

  ListBox4.Clear; ListBox5.clear; ListBox1.Clear;
  ListBox2.Clear;
  stop:=false; last0:=0; last1:=0;
  i:=0;
    ListBox1.Items.Add(inttostr(i));
    ListBox4.Items.Add(floattostrf(YtAvg,ffGeneral,6,6));

  repeat

    i:=i+1;
    if ColumnCount=2 then ListBox2.Items.Add(floattostrf(YmAvg[0],ffGeneral,6,6));
    DownFlow;

    ListBox5.Items.Add(floattostrf(YbAvg,ffGeneral,6,6));

    UpFlow;
    ListBox4.Items.Add(floattostrf(YtAvg,ffGeneral,6,6));

    if Abs(YbAvg-Last0)<0.00001 then Stop:=True;
    last0:=YbAvg;
    if Stop and (Abs(YtAvg-Last1)>0.00001) then Stop:=False;
    last1:=YtAvg;
    ListBox1.Items.Add(inttostr(i));
  until Stop or (i>100);
      DownFlow;
      ListBox5.Items.Add(floattostrf(YbAvg,ffGeneral,6,6));
  Edit1.text:=inttostr(i);
  Round:=i;
  for i:= 0 to ColumnCount-2 do
    ListBox2.Items.Add(floattostrf(YmAvg[i],ffGeneral,6,6));

end;

procedure TW_Mode1.BitBtn2Click(Sender: TObject);
begin
   Graph.PlotType:=1;
   W_Graph.Show;
end;

procedure TW_Mode1.ListBox4Click(Sender: TObject);
begin
  ListBox5.ItemIndex:=ListBox4.ItemIndex;
  ListBox5.Topindex:=Listbox4.TopIndex;
  ListBox1.ItemIndex:=ListBox4.ItemIndex;
  Listbox1.Topindex:=Listbox4.Topindex;
  ListBox2.TopIndex:=ListBox4.Topindex;
  ListBox2.Itemindex:=ListBox4.Itemindex;
end;

procedure TW_Mode1.ListBox5Click(Sender: TObject);
begin
```

```
   ListBox4.ItemIndex:=ListBox5.ItemIndex;
   ListBox4.TopIndex:=ListBox5.TopIndex;
   ListBox1.ItemIndex:=ListBox5.ItemIndex;
   ListBox1.TopIndex:=ListBox5.TopIndex;
   ListBox2.ItemIndex:=ListBox5.ItemIndex;
   ListBox2.TopIndex:=ListBox5.TopIndex;
end;
procedure TW_Mode1.ListBox1Click(Sender: TObject);
begin
  ListBox4.TopIndex:=ListBox1.Topindex;
  ListBox4.Itemindex:=ListBox1.Itemindex;
  ListBox5.TopIndex:=ListBox1.Topindex;
  ListBox5.Itemindex:=ListBox1.Itemindex;
  ListBox2.TopIndex:=ListBox1.Topindex;
  ListBox2.Itemindex:=ListBox1.Itemindex;
end;

procedure TW_Mode1.Edit6Change(Sender: TObject);
var item:integer;
begin
  item:=0;
  if (Edit6.text<>' ') and (Strlen(Pchar(Edit6.text))>0) then
  item:=strtoint(Edit6.Text);
  Edit7.Clear; Edit8.clear;
  if item<Round then
   begin
     Edit7.text:=Listbox4.Items[item];
     Edit8.text:=Listbox5.Items[item];
     Edit11.text:=Listbox2.Items[item];
  end;
end;

procedure TW_Mode1.BitBtn3Click(Sender: TObject);
begin
 Close;
end;

procedure TW_Mode1.BitBtn1Click(Sender: TObject);
begin
 ListBox1.Clear;
 ListBox2.Clear;
 ListBox4.Clear;
 ListBox5.Clear;
 InitAllValue;
 Calculate;
end;

procedure TW_Mode1.FormShow(Sender: TObject);
begin
 BitBtn2.Enabled:=False;
end;

procedure TW_Mode1.ListBox2Click(Sender: TObject);
begin
  ListBox4.TopIndex:=ListBox2.Topindex;
  ListBox4.Itemindex:=ListBox2.Itemindex;
  ListBox5.TopIndex:=ListBox2.Topindex;
  ListBox5.Itemindex:=ListBox2.Itemindex;
 ListBox1.TopIndex:=ListBox2.Topindex;
 ListBox1.Itemindex:=ListBox2.Itemindex;
end;

procedure TW_Mode1.Button2Click(Sender: TObject);
begin
 W_Monitor.ShowModal;
end;

function TW_Mode1.strleng(str : string) : integer;
var i,count: integer;
begin
  count:=0;
  for i:=1 to strlen(pchar(str)) do
    if ((str[i]<>'ั') and (str[i]<>'฿') and (str[i]<>'้') and
        (str[i]<>'๊') and (str[i]<>'่') and (str[i]<>'็') and
        (str[i]<>'๋') and (str[i]<>'๋') and (str[i]<>'ุ') and
        (str[i]<>'^') and (str[i]<>'ี') and (str[i]<>'์')) then count:=count+1;
  strleng:=count;
end;
function TW_Mode1.AddText(str : string;length : integer) : string;
var str2 : string;
    i,textleng: integer;
begin
   str2:='';
   textleng:=length-Trunc(strleng(Pchar(str))*1.65);
   if strleng(Pchar(str)) <1 then str:=' ';
   for i:=0 to Trunc(textleng/2) do   str2:=str2+' ';
   if strlen(pchar(str))>0    then    str2:=str2+str;
  for i:=0 to Trunc(textleng/2) do   str2:=str2+' ';
  AddText:=str2;
end;

procedure TW_Mode1.Button1Click(Sender: TObject);
var i : integer;
    str : string;
begin
 W_Report.Memo1.Clear;
 W_Report.Memo1.Lines.Add('         N                    YT                    YB
N                    YT                    YB');
 W_Report.Memo1.Lines.Add('  ');
  i:=0;

  repeat
    str := AddText(Listbox1.Items[i],20)+'      '+AddText(Listbox4.Items[i],20)+'       '+
AddText(ListBox5.Items[i],20);
    str := str +'       '+ AddText(Listbox1.Items[i+1],20)+'
'+AddText(Listbox4.Items[i+1],20)+'       '+ AddText(ListBox5.Items[i+1],20);
    W_Report.Memo1.Lines.Add(str);
    i:=i+2;
  until i>= Listbox1.Items.Count-1;
 W_Report.ShowModal;
end;

end.

// Program for TypeII Form  (  TypeII.Pas )
```

```
unit TypeII;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, Buttons, ComCtrls,Math;

type
  TW_Mode2 = class(TForm)
    Button1: TButton;
    BitBtn3: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn1: TBitBtn;
    Bevel1: TBevel;
    Bevel2: TBevel;
    Label4: TLabel;
    Label7: TLabel;
    Label1: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label2: TLabel;
    Label11: TLabel;
    Edit4: TEdit;
    Edit5: TEdit;
    ListBox4: TListBox;
    ListBox5: TListBox;
    Edit1: TEdit;
    ListBox1: TListBox;
```

```
Edit6: TEdit;
Edit7: TEdit;
Edit8: TEdit;
ListBox2: TListBox;
Edit11: TEdit;
Label14: TLabel;
Label15: TLabel;
ListBox3: TListBox;
ListBox6: TListBox;
Edit14: TEdit;
Edit15: TEdit;
Label16: TLabel;
Edit16: TEdit;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label3: TLabel;
Label5: TLabel;
Label6: TLabel;
Label17: TLabel;
Label18: TLabel;
Label19: TLabel;
Label10: TLabel;
Label12: TLabel;
EAp1: TEdit;
EBp1: TEdit;
ECp1: TEdit;
EAp2: TEdit;
EBp2: TEdit;
ECp2: TEdit;
```

```
EDp1: TEdit;
EDp2: TEdit;
TabSheet2: TTabSheet;
Label13: TLabel;
Label20: TLabel;
Label21: TLabel;
Label22: TLabel;
Label23: TLabel;
Label24: TLabel;
Label25: TLabel;
Label26: TLabel;
EAn1: TEdit;
EBn1: TEdit;
ECn1: TEdit;
EAn2: TEdit;
EBn2: TEdit;
ECn2: TEdit;
EDn1: TEdit;
EDn2: TEdit;
Label27: TLabel;
Label28: TLabel;
EAn3: TEdit;
EBn3: TEdit;
Label29: TLabel;
Label30: TLabel;
ECn3: TEdit;
EDn3: TEdit;
Label31: TLabel;
Label32: TLabel;
```

```
ECp3: TEdit;
EDp3: TEdit;
Label33: TLabel;
Label34: TLabel;
Eap3: TEdit;
EBp3: TEdit;
KCheckBox: TCheckBox;

procedure SetMatrixDown;
procedure SetMatrixUp;
procedure CalMatrix;
procedure ShiftUp;
procedure ClearMatrix;
procedure ShiftDown;
procedure DownFlow;
procedure InitAllvalue;
procedure UpFlow;
procedure Calculate;
procedure BitBtn2Click(Sender: TObject);
procedure SetRightStep;
procedure ListBox4Click(Sender: TObject);
procedure ListBox5Click(Sender: TObject);
procedure ListBox1Click(Sender: TObject);
procedure Edit6Change(Sender: TObject);
procedure BitBtn3Click(Sender: TObject);
procedure BitBtn1Click(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure ListBox3Click(Sender: TObject);
```

```
    procedure ListBox6Click(Sender: TObject);
    procedure ListBox2Click(Sender: TObject);
    function  Kfunction(y : double):double;
    procedure Button2Click(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  W_Mode2: TW_Mode2;
  MatrixV : Array [0..1,0..100,0..100] of double;
  Ap,Bp,Cp,Dp,An,Bn,Cn,Dn : Array [1..3] of double;
  MatrixY,MatrixSum,MatrixDev : Array [0..1,0..100] of double;
  V,VB,YmAvg,OldYmAvg,Yt1Avg,Yt3Avg,Yb1Avg,Y0,Yb3Avg : double;
  Step,Round,ColumnIndex,ColumnCount,Cycle,Kindex,StepCount:integer;



implementation


uses Graph, Monitor, Printing, TypeI;


{$R *.DFM}


 procedure TW_Mode2.CalMatrix;
 var i,j: integer;
 begin
 for j:=0 to Step do
  begin
   MatrixSum[ColumnIndex,j]:=0;
   for i:=0 to Step do

MatrixSum[ColumnIndex,j]:=MatrixSum[ColumnIndex,j]+MatrixY[ColumnIndex,i]*Matrix
V[ColumnIndex,i,j];
     MatrixSum[ColumnIndex,j]:=MatrixSum[ColumnIndex,j]/MatrixDev[ColumnIndex,j];
  end;
 end;

 procedure TW_Mode2.ClearMatrix; var i,j :integer; begin   for i:=0 to Step do
   for j:=0 to Step do MatrixV[ColumnIndex,i,j]:=0;
 end;

 function TW_Mode2.Kfunction(y : double):double;
  var i : integer;
     YY : double;
  begin
     i:=Kindex;
     if KCheckBox.Checked then YY:=1 else YY:=y;
     if ColumnIndex  = 0 then Kfunction := Ap[i]*Power(y,-2)+Bp[i]*Power(y,-1)+Cp[i]
       +Dp[i]/Power(YY,-1) else          Kfunction :=
An[i]*Power(y,2)+Bn[i]*Power(y,1)+Cn[i]+Dn[i]/Power(YY,1);  end;
procedure TW_Mode2.SetMatrixUp; var i,j: integer; begin

