Experiment No. : 01 Date : 14/12/2020

Program for Vander Waal's Equation.

Aim:- Write a program to determine the molar volume of gas using Vander-waal's Equation of State.

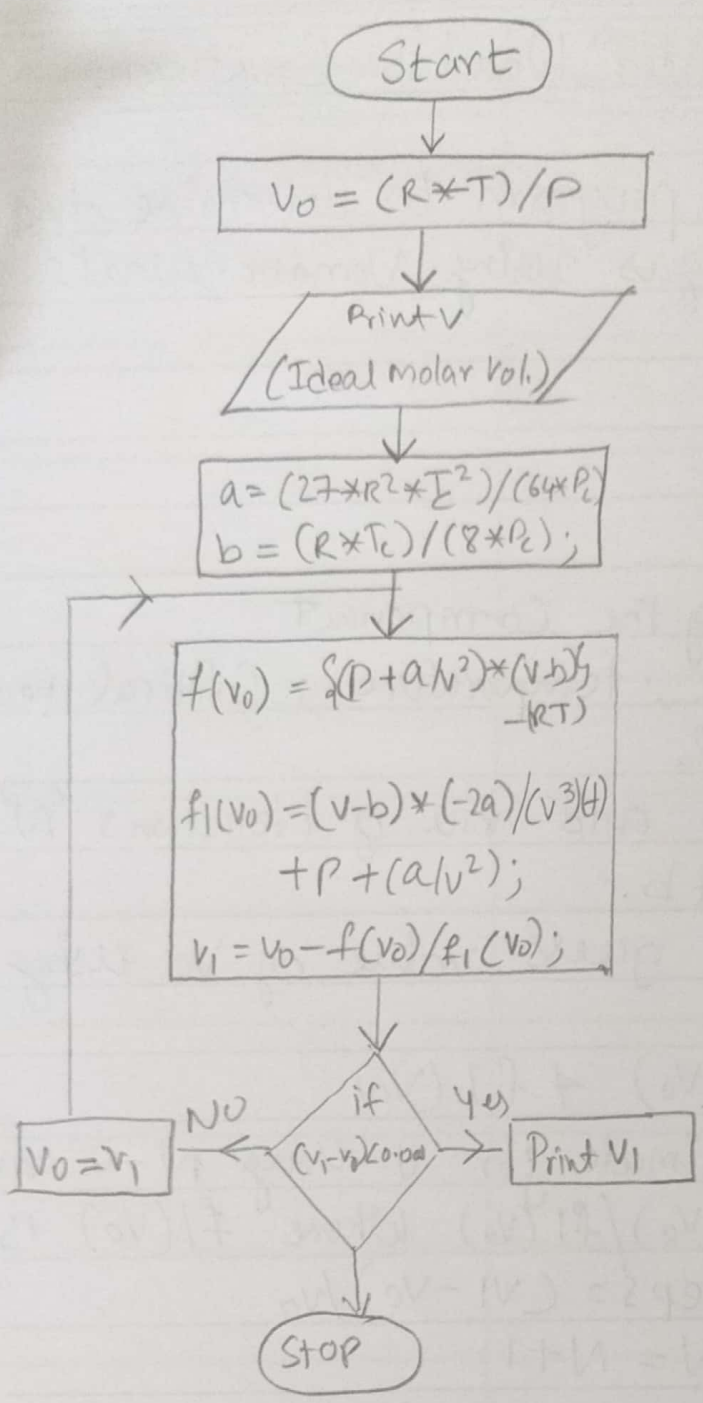
Algorithm:-

1. Start
2. Input name of the component
3. Input pressure, temperature ; Critical temp. & Critical pressure.
4. Input accuracy and no. of iterations 'N'.
5. Calculate a & b.
6. Calculate the guess value of V_0 using perfect gas lane.
7. Calculate $f(V_0)$ & $f1(V_0)$
8. Find nere estimate for V using N-R methods. $V1 = V_0 - f(V_0)/f1(V_0)$ Where $f1(V_0)$ is not zero.
9. Find error, $eps = (V1 - V_0)/V_0$
10. Increment $N = N+1$
11. Check if $N <$ no. of iterations.
12. If eps < accuracy, $V_0 \leftarrow V_1$, goto Step 7. else print V & N.