  ClearMatrix;

   for i:=1 to Step do
 MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i];

if ColumnIndex=1 then
begin
 if Cycle=2 then  MatrixY[ColumnIndex,0]:=Yb3Avg
  else if Cycle=4 then  MatrixY[ColumnIndex,0]:=Yb1Avg;
end else  MatrixY[ColumnIndex,0]:=YmAvg;

 if ColumnIndex =1 then Kindex:=2 else
   begin
     if Cycle=2 then Kindex:=1;
     if Cycle=4 then Kindex:=3;
   end;

  i:=0;
for j:= 0 to Step-1 do
   begin
    MatrixV[ColumnIndex,i+1,j]:= VB*Kfunction(MatrixY[ColumnIndex,j+1]);
    MatrixV[ColumnIndex,i,j]:= V;
    i:=i+1;
   end;
   MatrixV[ColumnIndex,Step,Step]:=VB*Kfunction(MatrixY[ColumnIndex,Step])+V;

   for i:=1 to Step-1 do
     MatrixDev[ColumnIndex,i]:= VB*Kfunction(MatrixY[ColumnIndex,i+1])+V;

  if ColumnIndex =0 then Kindex:=2 else
   begin
     if Cycle=2 then Kindex:=3;
```

```
        if Cycle=4 then Kindex:=1;
      end;
       MatrixDev[ColumnIndex,Step]:=MatrixV[ColumnIndex,Step,Step];
       MatrixDev[ColumnIndex,0]:=VB*Kfunction(MatrixY[ColumnIndex,1])+V;

 end;

procedure TW_Mode2.SetMatrixDown;
 var i,j: integer;
 begin

   ClearMatrix;

   for i:=0 to Step-1 do MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i];

   if ColumnIndex=0 then
   begin
     if Cycle=1 then  MatrixY[ColumnIndex,Step]:=Yt1Avg
      else if Cycle=3 then  MatrixY[ColumnIndex,Step]:=Yt3Avg;
   end else  MatrixY[ColumnIndex,Step]:=YmAvg;

   if (ColumnIndex mod 2)=0 then Kindex:=2 else
      begin
        if Cycle=1 then Kindex:=1;
        if Cycle=3 then Kindex:=3;
      end;
      i:=0;
   for j:= 1 to Step do
      begin
      MatrixV[ColumnIndex,i,j]:= VB*Kfunction(MatrixY[ColumnIndex,j-1]);
      MatrixV[ColumnIndex,i+1,j]:= V;
      i:=i+1;
     end;
    MatrixV[ColumnIndex,0,0]:=VB*Kfunction(MatrixY[ColumnIndex,0])+V;

   for i:=1 to Step-1 do
    MatrixDev[ColumnIndex,i]:= V+VB*Kfunction(MatrixY[ColumnIndex,i-1]);

    if ColumnIndex =1 then Kindex:=2 else
     begin
      if Cycle=1 then Kindex:=1;
      if Cycle=3 then Kindex:=3;
     end;

     MatrixDev[ColumnIndex,0]:=MatrixV[ColumnIndex,0,0];
     MatrixDev[ColumnIndex,Step]:=V+VB*Kfunction(MatrixY[ColumnIndex,Step-1]);

 end;

procedure TW_Mode2.ShiftUp;
var i,j: integer;
 begin

  for i:=0 to Step-1 do MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i+1];

  if ColumnIndex=0 then
    begin
     if Cycle=1 then  MatrixY[ColumnIndex,Step]:=Yt1Avg else
   if Cycle=3 then  MatrixY[ColumnIndex,Step]:=Yt3Avg;
 end else  MatrixY[ColumnIndex,Step]:=YmAvg;
  i:=0;
for j:= 1 to Step do
  begin

 if ColumnIndex =0 then
      begin
        if Cycle=1 then
          begin
            if j>Step-StepCount+1 then   Kindex:=1 else Kindex:=2;
          end else
          begin
            if j>Step-StepCount+1 then   Kindex:=3 else Kindex:=2;
          end;
      end else
      begin
       if Cycle=1 then
        begin
          if j>Step-StepCount+1 then   Kindex:=2 else Kindex:=1;
        end else
        begin
          if j>Step-StepCount+1 then   Kindex:=2 else Kindex:=3;
        end
      end;

    MatrixV[ColumnIndex,i,j]:= VB*Kfunction(MatrixY[ColumnIndex,j-1]);
    MatrixV[ColumnIndex,i+1,j]:= V;
    i:=i+1;
  end;
 if (ColumnIndex mod 2)=0 then Kindex:=2 else
  begin
    if Cycle=1 then Kindex:=1;
    if Cycle=3 then Kindex:=3;
  end;
 MatrixV[ColumnIndex,0,0]:=VB*Kfunction(MatrixY[ColumnIndex,0])+V;


for i:=1 to Step do
 begin
     if ColumnIndex =0 then
      begin
         if Cycle=1 then
           begin
             if i>Step-StepCount then   Kindex:=1 else Kindex:=2;
           end else
           begin
             if i>Step-StepCount then   Kindex:=3 else Kindex:=2;
           end;
      end else
      begin
        if Cycle=1 then
         begin
           if i>Step-StepCount then   Kindex:=2 else Kindex:=1;
         end else
         begin
           if i>Step-StepCount then   Kindex:=2 else Kindex:=3;
         end
      end;
      MatrixDev[ColumnIndex,i]:= V+VB*Kfunction(MatrixY[ColumnIndex,i-1]);
    end;
      MatrixDev[ColumnIndex,0]:=MatrixV[ColumnIndex,0,0];

 end;

procedure TW_Mode2.ShiftDown;
var i,j : integer;
begin

   for i:=step downto 1 do  MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i-1];

   if ColumnIndex=1 then
   begin
     if Cycle=2 then MatrixY[ColumnIndex,0]:=Yb3Avg else
      if Cycle=4 then MatrixY[ColumnIndex,0]:=Yb1Avg;
   end else   MatrixY[ColumnIndex,0]:=YmAvg;

     i:=0;
   for j:= 0 to Step-1 do
      begin

       if ColumnIndex  =1 then
       begin
        if Cycle=2 then
          begin
            if j<StepCount-1 then  Kindex:=3 else Kindex:=2;
          end else
          begin
    if j<StepCount-1 then  Kindex:=1 else Kindex:=2;
   end;
 end else
 begin
  if Cycle=2 then
   begin
    if j<StepCount-1 then  Kindex:=2 else Kindex:=1;
   end else
   begin
    if j<StepCount-1 then  Kindex:=2 else Kindex:=3;
   end;
 end;

 MatrixV[ColumnIndex,i+1,j]:= VB*Kfunction(MatrixY[ColumnIndex,j+1]);
 MatrixV[ColumnIndex,i,j]:= V;
 i:=i+1;
end;
if ColumnIndex =1 then Kindex:=2 else
  begin
    if Cycle=2 then Kindex:=1 else Kindex:=3;
  end;

MatrixV[ColumnIndex,Step,Step]:=VB*Kfunction(MatrixY[ColumnIndex,Step])+V;

for i:=0 to Step-1 do
begin
  if ColumnIndex  =1 then
 begin
  if Cycle=2 then
```

```
     begin
      if i<StepCount then  Kindex:=3 else Kindex:=2;
     end else
     begin
      if i<StepCount then  Kindex:=1 else Kindex:=2;
     end;
    end else
    begin
     if Cycle=2 then
      begin
       if i<StepCount then  Kindex:=2 else Kindex:=1;
      end else
      begin
       if i<StepCount then  Kindex:=2 else Kindex:=3;
      end;
    end;
     MatrixDev[ColumnIndex,i]:= VB*Kfunction(MatrixY[ColumnIndex,i+1])+V;
   end;

    MatrixDev[ColumnIndex,Step]:=MatrixV[ColumnIndex,Step,Step];

end;

procedure TW_Mode2.DownFlow;var i: integer;    averg1: double;begin  StepCount:=1;
ColumnIndex:=0;
  SetMatrixDown;
  ColumnIndex:=1;
  SetMatrixDown;
```

```
 averg1:=0;
 for i:=0 to Step-1 do
  begin
  ColumnIndex:=0;
  Inc(StepCount);
  CalMatrix;
  ShiftUp;
  YmAvg:=OldYmAvg;
  OldYmAvg:=(OldYmAvg*(Step*2-1)+MatrixSum[0,0])/(2*Step);
  ColumnIndex:=1;
  CalMatrix;
  ShiftUp;
  averg1:=averg1+MatrixSum[ColumnIndex,0];
  end;

  if Cycle=1 then   Yb1Avg:=((Yb1Avg*Step)+averg1)/(2*Step)
   else if Cycle=3 then Yb3Avg:=((Yb3Avg*Step)+averg1)/(2*Step);

end;

procedure TW_Mode2.UpFlow;
var i: integer;
    averg0: double;
begin
  StepCount:=1;
  ColumnIndex:=0;
  SetMatrixUp;
  ColumnIndex:=1;
  SetMatrixUp;
 averg0:=0;
 for i:=0 to Step-1 do
  begin
  ColumnIndex:=1;
  Inc(StepCount);
  CalMatrix;
  ShiftDown;
  YmAvg:=OldYmAvg;
  OldYmAvg:=(OldYmAvg*(Step*2-1)+MatrixSum[1,Step])/(2*Step);
  ColumnIndex:=0;
  CalMatrix;
  ShiftDown;
  averg0:=averg0+MatrixSum[ColumnIndex,Step];
  end;
  if Cycle=2 then  Yt1Avg:=((Yt1Avg*Step)+averg0)/(2*Step)
    else if Cycle=4 then  Yt3Avg:=((Yt3Avg*Step)+averg0)/(2*Step);
end;

procedure TW_Mode2.InitAllValue;var i,j: integer;begin  ColumnCount:=2;
BitBtn2.Enabled:=True;  Ap[1]:=strtofloat(EAp1.Text);  Bp[1]:=strtofloat(EBp1.Text);
  Cp[1]:=strtofloat(ECp1.Text);
  Dp[1]:=strtofloat(EDp1.Text);
  Ap[2]:=strtofloat(EAp2.Text);
  Bp[2]:=strtofloat(EBp2.Text);
  Cp[2]:=Strtofloat(ECp2.Text);
  Dp[2]:=strtofloat(EDp2.Text);
  Ap[3]:=strtofloat(EAp3.Text);
  Bp[3]:=strtofloat(EBp3.Text);
  Cp[3]:=Strtofloat(ECp3.Text);
```

```
Dp[3]:=strtofloat(EDp3.Text);

An[1]:=strtofloat(EAn1.Text);
Bn[1]:=strtofloat(EBn1.Text);
Cn[1]:=strtofloat(ECn1.Text);
Dn[1]:=strtofloat(EDn1.Text);
An[2]:=strtofloat(EAn2.Text);
Bn[2]:=strtofloat(EBn2.Text);
Cn[2]:=Strtofloat(ECn2.Text);
Dn[2]:=strtofloat(EDn2.Text);
An[3]:=strtofloat(EAn3.Text);
Bn[3]:=strtofloat(EBn3.Text);
Cn[3]:=Strtofloat(ECn3.Text);
Dn[3]:=strtofloat(EDn3.Text);

V:=strtofloat(Edit4.Text);
Y0:=strtofloat(Edit16.Text);
OldYmAvg:=Y0;
Yt1Avg:=Y0;  Yb1Avg:=Y0; YmAvg:=Y0;
Yt3Avg:=Y0;  Yb3Avg:=Y0;
Cycle:=1;
if strlen(Pchar(Edit5.text))<1 then SetRightStep else
 begin
   if strtoint(Edit5.Text)<2 then Edit5.Text:='2';
   Step:=Strtoint(Edit5.Text);
 end;

OldYmAvg:=Y0;
Yt1Avg:=Y0;  Yb1Avg:=Y0; YmAvg:=Y0;
```

```
 Yt3Avg:=Y0;  Yb3Avg:=Y0;

 V:=strtofloat(Edit4.text)/(2*Step);
 VB:=V/3;

 for j:=0 to ColumnCount-1 do
 begin
  for i:=0 to Step do
  begin
    MatrixY[j,i]:=Y0;
    MatrixSum[j,i]:=Y0;
  end;
 end;

end;

procedure TW_Mode2.SetRightStep;
var i,j: integer;
begin

 Step:=4;
 repeat
  Step:=Step+1;
  V:=strtofloat(Edit4.text)/(2*Step);
  VB:=V/3;
   for j:=0 to ColumnCount-1 do
   begin
     for i:=0 to Step do
       begin
```

```
            MatrixY[j,i]:=Y0;
            MatrixSum[j,i]:=Y0;
           end;
       end;
      DownFlow;
    until MatrixSum[0,0]>0.94;
    Edit5.Text:=inttostr(Step);
end;
procedure TW_Mode2.Calculate;
var i :integer;
   last0, last1 : double;
   stop:boolean;
begin

   ListBox4.Clear; ListBox5.clear; ListBox1.Clear;
   ListBox2.Clear;
   stop:=false; last0:=0; last1:=0;
   i:=0;
     ListBox1.Items.Add(inttostr(i));
     ListBox4.Items.Add(floattostrf(Yt1Avg,ffGeneral,6,6));
     ListBox3.Items.Add(floattostrf(Yb3Avg,ffGeneral,6,6));
     ListBox6.Items.Add(floattostrf(Yt3Avg,ffGeneral,6,6));
     ListBox2.Items.Add(floattostrf(YmAvg,ffGeneral,6,6));
   repeat

     i:=i+1;
     Cycle:=1;
     DownFlow;
```

```
  ListBox5.Items.Add(floattostrf(Yb1Avg,ffGeneral,6,6));

  Cycle:=2;
  UpFlow;
  ListBox4.Items.Add(floattostrf(Yt1Avg,ffGeneral,6,6));

   Cycle:=3;
  DownFlow;

  ListBox6.Items.Add(floattostrf(Yb3Avg,ffGeneral,6,6));

  Cycle:=4;
  UpFlow;
  ListBox3.Items.Add(floattostrf(Yt3Avg,ffGeneral,6,6));

  ListBox2.Items.Add(floattostrf(YmAvg,ffGeneral,6,6));

  if Abs(Yb1Avg-Last0)<0.00001 then Stop:=True;
  last0:=Yb1Avg;
  if Stop and (Abs(Yt1Avg-Last1)>0.00001) then Stop:=False;
  last1:=Yt1Avg;

  ListBox1.Items.Add(inttostr(i));
until Stop or (i>100);
Cycle:=1;
DownFlow;
ListBox5.Items.Add(floattostrf(Yb1Avg,ffGeneral,6,6));
Edit1.text:=inttostr(i);
Round:=i;
```

```
end;

procedure TW_Mode2.BitBtn2Click(Sender: TObject);
begin
  Graph.Plottype:=2;
  W_Graph.Show;
end;

procedure TW_Mode2.ListBox4Click(Sender: TObject);
begin
  ListBox5.ItemIndex:=ListBox4.ItemIndex;
  ListBox5.Topindex:=Listbox4.TopIndex;
  ListBox1.ItemIndex:=ListBox4.ItemIndex;
  Listbox1.Topindex:=Listbox4.Topindex;
  ListBox3.ItemIndex:=ListBox4.ItemIndex;
  ListBox3.Topindex:=Listbox4.TopIndex;
  ListBox6.ItemIndex:=ListBox4.ItemIndex;
  Listbox6.Topindex:=Listbox4.Topindex;
  ListBox2.ItemIndex:=ListBox4.ItemIndex;
  Listbox2.Topindex:=Listbox4.Topindex;
end;

procedure TW_Mode2.ListBox5Click(Sender: TObject);
begin
  ListBox4.ItemIndex:=ListBox5.ItemIndex;
  ListBox4.Topindex:=Listbox5.TopIndex;
  ListBox1.ItemIndex:=ListBox5.ItemIndex;
  Listbox1.Topindex:=Listbox5.Topindex;
  ListBox3.ItemIndex:=ListBox5.ItemIndex;
  ListBox3.Topindex:=Listbox5.TopIndex;
  ListBox6.ItemIndex:=ListBox5.ItemIndex;
  Listbox6.Topindex:=Listbox5.Topindex;
  ListBox2.ItemIndex:=ListBox5.ItemIndex;
  Listbox2.Topindex:=Listbox5.Topindex;
end;

procedure TW_Mode2.ListBox1Click(Sender: TObject);
begin
  ListBox5.ItemIndex:=ListBox1.ItemIndex;
  ListBox5.Topindex:=Listbox1.TopIndex;
  ListBox4.ItemIndex:=ListBox1.ItemIndex;
  Listbox4.Topindex:=Listbox1.Topindex;
  ListBox3.ItemIndex:=ListBox1.ItemIndex;
  ListBox3.Topindex:=Listbox1.TopIndex;
  ListBox6.ItemIndex:=ListBox1.ItemIndex;
  Listbox6.Topindex:=Listbox1.Topindex;
  ListBox2.ItemIndex:=ListBox1.ItemIndex;
  Listbox2.Topindex:=Listbox1.Topindex;
end;

procedure TW_Mode2.Edit6Change(Sender: TObject);
var item:integer;
begin
  item:=0;
  if (Edit6.text<>' ') and (Strlen(Pchar(Edit6.text))>0) then
  item:=strtoint(Edit6.Text);
  Edit7.Clear; Edit8.clear;
  if item<Round then
  begin
    Edit7.text:=Listbox4.Items[item];
    Edit8.text:=Listbox5.Items[item];
    Edit15.text:=Listbox3.Items[item];
    Edit14.text:=Listbox6.Items[item];
    Edit11.text:=Listbox2.Items[item];
  end;
end;
procedure TW_Mode2.BitBtn3Click(Sender: TObject);
begin
 Close;
end;
procedure TW_Mode2.BitBtn1Click(Sender: TObject);
begin
 ListBox1.Clear;
 ListBox2.Clear;
 ListBox4.Clear;
 ListBox5.Clear;
 ListBox3.Clear;
 ListBox6.Clear;
 InitAllValue;
 Calculate;
end;

procedure TW_Mode2.Button1Click(Sender: TObject);
var i : integer;
   str : string;
begin
 W_Report.Memo1.Clear;
 W_Report.Memo1.Lines.Add('        N                  YT1                 YB1
YT3                 YB3                 YM');
 W_Report.Memo1.Lines.Add(' ');
  i:=0;

  repeat
    str := W_Mode1.AddText(Listbox1.Items[i],20)+'     '+W_Mode1.AddText(
            Listbox4.Items[i],20)+'      '+ W_Mode1.AddText(ListBox5.Items[i],20);
    str := str +'            '+ W_Mode1.AddText(Listbox3.Items[i],20)+'
'+W_Mode1.AddText
            (Listbox6.Items[i],20)+'      '+ W_Mode1.AddText(ListBox2.Items[i],20);
    W_Report.Memo1.Lines.Add(str);
    i:=i+1;
  until i>= Listbox1.Items.Count-1;
 W_Report.ShowModal;
end;

procedure TW_Mode2.FormShow(Sender: TObject);
begin

 BitBtn2.Enabled:=False;
end;

procedure TW_Mode2.ListBox3Click(Sender: TObject);
begin
  ListBox5.ItemIndex:=ListBox3.ItemIndex;
  ListBox5.Topindex:=Listbox3.TopIndex;
  ListBox1.ItemIndex:=ListBox3.ItemIndex;
  Listbox1.Topindex:=Listbox3.Topindex;
 ListBox4.ItemIndex:=ListBox3.ItemIndex;
 ListBox4.Topindex:=Listbox3.TopIndex;
 ListBox6.ItemIndex:=ListBox3.ItemIndex;
 Listbox6.Topindex:=Listbox3.Topindex;
 ListBox2.ItemIndex:=ListBox3.ItemIndex;
 Listbox2.Topindex:=Listbox3.Topindex;
end;

procedure TW_Mode2.ListBox6Click(Sender: TObject);
begin
 ListBox5.ItemIndex:=ListBox6.ItemIndex;
 ListBox5.Topindex:=Listbox6.TopIndex;
 ListBox1.ItemIndex:=ListBox6.ItemIndex;
 Listbox1.Topindex:=Listbox6.Topindex;
 ListBox3.ItemIndex:=ListBox6.ItemIndex;
 ListBox3.Topindex:=Listbox6.TopIndex;
 ListBox4.ItemIndex:=ListBox6.ItemIndex;
 Listbox4.Topindex:=Listbox6.Topindex;
 ListBox2.ItemIndex:=ListBox6.ItemIndex;
 Listbox2.Topindex:=Listbox6.Topindex;
end;

procedure TW_Mode2.ListBox2Click(Sender: TObject);
begin
 ListBox5.ItemIndex:=ListBox2.ItemIndex;
 ListBox5.Topindex:=Listbox2.TopIndex;
 ListBox1.ItemIndex:=ListBox2.ItemIndex;
 Listbox1.Topindex:=Listbox2.Topindex;
 ListBox3.ItemIndex:=ListBox2.ItemIndex;
 ListBox3.Topindex:=Listbox2.TopIndex;
 ListBox6.ItemIndex:=ListBox2.ItemIndex;
 Listbox6.Topindex:=Listbox2.Topindex;
 ListBox4.ItemIndex:=ListBox2.ItemIndex;
 Listbox4.Topindex:=Listbox2.Topindex;
end;

procedure TW_Mode2.Button2Click(Sender: TObject);
begin
 W_Monitor.ShowModal;
end;
end.
```

// Program for TypeIII Form (TypeIII.Pas)

```
unit TypeIII;interfaceuses  Windows, Messages, SysUtils, Classes, Graphics, Controls,
Forms, Dialogs,  StdCtrls, ExtCtrls, Buttons, ComCtrls,Math;type  TW_Mode3 =
class(TForm)    Bevel1: TBevel;
    Bevel2: TBevel;
    Label4: TLabel;
    Label7: TLabel;
    Label1: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label2: TLabel;
    Label11: TLabel;
    Button1: TButton;
    BitBtn3: TBitBtn;
```

```
BitBtn2: TBitBtn;
BitBtn1: TBitBtn;
Edit4: TEdit;
Edit5: TEdit;
ListBox4: TListBox;
ListBox5: TListBox;
Edit1: TEdit;
ListBox1: TListBox;
Edit6: TEdit;
Edit7: TEdit;
Edit8: TEdit;
ListBox2: TListBox;
Edit11: TEdit;
Label12: TLabel;
ListBox3: TListBox;
Edit12: TEdit;
Label15: TLabel;
Edit15: TEdit;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label3: TLabel;
Label5: TLabel;
Label6: TLabel;
Label17: TLabel;
Label18: TLabel;
Label19: TLabel;
Label10: TLabel;
Label13: TLabel;
Label31: TLabel;
```

```
Label32: TLabel;
Label33: TLabel;
Label34: TLabel;
EAp1: TEdit;
EBp1: TEdit;
ECp1: TEdit;
EAp2: TEdit;
EBp2: TEdit;
ECp2: TEdit;
EDp1: TEdit;
EDp2: TEdit;
ECp3: TEdit;
EDp3: TEdit;
Eap3: TEdit;
EBp3: TEdit;
TabSheet2: TTabSheet;
Label14: TLabel;
Label20: TLabel;
Label21: TLabel;
Label22: TLabel;
Label23: TLabel;
Label24: TLabel;
Label25: TLabel;
Label26: TLabel;
Label27: TLabel;
Label28: TLabel;
Label29: TLabel;
Label30: TLabel;
EAn1: TEdit;
```

```
EBn1: TEdit;
ECn1: TEdit;
EAn2: TEdit;
EBn2: TEdit;
ECn2: TEdit;
EDn1: TEdit;
EDn2: TEdit;
EAn3: TEdit;
EBn3: TEdit;
ECn3: TEdit;
EDn3: TEdit;
KCheckBox: TCheckBox;

procedure SetMatrixDown;
procedure SetMatrixUp;
procedure CalMatrix;
procedure ShiftUp;
procedure ClearMatrix;
procedure ShiftDown;
procedure DownFlow;
procedure InitAllvalue;
procedure UpFlow;
procedure Calculate;
procedure BitBtn2Click(Sender: TObject);
procedure SetRightStep;
procedure ListBox4Click(Sender: TObject);
procedure ListBox5Click(Sender: TObject);
procedure ListBox1Click(Sender: TObject);
procedure Edit6Change(Sender: TObject);
```