Equations:-

$$\left(P + \frac{a}{V^2}\right)(V-b) = RT$$

Flowchart :-



Output :

Ideal molar volume is 0.734370
Value of a & b are 3.5960695 & 0.042743
Molar Volume is 0.679417

Experiment No. :..... Date :.....

$a = (27R^2T_c^2)/64P_c$

$b = RT_c/8P_c$

$f(v) = (P + a/v^2)(v-b) = RT$

Source code :—

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main()
{
clrscr();
float fVO, flVO, a, b, V1, V0, P, T, Pc = 70, Tc = 304, R = 0.0082;
int i;
Printf("Enter the value of temperature & Pressure");
Scanf("%f %f", &P, &T);
V0 = (R*T)/P;
printf("Ideal molar Volume is %.f \n", V0);
a = (27*R*R*Tc*Tc)/(64*Pc);
b = (R*Tc)/8*Pc;
printf(" Value of a & b are %f & %f \n"), a, b);

fV0 = ((P+a/V0*V0)*(V0-b)) - R*T;
flV0 = (V0-b)*(-2*a)/(V0*V0*V0)+P+(a/V0*V0);
V1 = V0 - fV0/flV0;
if (fabs(V1-V0) < 0.001)
{
Printf("Molar Volume is %f \n", V1);
}
```

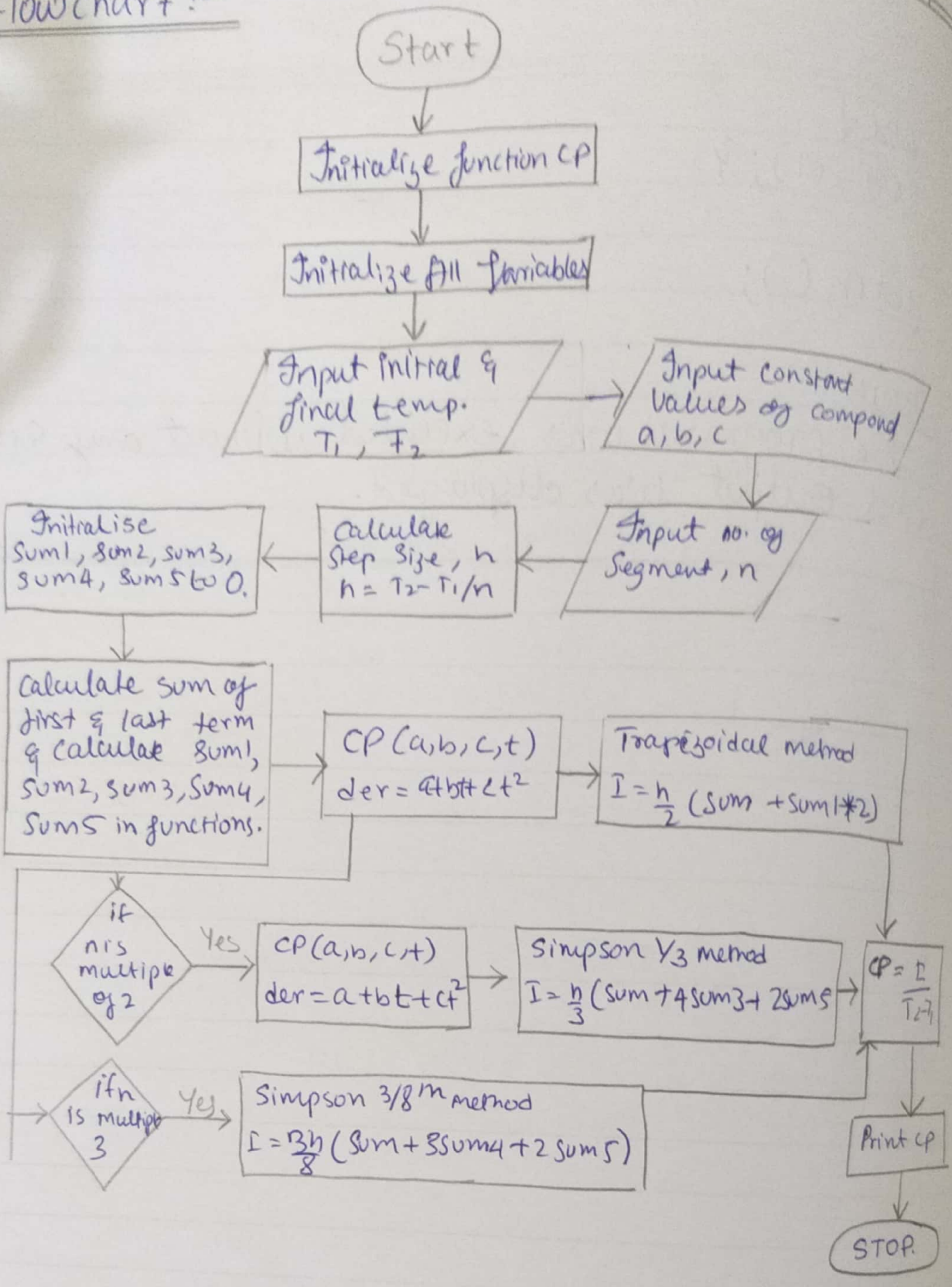
Experiment No. :..... Date :.....

```
else {
(v0 = v1); }
}
getch();
```

Result:-

The program was executed without any errors and output was displayed.

Flowchart :-



Experiment No. : 02 Date : 14/12/2020

Solution of definite Integral Equation.

Aim:- Write a program to find average specific heat of methane using Simpsons 1/3rd, 3/8th rule $T_1 = 300k$, $T_2 = 400k$.

Algorithm:-

1. Start
2. Input Compound name.
3. Input the initial and final temperature.
4. Input constants of Compound a, b, c.
5. Assign no. of Segments, n
6. Calculate Step Size $h = (T_2 - T_1)/n$
7. Calculate the sum of first and last value of the function $a + bT + cT^2$
8. Calculate sum of other values of the function
9. Calculate Integral value
   Simpson's 1/3rd method for n = multiple of 2
   $I = h/3 (Sum + 4 Sum3 + 2 Sum3)$
   Simpson's 3/8th method for n = multiple of 3.
   $I = 3h/8 (Sum + 3Sum4 + 2 Sum5)$
10. Check if number of Segment 'n' is multiple of 2 or multiple of 3 or both.
11. Print IP Value corresponding to 'n' type /* function */ /* function */
    1. Initialise variable that are required.
    2. Calculate der = $a + bT + cT^2$.
    3. return der.

Output :-

Input initial & final temperature in kelvin - 300, 400

Input constant values of Compound a, b, c - 0.003431, 0.00000 54 69, 0.00000000 3661

Enter no. of segment 6
h = 16.666666

Trapezoidal method cp is 0.023024 J/mol.k
Simpson's 1/3 rd cp is 0.023024 J/mol.k
Simpson's 3/8 rd cp is 0.023024 J/mol.k

Experiment No. :..... Date :.....

Source Code:-

```
#include<stdio.h>
float CP(float, float, float, float);
void main()
{
float t, t1, t2, a, b, c, h, sum1, sum2, sum3, sum4, sum5,
 sum, i1, i2, i3, cp1, cp2, cp3;
int i, n;

Printf("Input Initial and final temperature in kelvin");
Scanf("%f %f", &t1, &t2);
Printf("Enter the value of constants a, b, c");
Scanf("%f %f %f", &a, &b, &c);
printf("Enter the number of segments");
Scanf("%d", &n);
h = (t2-t1)/n;
Printf("\n h %f", h);
Sum1 = 0;
Sum2 = 0;
Sum3 = 0;
Sum4 = 0;
Sum5 = 0;
Sum = CP(a, b, c, t1) + CP(a, b, c, t2);
for (i=1; i<n; i++)
Sum1 = Sum1 + (CP(a, b, c, (t1 + (i*h))));
for (i=2; i<n; i = i+2)
Sum2 = Sum2 + (CP(a, b, c, (t1 + (i*h))));
for (i=3; i<n; i=i+2)
Sum3 = Sum3 + (CP(a, b, c, (t1 + (i*h))));
```

Experiment No. :..... Date :.....