```
    procedure BitBtn3Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure ListBox2Click(Sender: TObject);
    procedure ListBox3Click(Sender: TObject);
    function  Kfunction(y : double):double;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  W_Mode3: TW_Mode3;
  MatrixV : Array [0..1,0..100,0..100] of double;
  Ap,Bp,Cp,Dp,An,Bn,Cn,Dn : Array [1..3] of double;
  MatrixY,MatrixSum,MatrixDev : Array [0..1,0..100] of double;
  V,VB,Y0,Ym1Avg,OldYm1Avg,
  Ym3Avg,OldYm3Avg,YtAvg,YbAvg : double;
  Step,Round,ColumnIndex,Cycle,Kindex,StepCount:integer;

implementation

uses Graph, Monitor, Printing, Typel;

{$R *.DFM}

 procedure TW_Mode3.CalMatrix;
```

```
var i,j: integer;
begin

 for j:=0 to Step do
  begin
   MatrixSum[ColumnIndex,j]:=0;
   for i:=0 to Step do
     MatrixSum[ColumnIndex,j]:=MatrixSum[ColumnIndex,j]+MatrixY[ColumnIndex,i]
        *MatrixV[ColumnIndex,i,j];
MatrixSum[ColumnIndex,j]:=MatrixSum[ColumnIndex,j]/MatrixDev[ColumnIndex,j];
  end;
end;

procedure TW_Mode3.ClearMatrix;
var i,j :integer;
begin
  for i:=0 to Step do
  for j:=0 to Step do MatrixV[ColumnIndex,i,j]:=0;
end;
function TW_Mode3.Kfunction(y : double):double;
 var i : integer;
     YY : double;
 begin
     i:=Kindex;
     if KCheckBox.Checked then YY:=1 else YY:=y;
     if ColumnIndex  = 0 then Kfunction := Ap[i]*Power(y,-2)+Bp[i]*Power(y,-1)+Cp[i]
        +Dp[i]/Power(YY,-1) else         Kfunction :=
An[i]*Power(y,2)+Bn[i]*Power(y,1)+Cn[i]+Dn[i]/Power(YY,1);
 end;
```

```
procedure TW_Mode3.SetMatrixUp;
 var i,j: integer;
 begin

   ClearMatrix;

    for i:=1 to Step do   MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i];

   if ColumnIndex=1 then  MatrixY[ColumnIndex,0]:=YbAvg
     else  begin
            if Cycle=2 then MatrixY[ColumnIndex,0]:=Ym3Avg
              else MatrixY[ColumnIndex,0]:=Ym1Avg;
          end;

   if ColumnIndex =1 then
         begin
           if Cycle= 2 then  Kindex:=3  else   Kindex:=1;
         end else Kindex:=2;

     i:=0;
   for j:= 0 to Step-1 do
    begin
     MatrixV[ColumnIndex,i+1,j]:= VB*Kfunction(MatrixY[ColumnIndex,j+1]);
     MatrixV[ColumnIndex,i,j]:= V;
     i:=i+1;
    end;

   MatrixV[ColumnIndex,Step,Step]:=VB*Kfunction(MatrixY[ColumnIndex,Step])+V;
```

```
  for i:=1 to Step-1 do   MatrixDev[ColumnIndex,i]:=
                 VB*Kfunction(MatrixY[ColumnIndex,i+1])+V;   if ColumnIndex =1 then
Kindex:=2 else
             Begin
               if Cycle=2 then Kindex:=3 else
                  Kindex:=1;
             End;

  MatrixDev[ColumnIndex,Step]:=MatrixV[ColumnIndex,Step,Step];
  MatrixDev[ColumnIndex,0]:=VB*Kfunction(MatrixY[ColumnIndex,1])+V;

 end;

procedure TW_Mode3.SetMatrixDown;
 var i,j: integer;
 begin

  ClearMatrix;

  for i:=0 to Step-1 do  MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i];
  if ColumnIndex=0 then  MatrixY[ColumnIndex,Step]:=YtAvg
   else begin
        if Cycle=1 then MatrixY[ColumnIndex,Step]:=Ym3Avg else
        if Cycle=3 then MatrixY[ColumnIndex,Step]:=Ym1Avg;
      end;

  if (ColumnIndex mod 2)=0 then
    begin
    if Cycle=1 then  Kindex:=1 else Kindex:=3;
    End else Kindex:=2;

   i:=0;
  for j:= 1 to Step do
   begin
    MatrixV[ColumnIndex,i,j]:=  VB*Kfunction(MatrixY[ColumnIndex,j-1]);
    MatrixV[ColumnIndex,i+1,j]:= V;
    i:=i+1;
   end;
   MatrixV[ColumnIndex,0,0]:=VB*Kfunction(MatrixY[ColumnIndex,0])+V;

   for i:=1 to Step-1 do MatrixDev[ColumnIndex,i]:=
              V+VB*Kfunction(MatrixY[ColumnIndex,i-1]);   if (ColumnIndex mod 2)=0 then
Kindex:=2 else      Begin       if Cycle=1 then  Kindex:=3 else  Kindex:=1;      end;
MatrixDev[ColumnIndex,0]:=MatrixV[ColumnIndex,0,0];
   MatrixDev[ColumnIndex,Step]:=V+VB*Kfunction(MatrixY[ColumnIndex,Step-1]);

 end;

procedure TW_Mode3.ShiftUp;
var i,j : integer;
begin

  for i:=0 to Step-1 do  MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i+1];

  if ColumnIndex=0 then MatrixY[ColumnIndex,Step]:=YtAvg else
  begin
     if Cycle=1 then MatrixY[ColumnIndex,Step]:=Ym3Avg else
      if Cycle = 3 then MatrixY[ColumnIndex,Step] :=Ym1Avg;
```

```
end;

 i:=0;
for j:= 1 to Step do
 begin

  if ColumnIndex =0 then
      begin
        if Cycle=1 then
          begin
            if j>Step-StepCount+1 then   Kindex:=2 else Kindex:=1;
          end else
          begin
            if j>Step-StepCount+1 then   Kindex:=2 else Kindex:=3;
          end;
      end else
      begin
       if Cycle=1 then
        begin
          if j>Step-StepCount+1 then   Kindex:=3 else Kindex:=2;
        end else
        begin
          if j>Step-StepCount+1 then   Kindex:=1 else Kindex:=2;
        end
      end;

  MatrixV[ColumnIndex,i,j]:=  VB*Kfunction(MatrixY[ColumnIndex,j-1]);
  MatrixV[ColumnIndex,i+1,j]:= V;
  i:=i+1;
  end;
 if ColumnIndex =0 then
  begin
    if Cycle=1 then Kindex:=1 else Kindex:=3;
  end   else Kindex:=2;

 MatrixV[ColumnIndex,0,0]:=VB*Kfunction(MatrixY[ColumnIndex,0])+V;


 for i:=1 to Step do
  begin
  if ColumnIndex =0 then
      begin
        if Cycle=1 then
         begin
           if i>Step-StepCount then   Kindex:=2 else Kindex:=1;
         end else
         begin
           if i>Step-StepCount then   Kindex:=2 else Kindex:=3;
         end;
      end else
      begin
        if Cycle=1 then
        begin
          if i>Step-StepCount then   Kindex:=3 else Kindex:=2;
        end else
        begin
          if i>Step-StepCount then   Kindex:=1 else Kindex:=2;
        end
      end;
     MatrixDev[ColumnIndex,i]:= V+VB*Kfunction(MatrixY[ColumnIndex,i-1]);
    end;

  MatrixDev[ColumnIndex,0]:=MatrixV[ColumnIndex,0,0];

end;

procedure TW_Mode3.ShiftDown;
var i,j : integer;
begin

  i:=Step;
  while i>0 do
  begin
    MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i-1];
    i:=i-1;
  end;

  if ColumnIndex=1 then MatrixY[ColumnIndex,0]:=YbAvg else
  begin
    if Cycle = 2 then MatrixY[ColumnIndex,0]:=Ym3Avg else
    if Cycle = 4 then MatrixY[ColumnIndex,0]:=Ym1Avg;
  end;
   i:=0;
  for j:= 0 to Step-1 do
   begin
   if ColumnIndex  =0 then
     begin
  if Cycle=2 then
    begin
      if j<StepCount-1 then  Kindex:=3 else Kindex:=2;
    end else
    begin
      if j<StepCount-1 then  Kindex:=1 else Kindex:=2;
    end;
  end else
  begin
    if Cycle=2 then
     begin
       if j<StepCount-1 then  Kindex:=2 else Kindex:=3;
     end else
     begin
       if j<StepCount-1 then  Kindex:=2 else Kindex:=1;
     end;
  end;

 MatrixV[ColumnIndex,i+1,j]:= VB*Kfunction(MatrixY[ColumnIndex,j+1]);
 MatrixV[ColumnIndex,i,j]:= V;
 i:=i+1;
end;
   if ColumnIndex =0 then Kindex:=2 else
    begin
      if Cycle=2 then Kindex:=3 else Kindex:=1;
    end;
MatrixV[ColumnIndex,Step,Step]:=VB*Kfunction(MatrixY[ColumnIndex,Step])+V;

for i:=0 to Step-1 do
```

```
    begin
       if ColumnIndex  =0 then
      begin
        if Cycle=2 then
        begin
          if i<StepCount then  Kindex:=3 else Kindex:=2;
        end else
        begin
          if i<StepCount then  Kindex:=1 else Kindex:=2;
        end;
      end else
      begin
        if Cycle=2 then
         begin
          if i<StepCount then  Kindex:=2 else Kindex:=3;
         end else
         begin
          if i<StepCount then  Kindex:=2 else Kindex:=1;
         end;
      end;

      MatrixDev[ColumnIndex,i]:= VB*Kfunction(MatrixY[ColumnIndex,i+1])+V;
    end;

   MatrixDev[ColumnIndex,Step]:=MatrixV[ColumnIndex,Step,Step];
end;
procedure TW_Mode3.DownFlow;
var i: integer;
    averg1,averg0: double;
```

```
begin
 ColumnIndex:=0;
  StepCount:=1;
 SetMatrixDown;
 ColumnIndex:=1;
 SetMatrixDown;

 averg0:=0;
 averg1:=0;
 for i:=0 to Step-1 do
  begin
  Inc(StepCount);
  ColumnIndex:=0;
  CalMatrix;
   ShiftUp;

   averg0:=averg0+MatrixSum[0,0];
   ColumnIndex:=1;
  CalMatrix;
  ShiftUp;
  averg1:=averg1+MatrixSum[1,0];
  end;
  YbAvg:=((YbAvg*Step)+averg1)/(2*Step);
  if Cycle = 1 then
    begin
      Ym1Avg:=((Ym1Avg*Step)+averg0)/(2*Step);
      OldYm1Avg:=Ym1Avg;
    end else  if Cycle = 3 then
      begin
```

```
          Ym3Avg:=((Ym3Avg*Step)+averg0)/(2*Step);
          OldYm3Avg:=Ym3Avg;
        end;
end;

procedure TW_Mode3.UpFlow;
var i: integer;
    averg0: double;
begin
  StepCount:=1;
  ColumnIndex:=1;
  SetMatrixUp;

  averg0:=0;
  for i:=0 to Step-1 do
   begin
     Inc(StepCount);
     ColumnIndex:=1;
     CalMatrix;
     ShiftDown;
   if Cycle = 2 then
   begin
     Ym3Avg:=OldYm3Avg;
     OldYm3Avg:=(OldYm3Avg*(Step*2-1)+MatrixSum[1,Step])/(2*Step);
   end else if Cycle = 4 then
   begin
     Ym1Avg:=OldYm1Avg;
     OldYm1Avg:=(OldYm1Avg*(Step*2-1)+MatrixSum[1,Step])/(2*Step);
   end;
  ColumnIndex:=0;

  if i= 0 then SetMatrixUp;
  if Cycle=2 then MatrixY[ColumnIndex,0]:=Ym3Avg else
     if Cycle=4 then MatrixY[ColumnIndex,0]:=Ym1Avg;
  CalMatrix;
  ShiftDown;
  averg0:=averg0+MatrixSum[0,Step];
  end;

  YtAvg:=((YtAvg*Step)+averg0)/(2*Step);
end;

procedure TW_Mode3.InitAllValue;
var i,j: integer;
begin

  BitBtn2.Enabled:=True;