```
for(i=1; i<n; i++)
{
if(i%3==0)
sum5 = sum5+(cp(a,b,c,(t1+(i*h))));
else
sum4 = sum4+(cp(a,b,c,(t1+(i*h))));
}
// trapezoidal method
i1 =(h*(sum+(2*sum)))/2;
cp1=i1/(t2-t1);
printf("\n trapezoidal method cp is %f j/molk",cp1);

// Simpsons 1/3rd method
if(n%2==0)
{
i2 =(h*(sum+(4*sum2)+(2*sum3)))/3;
cp2 = i2/(t2-t1);
printf("\n Simpsons 1/3 rule method cp is %f
        j/molk", cp2);
}
// Simpsons 3/8th rule
if(n%3==0)
{
i3= ((3*h)*(sum+(3*sum4)+(2*sum5)))/8;
cp3=i3/(t2-t1);
printf("\n Simpsons 3/8th rule cp is %f j/molk",cp3);
}}
float cp(float a, float b, float c, float t)
```

Experiment No. :..... Date :.....

```
{
float der;
der = a + (b*t) + (c*t*t);
return der;
}
```

Result:-

The program was Executed Successfully.

Experiment No. : 03 Date : 16/12/2020

Runge kutta Method

Aim:- The temperature gradient in a furnace wall heated from one side at a particular instant of time is given by equation
$dT/dx = -40x^3 + 72x^2 + 24x - 70$, $x(m)$. Write a C program to calculate the temperature profile (T as a func. of x) for the range $0 \le x \le 0.5$ in steps of 0.05m, given $T = 1000K$ at $x = 0$ using Runge kutta method.

Theory :-

Consider $\frac{dy}{dx} = f(x, y) = y(x_0) = y_0$

$$x_1 = x_0 + h$$

According to Eulur's method $y_1 = y_0 + h f(x_0, y_0)$

$k_1 = h * f(x_0, y_0)$

$k_2 = hf(x_0 + h/2, y_0 + k_1/2)$

$k_3 = hf(x_0 + h/2, y_0 + k_2/2)$

$k_4 = hf(x_0 + h, y_0 + k_3)$

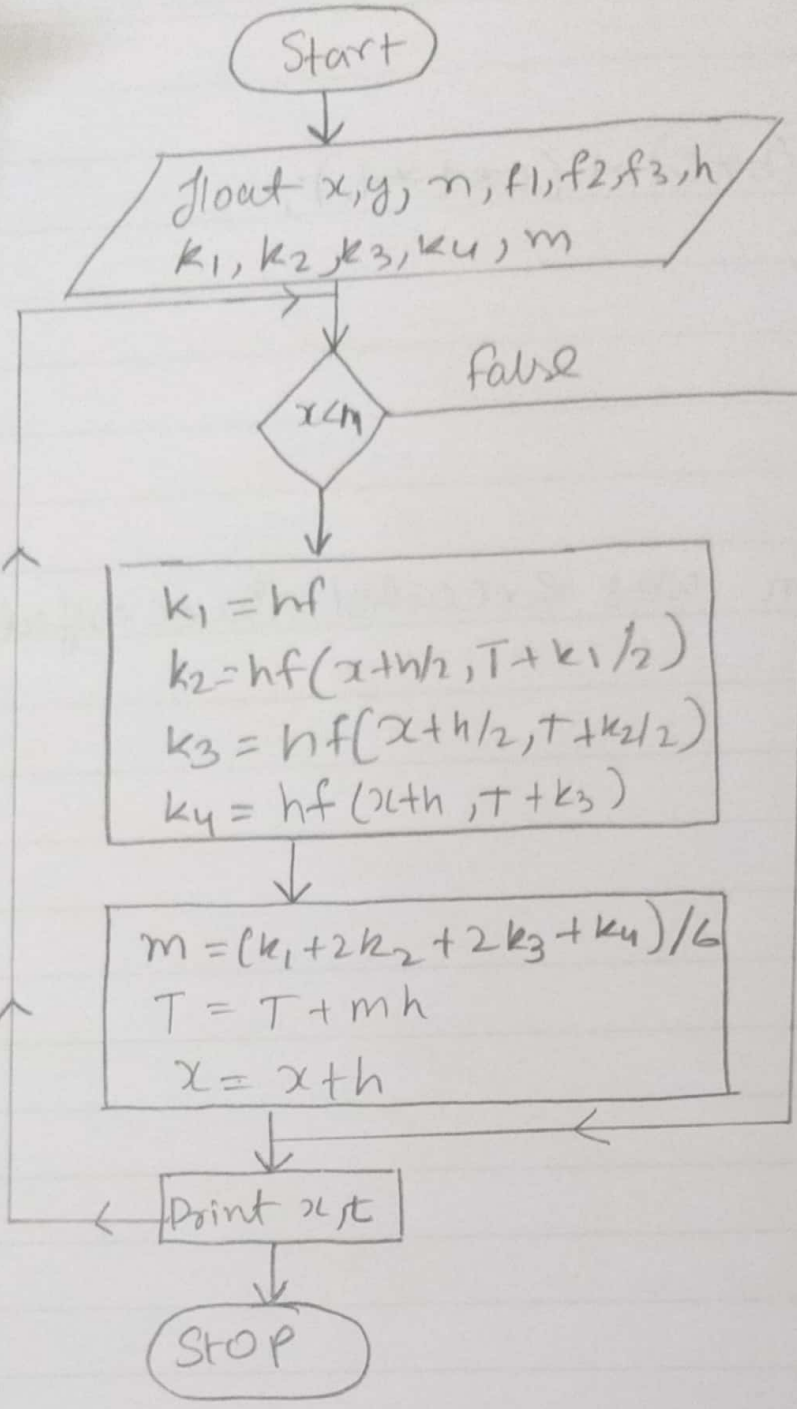
$$y_{n+1} = y_n + \frac{1}{6}\left[k_1 + 2k_2 + 2k_3 + k_4\right]$$