  Ap[1]:=strtofloat(EAp1.Text);
  Bp[1]:=strtofloat(EBp1.Text);
  Cp[1]:=strtofloat(ECp1.Text);
  Dp[1]:=strtofloat(EDp1.Text);
  Ap[2]:=strtofloat(EAp2.Text);
  Bp[2]:=strtofloat(EBp2.Text);
  Cp[2]:=Strtofloat(ECp2.Text);
  Dp[2]:=strtofloat(EDp2.Text);
  Ap[3]:=strtofloat(EAp3.Text);
  Bp[3]:=strtofloat(EBp3.Text);
```

```
Cp[3]:=Strtofloat(ECp3.Text);
Dp[3]:=strtofloat(EDp3.Text);

An[1]:=strtofloat(EAn1.Text);
Bn[1]:=strtofloat(EBn1.Text);
Cn[1]:=strtofloat(ECn1.Text);
Dn[1]:=strtofloat(EDn1.Text);
An[2]:=strtofloat(EAn2.Text);
Bn[2]:=strtofloat(EBn2.Text);
Cn[2]:=Strtofloat(ECn2.Text);
Dn[2]:=strtofloat(EDn2.Text);
An[3]:=strtofloat(EAn3.Text);
Bn[3]:=strtofloat(EBn3.Text);
Cn[3]:=Strtofloat(ECn3.Text);
Dn[3]:=strtofloat(EDn3.Text);

Y0:=strtofloat(Edit15.Text);
OldYm1Avg:=Y0;
OldYm3Avg:=Y0;
YtAvg:=Y0;
Ym1Avg:=Y0;
Ym3Avg:=Y0;
YbAvg:=Y0;
if strlen(Pchar(Edit5.text))<1 then SetRightStep else
 begin
  if strtoint(Edit5.Text)<2 then Edit5.Text:='2';
  Step:=Strtoint(Edit5.Text);
 end;
 OldYm1Avg:=Y0;
 OldYm3Avg:=Y0;
 Ym1Avg:=Y0;
 Ym3Avg:=Y0;
 YtAvg:=Y0;
 YbAvg:=Y0;

 V:=strtofloat(Edit4.text)/(2*Step);
 VB:=V/3;

 for j:=0 to 1 do
 begin
  for i:=0 to Step do
  begin
   MatrixY[j,i]:=Y0;
   MatrixSum[j,i]:=Y0;
  end;
 end;

end;

procedure TW_Mode3.SetRightStep;var i,j: integer;begin
 Step:=4;
 repeat
  Step:=Step+1;
  V:=strtofloat(Edit4.text)/(2*Step);
  VB:=V/3;
```

```
        for i:=0 to Step do
          begin
            MatrixY[j,i]:=Y0;
            MatrixSum[j,i]:=Y0;
          end;
      end;
    DownFlow;
  until MatrixSum[0,0]>0.94;

  Edit5.Text:=inttostr(Step);

end;

procedure TW_Mode3.Calculate;
var i :integer;
  last0, last1 : double;
  stop:boolean;
begin

  ListBox4.Clear; ListBox5.clear; ListBox1.Clear;
  ListBox2.Clear;
  stop:=false; last0:=0; last1:=0;
  i:=1;
    ListBox1.Items.Add(inttostr(i));
    ListBox5.Items.Add(floattostrf(YbAvg,ffGeneral,6,6));
    ListBox4.Items.Add(floattostrf(YtAvg,ffGeneral,6,6));
    ListBox2.Items.Add(floattostrf(Ym1Avg,ffGeneral,6,6));
    ListBox3.Items.Add(floattostrf(Ym3Avg,ffGeneral,6,6));
```

```
repeat
  i:=i+1;
  Cycle:=1;
  DownFlow;

  ListBox2.Items.Add(floattostrf(Ym1Avg,ffGeneral,6,6));


  Cycle:=2;
  UpFlow;

  Cycle:=3;
  DownFlow;
  ListBox3.Items.Add(floattostrf(Ym3Avg,ffGeneral,6,6));
  ListBox5.Items.Add(floattostrf(YbAvg,ffGeneral,6,6));

  Cycle:=4;
  UpFlow;
  ListBox4.Items.Add(floattostrf(YtAvg,ffGeneral,6,6));

  if (YbAvg-Last0)<0.00001 then Stop:=True;
  last0:=YbAvg;
  if Stop and ((YtAvg-Last1)>0.00001) then Stop:=False;
  last1:=YtAvg;

  ListBox1.Items.Add(inttostr(i));

until Stop or (i>100);
Edit1.text:=inttostr(i);
Round:=i;
```

```
end;

procedure TW_Mode3.BitBtn2Click(Sender: TObject);
begin
  Graph.PlotType:=3;
  W_Graph.Show;
end;

procedure TW_Mode3.ListBox4Click(Sender: TObject);
begin
 ListBox5.ItemIndex:=ListBox4.ItemIndex;
 ListBox5.Topindex:=Listbox4.TopIndex;
 ListBox1.ItemIndex:=ListBox4.ItemIndex;
 Listbox1.Topindex:=Listbox4.Topindex;
 ListBox2.ItemIndex:=ListBox4.ItemIndex;
 Listbox2.Topindex:=Listbox4.Topindex;
 ListBox3.ItemIndex:=ListBox4.ItemIndex;
 Listbox3.Topindex:=Listbox4.Topindex;
end;

procedure TW_Mode3.ListBox5Click(Sender: TObject);
begin

   ListBox4.ItemIndex:=ListBox5.ItemIndex;
   ListBox4.TopIndex:=ListBox5.TopIndex;
   ListBox1.ItemIndex:=ListBox5.ItemIndex;
   ListBox1.TopIndex:=ListBox5.TopIndex;
   ListBox2.ItemIndex:=ListBox5.ItemIndex;
   ListBox2.TopIndex:=ListBox5.TopIndex;
```

```
    ListBox3.ItemIndex:=ListBox5.ItemIndex;
    ListBox3.TopIndex:=ListBox5.TopIndex;
end;

procedure TW_Mode3.ListBox1Click(Sender: TObject);
begin

  ListBox4.TopIndex:=ListBox1.Topindex;
  ListBox4.Itemindex:=ListBox1.Itemindex;
  ListBox5.TopIndex:=ListBox1.Topindex;
  ListBox5.Itemindex:=ListBox1.Itemindex;
  ListBox2.TopIndex:=ListBox1.Topindex;
  ListBox2.Itemindex:=ListBox1.Itemindex;
  ListBox3.TopIndex:=ListBox1.Topindex;
  ListBox3.Itemindex:=ListBox1.Itemindex;
end;

procedure TW_Mode3.Edit6Change(Sender: TObject);
var item:integer;
begin

  item:=0;
  if (Edit6.text<>' ') and (Strlen(Pchar(Edit6.text))>0) then
  item:=strtoint(Edit6.Text);
  Edit7.Clear; Edit8.clear;
  if item<Round then
   begin
     Edit7.text:=Listbox4.Items[item];
     Edit8.text:=Listbox5.Items[item];
    Edit11.text:=Listbox2.Items[item];
    Edit12.text:=Listbox3.Items[item];
  end;
end;

procedure TW_Mode3.BitBtn3Click(Sender: TObject);
begin

 Close;
end;

procedure TW_Mode3.BitBtn1Click(Sender: TObject);
begin
 ListBox1.Clear;
 ListBox2.Clear;
 ListBox4.Clear;
 ListBox5.Clear;
 ListBox3.Clear;
 InitAllValue;
 Calculate;
end;

procedure TW_Mode3.Button1Click(Sender: TObject);
var i : integer;
   str : string;
begin
 W_Report.Memo1.Clear;
 W_Report.Memo1.Lines.Add('       N                YT                  YB
YM1                YM3                YM');
```

```
 W_Report.Memo1.Lines.Add(' ');
  i:=0;

  repeat
    str := W_Mode1.AddText(Listbox1.Items[i],20)+'
'+W_Mode1.AddText(Listbox4.Items[i],20)+'      '+
W_Mode1.AddText(ListBox5.Items[i],20);
    str := str +'            '+ W_Mode1.AddText(Listbox2.Items[i],20)+'
'+W_Mode1.AddText(Listbox2.Items[i],20)+'      '+
W_Mode1.AddText(ListBox2.Items[i],20);
    W_Report.Memo1.Lines.Add(str);
    i:=i+1;
  until i>= Listbox1.Items.Count-1;
 W_Report.ShowModal;
end;

procedure TW_Mode3.FormShow(Sender: TObject);
begin

 BitBtn2.Enabled:=False;
end;

procedure TW_Mode3.ListBox2Click(Sender: TObject);
begin
    ListBox4.ItemIndex:=ListBox2.ItemIndex;
    ListBox4.TopIndex:=ListBox2.TopIndex;
    ListBox1.ItemIndex:=ListBox2.ItemIndex;
    ListBox1.TopIndex:=ListBox2.TopIndex;
    ListBox3.ItemIndex:=ListBox2.ItemIndex;
```

```
   ListBox3.TopIndex:=ListBox2.TopIndex;
   ListBox5.ItemIndex:=ListBox2.ItemIndex;
   ListBox5.TopIndex:=ListBox2.TopIndex;
end;

procedure TW_Mode3.ListBox3Click(Sender: TObject);
begin
   ListBox4.ItemIndex:=ListBox3.ItemIndex;
   ListBox4.TopIndex:=ListBox3.TopIndex;
   ListBox1.ItemIndex:=ListBox3.ItemIndex;
   ListBox1.TopIndex:=ListBox3.TopIndex;
   ListBox2.ItemIndex:=ListBox3.ItemIndex;
   ListBox2.TopIndex:=ListBox3.TopIndex;
   ListBox5.ItemIndex:=ListBox3.ItemIndex;
   ListBox5.TopIndex:=ListBox3.TopIndex;
end;
end.
```

Program for TypeIV Form (TypeIV.Pas)

```
unit TypeIV;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, Buttons, ComCtrls,Math;

type
```

```
TW_Mode4 = class(TForm)
 Button1: TButton;
 BitBtn3: TBitBtn;
 BitBtn2: TBitBtn;
 BitBtn1: TBitBtn;
 Bevel1: TBevel;
 Bevel2: TBevel;
 Label4: TLabel;
 Label7: TLabel;
 Label1: TLabel;
 Label8: TLabel;
 Label9: TLabel;
 Label2: TLabel;
 Label11: TLabel;
 Edit4: TEdit;
 Edit5: TEdit;
 ListBox4: TListBox;
 ListBox5: TListBox;
 Edit1: TEdit;
 ListBox1: TListBox;
 Edit6: TEdit;
 Edit7: TEdit;
 Edit8: TEdit;
 ListBox2: TListBox;
 Edit11: TEdit;
 Label13: TLabel;
 ListBox3: TListBox;
 Edit13: TEdit;
 Label15: TLabel;
```

```
Edit15: TEdit;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label3: TLabel;
Label5: TLabel;
Label6: TLabel;
Label17: TLabel;
Label18: TLabel;
Label19: TLabel;
Label10: TLabel;
Label12: TLabel;
Label31: TLabel;
Label32: TLabel;
Label33: TLabel;
Label34: TLabel;
EAp1: TEdit;
EBp1: TEdit;
ECp1: TEdit;
EAp2: TEdit;
EBp2: TEdit;
ECp2: TEdit;
EDp1: TEdit;
EDp2: TEdit;
ECp3: TEdit;
EDp3: TEdit;
Eap3: TEdit;
EBp3: TEdit;
TabSheet2: TTabSheet;
Label14: TLabel;
```

```
Label20: TLabel;
Label21: TLabel;
Label22: TLabel;
Label23: TLabel;
Label24: TLabel;
Label25: TLabel;
Label26: TLabel;
Label27: TLabel;
Label28: TLabel;
Label29: TLabel;
Label30: TLabel;
EAn1: TEdit;
EBn1: TEdit;
ECn1: TEdit;
EAn2: TEdit;
EBn2: TEdit;
ECn2: TEdit;
EDn1: TEdit;
EDn2: TEdit;
EAn3: TEdit;
EBn3: TEdit;
ECn3: TEdit;
EDn3: TEdit;
KCheckBox: TCheckBox;

procedure SetMatrixDown;
procedure SetMatrixUp;
procedure CalMatrix;
procedure ShiftUp;
```

```
    procedure ClearMatrix;
    procedure ShiftDown;
    procedure DownFlow;
    procedure InitAllvalue;
    procedure UpFlow;
    procedure Calculate;
    procedure BitBtn2Click(Sender: TObject);
    procedure SetRightStep;
    procedure ListBox4Click(Sender: TObject);
    procedure ListBox5Click(Sender: TObject);
    procedure ListBox1Click(Sender: TObject);
    procedure Edit6Change(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure ListBox2Click(Sender: TObject);
    procedure ListBox3Click(Sender: TObject);
    function Kfunction(y : double):double;
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  W_Mode4: TW_Mode4;
  MatrixV : Array [0..1,0..100,0..100] of double;
```

```
 Ap,Bp,Cp,Dp,An,Bn,Cn,Dn : Array [1..3] of double;
 MatrixY,MatrixSum,MatrixDev : Array [0..1,0..100] of double;
 V,VB,Y0,YsAvg,YmAvg,OldYmAvg,YtAvg,YbAvg : double;
 Step,Round,ColumnIndex,Cycle,Kindex,StepCount:integer;

implementation

uses Graph, Monitor, Printing, TypeI;

{$R *.DFM}

 procedure TW_Mode4.CalMatrix;
 var i,j: integer;
 begin
  for j:=0 to Step do
   begin
    MatrixSum[ColumnIndex,j]:=0;
    for i:=0 to Step do
      MatrixSum[ColumnIndex,j]:=MatrixSum[ColumnIndex,j]+MatrixY[ColumnIndex,i]
        *MatrixV[ColumnIndex,i,j];
    MatrixSum[ColumnIndex,j]:=MatrixSum[ColumnIndex,j]/MatrixDev[ColumnIndex,j];
   end;
 end;

 procedure TW_Mode4.ClearMatrix;
 var i,j :integer;
 begin
   for i:=0 to Step do
   for j:=0 to Step do MatrixV[ColumnIndex,i,j]:=0;
```

```
end;

 function TW_Mode4.Kfunction(y : double):double;
 var i : integer;
     YY : double;
 begin
     i:=Kindex;
     if KCheckBox.Checked then YY:=1 else YY:=y;
     if ColumnIndex  = 0 then Kfunction := Ap[i]*Power(y,-2)+Bp[i]*Power(y,-1)+Cp[i]
         +Dp[i]/Power(YY,-1) else
         Kfunction := An[i]*Power(y,2)+Bn[i]*Power(y,1)+Cn[i]+Dn[i]/Power(YY,1);
 end;

procedure TW_Mode4.SetMatrixUp;
var i,j: integer;
begin

  ClearMatrix;

  for i:=1 to Step do  MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i];

  if Cycle=2 then
   begin
    if ColumnIndex=1 then   MatrixY[ColumnIndex,0]:=YbAvg
       else MatrixY[ColumnIndex,0]:=YmAvg;
   end else
     begin
       for i:=Step downto 1 do  MatrixY[ColumnIndex,i]:= MatrixY[ColumnIndex,i-1];
       MatrixY[ColumnIndex,0]:=YsAvg;
     end;

       if ColumnIndex =0 then
         begin
           if Cycle= 2 then  Kindex:=1  else   Kindex:=2;
         end else Kindex:=2;

   i:=0;
  for j:= 0 to Step-1 do
   begin
     MatrixV[ColumnIndex,i+1,j]:= VB*Kfunction(MatrixY[ColumnIndex,j+1]);
     MatrixV[ColumnIndex,i,j]:= V;
     i:=i+1;
   end;

   MatrixV[ColumnIndex,Step,Step]:=VB*Kfunction(MatrixY[ColumnIndex,Step])+V;
  for i:=1 to Step-1 do   MatrixDev[ColumnIndex,i]:=
VB*Kfunction(MatrixY[ColumnIndex,i+1])+V;

     if ColumnIndex =1 then  Kindex:=1 else
           Begin
             if Cycle=2 then Kindex:=2 else
                Kindex:=3;
           End;

   MatrixDev[ColumnIndex,Step]:=MatrixV[ColumnIndex,Step,Step];
   MatrixDev[ColumnIndex,0]:=VB*Kfunction(MatrixY[ColumnIndex,1])+V;

 end;
```

```
procedure TW_Mode4.SetMatrixDown;
 var i,j: integer;
 begin

   ClearMatrix;

   for i:=0 to Step-1 do MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i];

   if ColumnIndex=0 then  MatrixY[ColumnIndex,Step]:=YtAvg
    else MatrixY[ColumnIndex,Step]:=YmAvg;

     if (ColumnIndex mod 2)=0 then Kindex:=3 else Kindex:=1;

    i:=0;
   for j:= 1 to Step do
    begin
     MatrixV[ColumnIndex,i,j]:= VB*Kfunction(MatrixY[ColumnIndex,j-1]);
     MatrixV[ColumnIndex,i+1,j]:= V;
     i:=i+1;
    end;
   MatrixV[ColumnIndex,0,0]:=VB*Kfunction(MatrixY[ColumnIndex,0])+V;

   for i:=1 to Step-1 do  MatrixDev[ColumnIndex,i]:=
             V+VB*Kfunction(MatrixY[ColumnIndex,i-1]);

     if (ColumnIndex mod 2)=0 then  Kindex:=1
     else Kindex:=2;
```

```
  MatrixDev[ColumnIndex,0]:=MatrixV[ColumnIndex,0,0];
  MatrixDev[ColumnIndex,Step]:=V+VB*Kfunction(MatrixY[ColumnIndex,Step-1]);

 end;

procedure TW_Mode4.ShiftUp;
var i,j : integer;
begin


  for i:=0 to Step-1 do  MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i+1];

  if ColumnIndex=0 then MatrixY[ColumnIndex,Step]:=YtAvg else
    MatrixY[ColumnIndex,Step]:=YmAvg;

   i:=0;
   for j:= 1 to Step do
    begin
         if ColumnIndex =0 then
        begin
           if j>Step-StepCount+1 then   Kindex:=1 else Kindex:=3;
        end else
        begin
           if j>Step-StepCount+1 then   Kindex:=2 else Kindex:=1;
        end;

    MatrixV[ColumnIndex,i,j]:= VB*Kfunction(MatrixY[ColumnIndex,j-1]);
    MatrixV[ColumnIndex,i+1,j]:= V;
    i:=i+1;
   end;
   if ColumnIndex=0 then Kindex:=3 else Kindex:=1;

  MatrixV[ColumnIndex,0,0]:=VB*Kfunction(MatrixY[ColumnIndex,0])+V;

  for i:=1 to Step do
   begin
     if ColumnIndex =0 then
         begin
           if i>Step-StepCount then   Kindex:=1 else Kindex:=3;
         end else
         begin
           if i>Step-StepCount then   Kindex:=2 else Kindex:=1;
         end;
     MatrixDev[ColumnIndex,i]:= V+VB*Kfunction(MatrixY[ColumnIndex,i-1]);
   end;

  MatrixDev[ColumnIndex,0]:=MatrixV[ColumnIndex,0,0];
end;
procedure TW_Mode4.ShiftDown;
var i,j : integer;
begin

  for i:=step downto 1 do  MatrixY[ColumnIndex,i]:=MatrixSum[ColumnIndex,i-1];

  if Cycle=2 then
  begin
    if ColumnIndex=1 then MatrixY[ColumnIndex,0]:=YbAvg else
      MatrixY[ColumnIndex,0]:=YmAvg;
```

```
end else  MatrixY[ColumnIndex,0]:=YsAvg;


i:=0;
for j:= 0 to Step-1 do
 begin
     if ColumnIndex  =0 then
    begin
     if Cycle=2 then
       begin
         if j<StepCount-1 then  Kindex:=2 else Kindex:=1;
       end else
       begin
         if j<StepCount-1 then  Kindex:=3 else Kindex:=2;
       end;
    end else
    begin
         if j<StepCount-1 then  Kindex:=1 else Kindex:=2;
    end;
   MatrixV[ColumnIndex,i+1,j]:= VB*Kfunction(MatrixY[ColumnIndex,j+1]);
   MatrixV[ColumnIndex,i,j]:= V;
   i:=i+1;
  end;
 if ColumnIndex=0 then
   begin
      if Cycle=2 then Kindex:=1 else Kindex :=2;
   end else Kindex:=2;


 MatrixV[ColumnIndex,Step,Step]:=VB*Kfunction(MatrixY[ColumnIndex,Step])+V;
```

```
  for i:=0 to Step-1 do
    Begin
       if ColumnIndex  =0 then
      begin
       if Cycle=2 then
         begin
           if i<StepCount then  Kindex:=2 else Kindex:=1;
         end else
         begin
           if i<StepCount then  Kindex:=3 else Kindex:=2;
         end;
      end else
      begin
           if i<StepCount then  Kindex:=1 else Kindex:=2;
      end;

      MatrixDev[ColumnIndex,i]:= VB*Kfunction(MatrixY[ColumnIndex,i+1])+V;
    End;
    MatrixDev[ColumnIndex,Step]:=MatrixV[ColumnIndex,Step,Step];


end;
procedure TW_Mode4.DownFlow;
var i: integer;
    averg1,averg0: double;
begin
  StepCount:=1;
  ColumnIndex:=0;
  SetMatrixDown;
  ColumnIndex:=1;
```

```
 SetMatrixDown;

 averg1:=0;
 averg0:=0;
 for i:=0 to Step-1 do
  begin
  Inc(StepCount);
  ColumnIndex:=0;
  CalMatrix;
  ShiftUp;

  averg0:=averg0+MatrixSum[0,0];
   ColumnIndex:=1;
   CalMatrix;
   ShiftUp;
   averg1:=averg1+MatrixSum[1,0];
  end;
     YbAvg:=((YbAvg*Step)+averg1)/(2*Step);
     YsAvg:=((YsAvg*Step)+averg0)/(2*Step);
end;

procedure TW_Mode4.UpFlow;
var i: integer;
    averg0: double;
begin
   StepCount:=1;
 if Cycle = 2 then
  begin
    ColumnIndex:=1;
   SetMatrixUp;
  end;
  averg0:=0;
  for i:=0 to Step-1 do
   begin
     Inc(StepCount);
   if Cycle=2 then
     begin
       ColumnIndex:=1;
       CalMatrix;
       ShiftDown;
       YmAvg:=OldYmAvg;
       OldYmAvg:=(OldYmAvg*Step+MatrixSum[1,Step])/(Step+1);
     end;
   ColumnIndex:=0;

   if i= 0 then SetMatrixUp;
   if Cycle=2 then MatrixY[ColumnIndex,0]:=YmAvg;
   CalMatrix;
   ShiftDown;
   averg0:=averg0+MatrixSum[0,Step];


   end;
   if Cycle=2 then YtAvg:=((YtAvg*Step)+averg0)/(2*Step)
    else  begin
          OldYmAvg:=((YmAvg*Step)+averg0)/(2*Step);
          YmAvg:=OldYmAvg;
         end;
 end;
```

```
procedure TW_Mode4.InitAllValue;
var i,j: integer;
begin
 BitBtn2.Enabled:=True;

 Ap[1]:=strtofloat(EAp1.Text);
 Bp[1]:=strtofloat(EBp1.Text);
 Cp[1]:=strtofloat(ECp1.Text);
 Dp[1]:=strtofloat(EDp1.Text);
 Ap[2]:=strtofloat(EAp2.Text);
 Bp[2]:=strtofloat(EBp2.Text);
 Cp[2]:=Strtofloat(ECp2.Text);
 Dp[2]:=strtofloat(EDp2.Text);
 Ap[3]:=strtofloat(EAp3.Text);
 Bp[3]:=strtofloat(EBp3.Text);
 Cp[3]:=Strtofloat(ECp3.Text);
 Dp[3]:=strtofloat(EDp3.Text);

 An[1]:=strtofloat(EAn1.Text);
 Bn[1]:=strtofloat(EBn1.Text);
 Cn[1]:=strtofloat(ECn1.Text);
 Dn[1]:=strtofloat(EDn1.Text);
 An[2]:=strtofloat(EAn2.Text);
 Bn[2]:=strtofloat(EBn2.Text);
 Cn[2]:=Strtofloat(ECn2.Text);
 Dn[2]:=strtofloat(EDn2.Text);
 An[3]:=strtofloat(EAn3.Text);
 Bn[3]:=strtofloat(EBn3.Text);
 Cn[3]:=Strtofloat(ECn3.Text);
```

```
Dn[3]:=strtofloat(EDn3.Text);

Y0:=strtofloat(Edit15.Text);
OldYmAvg:=Y0;
YsAvg:=Y0;
YtAvg:=Y0;
YmAvg:=Y0;
YbAvg:=Y0;
if strlen(Pchar(Edit5.text))<1 then SetRightStep else
 begin
   if strtoint(Edit5.Text)<2 then Edit5.Text:='2';
   Step:=Strtoint(Edit5.Text);
 end;

OldYmAvg:=Y0;
YsAvg:=Y0;
YtAvg:=Y0;
YmAvg:=Y0;
YbAvg:=Y0;

V:=strtofloat(Edit4.text)/(2*Step);
VB:=V/3;



for j:=0 to 1 do
begin
  for i:=0 to Step do
  begin
    MatrixY[j,i]:=Y0;
```

```
     MatrixSum[j,i]:=Y0;
    end;
  end;

end;

procedure TW_Mode4.SetRightStep;
var i,j: integer;
begin

  Step:=4;
  repeat
    Step:=Step+1;
    V:=strtofloat(Edit4.text)/(2*Step);
    VB:=V/3;
      for j:=0 to 1 do
      begin
          for i:=0 to Step do
              begin
                MatrixY[j,i]:=Y0;
                MatrixSum[j,i]:=Y0;
              end;
      end;
    DownFlow;
  until MatrixSum[0,0]>0.94;
  Edit5.Text:=inttostr(Step);
end;

procedure TW_Mode4.Calculate;
```

```
var i :integer;
  last0, last1 : double;
  stop:boolean;
begin

 ListBox4.Clear; ListBox5.clear; ListBox1.Clear;
 ListBox2.Clear;
 stop:=false; last0:=0; last1:=0;
 i:=0;
   ListBox1.Items.Add(inttostr(i));
   ListBox4.Items.Add(floattostrf(YtAvg,ffGeneral,6,6));
   ListBox3.Items.Add(floattostrf(YsAvg,ffGeneral,6,6));
   ListBox2.Items.Add(floattostrf(YmAvg,ffGeneral,6,6));
 repeat

   i:=i+1;
   Cycle:=1;
   DownFlow;

   ListBox5.Items.Add(floattostrf(YbAvg,ffGeneral,6,6));
   ListBox3.Items.Add(floattostrf(YsAvg,ffGeneral,6,6));

   Cycle:=2;
   UpFlow;
   ListBox4.Items.Add(floattostrf(YtAvg,ffGeneral,6,6));

   Cycle:=3;
   UpFlow;
   ListBox2.Items.Add(floattostrf(YmAvg,ffGeneral,6,6));
```

```
    if Abs(YbAvg-Last0)<0.00001 then Stop:=True;
    last0:=YbAvg;
    if Stop and (Abs(YtAvg-Last1)>0.00001) then Stop:=False;
    last1:=YtAvg;

    ListBox1.Items.Add(inttostr(i));
  until Stop or (i>100);
  Cycle:=1;
  DownFlow;
  ListBox5.Items.Add(floattostrf(YbAvg,ffGeneral,6,6));
  Edit1.text:=inttostr(i);
  Round:=i;
end;

procedure TW_Mode4.BitBtn2Click(Sender: TObject);
begin
   Graph.PlotType:=4;
   W_Graph.Show;
end;

procedure TW_Mode4.ListBox4Click(Sender: TObject);
begin
  ListBox5.ItemIndex:=ListBox4.ItemIndex;
  ListBox5.Topindex:=Listbox4.TopIndex;
  ListBox1.ItemIndex:=ListBox4.ItemIndex;
  Listbox1.Topindex:=Listbox4.Topindex;
  ListBox2.ItemIndex:=ListBox4.ItemIndex;
  ListBox2.Topindex:=Listbox4.TopIndex;
  ListBox3.ItemIndex:=ListBox4.ItemIndex;
  Listbox3.Topindex:=Listbox4.Topindex;
end;

procedure TW_Mode4.ListBox5Click(Sender: TObject);
begin
   ListBox4.ItemIndex:=ListBox5.ItemIndex;
   ListBox4.TopIndex:=ListBox5.TopIndex;
   ListBox1.ItemIndex:=ListBox5.ItemIndex;
   ListBox1.TopIndex:=ListBox5.TopIndex;
   ListBox2.ItemIndex:=ListBox5.ItemIndex;
   ListBox2.TopIndex:=ListBox5.TopIndex;
   ListBox3.ItemIndex:=ListBox5.ItemIndex;
   ListBox3.TopIndex:=ListBox5.TopIndex;
end;

procedure TW_Mode4.ListBox1Click(Sender: TObject);
begin
  ListBox4.TopIndex:=ListBox1.Topindex;
  ListBox4.Itemindex:=ListBox1.Itemindex;
  ListBox5.TopIndex:=ListBox1.Topindex;
  ListBox5.Itemindex:=ListBox1.Itemindex;
  ListBox2.TopIndex:=ListBox1.Topindex;
  ListBox2.Itemindex:=ListBox1.Itemindex;
  ListBox3.TopIndex:=ListBox1.Topindex;
  ListBox3.Itemindex:=ListBox1.Itemindex;
end;

procedure TW_Mode4.Edit6Change(Sender: TObject);
var item:integer;
```

```
begin
 item:=0;
 if (Edit6.text<>' ') and (Strlen(Pchar(Edit6.text))>0) then
 item:=strtoint(Edit6.Text);
 Edit7.Clear; Edit8.clear;
 if item<Round then
  begin
    Edit7.text:=Listbox4.Items[item];
    Edit8.text:=Listbox5.Items[item];
    Edit11.text:=Listbox2.Items[item];
    Edit13.text:=Listbox3.Items[item];
  end;
end;

procedure TW_Mode4.BitBtn3Click(Sender: TObject);
begin
 Close;
end;

procedure TW_Mode4.BitBtn1Click(Sender: TObject);
begin
 ListBox1.Clear;
 ListBox2.Clear;
 ListBox4.Clear;
 ListBox5.Clear;
 ListBox3.Clear;
 BitBtn2.Enabled:=False;
 InitAllValue;
 Calculate;
```

```
end;

procedure TW_Mode4.Button1Click(Sender: TObject);
var i : integer;
   str : string;
begin
 W_Report.Memo1.Clear;
 W_Report.Memo1.Lines.Add('         N                    YT1                    YB1
YM                    YS ');
 W_Report.Memo1.Lines.Add('  ');
  i:=0;

  repeat
    str := W_Mode1.AddText(Listbox1.Items[i],20)+'     '+W_Mode1.AddText(
          Listbox4.Items[i],20)+'      '+ W_Mode1.AddText(ListBox5.Items[i],20);
    str := str +'             '+ W_Mode1.AddText(Listbox3.Items[i],20)+'     '+
          W_Mode1.AddText(Listbox2.Items[i],20);
    W_Report.Memo1.Lines.Add(str);
    i:=i+1;
  until i>= Listbox1.Items.Count-1;
 W_Report.ShowModal;
end;

procedure TW_Mode4.FormShow(Sender: TObject);
begin
 BitBtn2.Enabled:=False;
end;

procedure TW_Mode4.ListBox2Click(Sender: TObject);
```

```
begin
 ListBox4.TopIndex:=ListBox2.Topindex;
 ListBox4.Itemindex:=ListBox2.Itemindex;
 ListBox5.TopIndex:=ListBox2.Topindex;
 ListBox5.Itemindex:=ListBox2.Itemindex;
 ListBox1.TopIndex:=ListBox2.Topindex;
 ListBox1.Itemindex:=ListBox2.Itemindex;
 ListBox3.TopIndex:=ListBox2.Topindex;
 ListBox3.Itemindex:=ListBox2.Itemindex;
end;

procedure TW_Mode4.ListBox3Click(Sender: TObject);
begin
 ListBox4.TopIndex:=ListBox3.Topindex;
 ListBox4.Itemindex:=ListBox3.Itemindex;
 ListBox5.TopIndex:=ListBox3.Topindex;
 ListBox5.Itemindex:=ListBox3.Itemindex;
 ListBox2.TopIndex:=ListBox3.Topindex;
 ListBox2.Itemindex:=ListBox3.Itemindex;
 ListBox1.TopIndex:=ListBox3.Topindex;
 ListBox1.Itemindex:=ListBox3.Itemindex;
end;

procedure TW_Mode4.Button2Click(Sender: TObject);
begin
 W_Monitor.Show;
end;


end.
```

```
// Program for Graph Form  ( Graph.Pas )

unit Graph;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  OleCtrls, graphsv3, chartfx3, StdCtrls, Buttons, ExtCtrls;

type
  TW_Graph = class(TForm)
    BitBtn1: TBitBtn;
    Panel2: TPanel;
    RadioGroup1: TRadioGroup;
    RadioGroup2: TRadioGroup;
    RadioGroup3: TRadioGroup;
    Image1: TImage;
    Image2: TImage;
    RadioGroup4: TRadioGroup;
    procedure BitBtn1Click(Sender: TObject);
    procedure Plot;
    procedure FormShow(Sender: TObject);
    procedure ScrollBar1Change(Sender: TObject);
    procedure PlotI;
    procedure PlotII;
    procedure PlotIII;
    procedure PlotIV;
    procedure ScrollBar2Change(Sender: TObject);
    procedure RadioGroup1Click(Sender: TObject);
    procedure RadioGroup3Click(Sender: TObject);
    procedure RadioGroup4Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  W_Graph: TW_Graph;
  Org,round,Size,Xsize:integer;
  value,Y0 :real;
  PlotType,Fluid : integer;

implementation

uses Main, TypeI, TypeII, TypeIII, TypeIV;

{$R *.DFM}

procedure TW_Graph.PlotI;
var xx,yy,i : integer;
Begin
 Image2.Picture.LoadFromFile('Graph1'+inttostr(Fluid)+'.bmp');
 Image2.Top := 545-Image2.Height;
 Image2.Left:=125;
 Y0 := TypeI.Y0;
 round:= W_Mode1.ListBox4.items.Count ;
 image1.Canvas.font.Color:=clRed;
 image1.Canvas.TextOut(140,Trunc(Org/1.5)+60,'Yt');
 image1.Canvas.font.Color:=clGreen;
 image1.Canvas.TextOut(140,Trunc(Org/1.5)-60,'Yb');

 if round > 1000 then round:=1000;
 image1.Canvas.Moveto(55,Trunc(Org/1.5)-
Trunc((strtofloat(W_Mode1.listbox4.items[0])-Y0)*50*20/size/Y0));
 image1.Canvas.Pen.Color:=clRed;
 for i:=1 to round-1 do
  begin
   xx:= Trunc(55+i*80/Xsize);
   yy:= Trunc(Org/1.5)-Trunc((strtofloat(W_Mode1.listbox4.items[I])-Y0)*50*20/size/Y0);
   if (xx<750) and (yy<600) and (yy>0) then image1.Canvas.lineto(xx,yy);
  end;
 image1.Canvas.Moveto(55,Trunc(Org/1.5)-
Trunc((strtofloat(W_Mode1.listbox5.items[0])-Y0)*50*20/size*10/Y0));
 image1.Canvas.Pen.Color:=clGreen;

 for i:=1 to round-1 do
  begin
   xx:= Trunc(55+i*80/Xsize);
   yy:= Trunc(Org/1.5)-Trunc((strtofloat(W_Mode1.listbox5.items[I])-Y0)*50*20/size/Y0);
   if (xx<750) and (yy<600) and (yy>0) then  image1.Canvas.lineto(xx,yy);
  end;
End;

procedure TW_Graph.PlotII;
```

```
var i : integer;
Begin
 Image2.Picture.LoadFromFile('Graph2'+inttostr(Fluid)+'.bmp');
 Image2.Top := 541-Image2.Height;
 Image2.Left:=125;
 Y0 := TypeII.Y0;
 round:= W_Mode2.ListBox4.items.Count-1 ;
 if round > 1000 then round:=1000;

 if Fluid=1 then
 begin
 image1.Canvas.font.Color:=clRed;
 image1.Canvas.TextOut(140,Trunc(Org/1.5)+60,'Yt1');
 image1.Canvas.font.Color:=clGreen;
 image1.Canvas.TextOut(140,Trunc(Org/1.5)-60,'Yb1');
 image1.Canvas.Moveto(55,Trunc(Org/1.5)-
Trunc((strtofloat(W_Mode2.listbox4.items[0])-
         Y0)*50*20/size/Y0));
 image1.Canvas.Pen.Color:=clRed;
 for i:=1 to round-1 do
  begin
    image1.Canvas.lineto(Trunc(55+i*80/XSize),Trunc(Org/1.5)-Trunc((
            strtofloat(W_Mode2.listbox4.items[I])-Y0)*50*20/size/Y0));
  end;

 image1.Canvas.Moveto(55,Trunc(Org/1.5)-Trunc((strtofloat(
         W_Mode2.listbox5.items[0])-Y0)*50*20/size/Y0));
 image1.Canvas.Pen.Color:=clGreen;
 for i:=1 to round-1 do
 begin
  image1.Canvas.lineto(Trunc(55+i*80/XSize),Trunc(Org/1.5)-Trunc((
        strtofloat(W_Mode2.listbox5.items[I])-Y0)*50*20/size/Y0));
 end;
end else
begin
image1.Canvas.font.Color:=clBlue;
image1.Canvas.TextOut(140,Trunc(Org/1.5)-80,'Yt3');
image1.Canvas.font.Color:=clTeal;
image1.Canvas.TextOut(140,Trunc(Org/1.5)+80,'Yb3');
image1.Canvas.Moveto(55,Trunc(Org/1.5)-Trunc((strtofloat(
         W_Mode2.listbox3.items[0])-Y0)*50*20/size/Y0));
image1.Canvas.Pen.Color:=clBlue;
for i:=1 to round-1 do
 begin
  image1.Canvas.lineto(Trunc(55+i*80/XSize),Trunc(Org/1.5)-Trunc((
          strtofloat(W_Mode2.listbox3.items[I])-Y0)*50*20/size/Y0));
 end;
image1.Canvas.Moveto(55,Trunc(Org/1.5)-Trunc((strtofloat(
          W_Mode2.listbox6.items[0])-Y0)*50*20/size/Y0));
image1.Canvas.Pen.Color:=clTeal;
for i:=1 to round-1 do
 begin
  image1.Canvas.lineto(Trunc(55+i*80/XSize),Trunc(Org/1.5)-Trunc((
          strtofloat(W_Mode2.listbox6.items[I])-Y0)*50*20/size/Y0));
 end;
 end;
End;
```

```
procedure TW_Graph.PlotIII;
var xx,yy,i : integer;
Begin
 Y0 := TypeIII.Y0;
 round:= W_Mode3.ListBox4.items.Count ;
 image1.Canvas.font.Color:=clRed;
 image1.Canvas.TextOut(140,Trunc(Org/1.5)-60,'Yt');
 image1.Canvas.font.Color:=clGreen;
 image1.Canvas.TextOut(140,Trunc(Org/1.5)+60,'Yb');

 if round > 1000 then round:=1000;
 image1.Canvas.Moveto(55,Trunc(Org/1.5)-Trunc((strtofloat(
             W_Mode3.listbox4.items[0])-Y0)*50*20/size/Y0));
 image1.Canvas.Pen.Color:=clRed;
 for i:=1 to round-1 do
  begin
   xx:= Trunc(55+i*80/Xsize);
   yy:= Trunc(Org/1.5)-Trunc((strtofloat(W_Mode3.listbox4.items[I])-Y0)*50*20/size/Y0);
   if (xx<750) and (yy<600) and (yy>0) then image1.Canvas.lineto(xx,yy);
  end;
 image1.Canvas.Moveto(55,Trunc(Org/1.5)-Trunc((strtofloat(
            W_Mode3.listbox5.items[0])-Y0)*50*20/size/Y0));
 image1.Canvas.Pen.Color:=clGreen;

 for i:=1 to round-1 do
  begin
   xx:= Trunc(55+i*80/Xsize);
   yy:= Trunc(Org/1.5)-Trunc((strtofloat(W_Mode3.listbox5.items[I])-Y0)*50*20/size/Y0);
   if (xx<750) and (yy<600) and (yy>0) then  image1.Canvas.lineto(xx,yy);
   end;
 End;

 procedure TW_Graph.PlotIV;
var i : integer;
 Begin
   Image2.Picture.LoadFromFile('Graph4'+inttostr(Fluid)+'.bmp');
   Image2.Top := 545-Image2.Height;
  Image2.Left:=125;
  Y0 := TypeIV.Y0;
  round:= W_Mode4.ListBox1.items.Count-1 ;
  if round > 1000 then round:=1000;

  image1.Canvas.font.Color:=clGreen;
  image1.Canvas.TextOut(160,Trunc(Org/1.5)-20,'Yb');

  if Fluid=1 then
  begin
  image1.Canvas.font.Color:=clRed;
  image1.Canvas.TextOut(160,Trunc(Org/1.5)+60,'Yt');
  image1.Canvas.Pen.Color:=clRed;
  image1.Canvas.Moveto(55,Trunc(Org/1.5)-Trunc((strtofloat(
              W_Mode4.listbox4.items[0])-Y0)*50*20/size/Y0));
 for i:=1 to round-1 do
  begin
   image1.Canvas.lineto(Trunc(55+i*80/Xsize),Trunc(Org/1.5)-
Trunc((strtofloat(W_Mode4.listbox4.items[I])-Y0)*50*20/size/Y0));
  end;
 end else
 begin
 image1.Canvas.font.Color:=clBlue;
 image1.Canvas.TextOut(160,Trunc(Org/1.5)-60,'Ys');
 image1.Canvas.Pen.Color:=clBlue;
 image1.Canvas.Moveto(55,Trunc(Org/1.5)-Trunc((strtofloat(
            W_Mode4.listbox3.items[0])-Y0)*50*20/size/Y0));


 for i:=1 to round-1 do
  begin
    image1.Canvas.lineto(Trunc(55+i*80/XSize),Trunc(Org/1.5)-Trunc((
                strtofloat(W_Mode4.listbox3.items[I])-Y0)*50*20/size/Y0));
  end;
 end;

 image1.Canvas.Pen.Color:=clGreen;
 image1.Canvas.Moveto(55,Trunc(Org/1.5)-
Trunc((strtofloat(W_Mode4.listbox5.items[0])-Y0)*50*20/size/Y0));
 for i:=1 to round-1 do
  begin
    image1.Canvas.lineto(Trunc(55+i*80/XSize),Trunc(Org/1.5)-Trunc((
                strtofloat(W_Mode4.listbox5.items[I])-Y0)*50*20/size/Y0));
  end;
End;

procedure TW_Graph.Plot;
var i : integer;
begin
 size := strtoint(RadioGroup2.Items[RadioGroup2.ItemIndex]);
```

```
xsize := strtoint(RadioGroup1.Items[RadioGroup1.ItemIndex]);
image1.Canvas.brush.Style:=bsSolid;
image1.Canvas.FillRect(rect(0,0,image1.width,image1.height));
image2.Canvas.brush.Style:=bsSolid;
image2.Canvas.FillRect(rect(0,0,image2.width,image2.height));
Org:=screen.Height-(RadioGroup3.ItemIndex+1)*80;
image1.Canvas.Pen.Color:=clGreen;
image1.Canvas.moveto(42,trunc(Org/1.5));
image1.Canvas.lineto(image1.width-20,trunc(Org/1.5));
image1.Canvas.moveto(55,Org+200);
image1.Canvas.lineto(55,0);
value:=1;
image1.Canvas.Font.Name:='Courior New';
image1.Canvas.font.Color:=clNavy;
image1.Canvas.font.Style:=[fsbold];
image1.Canvas.Font.Size := 10;
image1.Canvas.pen.Width:=1;
for i:=0 to 30 do
 begin
  if i mod 2=0 then image1.Canvas.TextOut(12,Trunc(Org/1.5)-i*20-
                        8,floattostrf(value,ffGeneral,3,2));
  value:=Value+Size/50;
  image1.Canvas.moveto(50,Trunc(Org/1.5)-i*20);
  image1.Canvas.lineto(60,Trunc(Org/1.5)-i*20);
 end;
 value:=1;
 for i:=0 to 14 do
 begin
  if i mod 2 =0 then image1.Canvas.TextOut(12,Trunc(Org/1.5)+i*20-
                8,floattostrf(value,ffGeneral,3,2));
 image1.Canvas.moveto(50,Trunc(Org/1.5)+i*20);
 image1.Canvas.lineto(60,Trunc(Org/1.5)+i*20);
 if Value<=0 then break;
 value:=Value-Size/50;
 if Value<0.00001 then   Value:=0;

end;
value:=0;

for i:=0 to 14 do
begin
 image1.Canvas.TextOut(45+Trunc(i*80),Trunc(Org/1.5)-16,inttostr(Trunc(value)));
 value:=Value+XSize;
 image1.Canvas.moveto(55+Trunc(i*80),Trunc(Org/1.5)-2);
 image1.Canvas.lineto(55+Trunc(i*80),Trunc(Org/1.5)+2);
end;
 image1.Canvas.pen.Width:=2;
case PlotType of
 1 : PlotI;
 2 : PlotII;
 3 : PlotIII;
 4 : PlotIV;
 end;
end;

procedure TW_Graph.BitBtn1Click(Sender: TObject);
begin
 Close;
```

```
end;

procedure TW_Graph.FormShow(Sender: TObject);
begin
 RadioGroup2.ItemIndex:=2;
 RadioGroup1.ItemIndex:=0;
 plot;
end;

procedure TW_Graph.ScrollBar1Change(Sender: TObject);
begin
 plot;
end;

procedure TW_Graph.ScrollBar2Change(Sender: TObject);
begin
  Plot;
end;

procedure TW_Graph.RadioGroup1Click(Sender: TObject);
begin
 Plot;
end;

procedure TW_Graph.RadioGroup3Click(Sender: TObject);
begin
 Plot;
end;
```

```
procedure TW_Graph.RadioGroup4Click(Sender: TObject);
begin
  Fluid := RadioGroup4.ItemIndex+1;
  Plot;
end;

procedure TW_Graph.FormCreate(Sender: TObject);
begin
 Fluid:=1;
end;

end.


// Program for Printing Form (Printing.Pas)

unit Printing;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, quickrpt, Qrctrls, StdCtrls, Buttons, ComCtrls;

type
  TW_Report = class(TForm)
    BitBtn1: TBitBtn;
    SaveDialog1: TSaveDialog;
    BitBtn3: TBitBtn;
    Memo1: TRichEdit;
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  W_Report: TW_Report;

implementation

{$R *.DFM}
procedure TW_Report.BitBtn1Click(Sender: TObject);
begin
 if SaveDialog1.Execute then
   Memo1.Lines.SaveToFile(Savedialog1.filename);
end;

procedure TW_Report.BitBtn3Click(Sender: TObject);
begin
  Close;
end;

end.
```

ประวัติผู้เขียน

นายศุภมิตร จตุพรฆ้องชัย เกิดวันที่ 31 กรกฎาคม พ.ศ.2516 ที่อำเภอเมือง จังหวัดปราจีนบุรี สำเร็จการศึกษาปริญญาตรีวิทยาศาสตร์บัณฑิต สาขาเคมีเทคนิค ภาควิชาเคมีวิศวกรรมคณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2537 และเข้าศึกษาต่อในหลักสูตร วิศวกรรมศาสตร์มหาบัณฑิต ที่ จุฬาลงกรณ์มหาวิทยาลัย เมื่อ พ.ศ.2539