This method is of arranging the slopes at different points in a given interval.

Algorithm :-

1. Start
2. Assign initial value of (x, T).

Flowchart :-



Output :-

$x = 0.05$	$T = 999.325195$
$x = 0.10$	$T = 999.654297$
$x = 0.15$	$T = 999.488159$
$x = 0.20$	$T = 999.323767$
$x = 0.25$	$T = 999.173767$
$x = 0.30$	$T = 999.027283$
$x = 0.35$	$T = 998.889038$
$x = 0.40$	$T = 998.640808$
$x = 0.45$	$T = 998.640808$
$x = 0.50$	$T = 998.532471$

Experiment No. : Date :

3. Input values of x at which T is required
4. Decide the step size h.
5. Compute the no. of steps required $n=(x-x_0)/h$
6. Find k_1, k_2, k_3, k_4 for $n=0$.
7. Compute $T_{n+1} = \frac{1}{6}(k_1+2k_2+2k_3+k_4)$
8. Increase the values of x_n by 1 step
9. Repeat steps till $x_n \leq x$.
10. Display values of x & T.

Source Code :-

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
#define f(x,T) -40*pow(x,3)+72*pow(x,2)+24*x
                -2.

Void main()
{
float x, T, x0, h, f, f1, f2, f3, k1, k2, k3, k4, n;
printf("Take the initial value of x");
Scanf("%f", &T);
printf("Enter last values of x");
Scanf("%f %f", &x, &h);
while(x<n)
{
 f = f(x,T);
 k1 = h*f;
 f1 = f(x+h/2, T+(k1/2));
 k2 = h*f1;
 f2 = f((x+(h/2)), T+(k2/2));
```

Experiment No. :..... Date :.....

```
k3 = h*f2;
f3 = f(x+h, f+k3);
k4 = h* f3;
m=(k1 + 2k2 +2k3 + k4);
T = T + (m*h);
x = x+h;
printf("for x = %f , T = %f", x, T);
}
getch()
}
```

Result :-

The program was Executed sucessfully.

Experiment No. 04 Date 16/12/2020

Solution of Simultainous Linear Equations :-

Aim :-

A polymer blend is to be formed from three feed streams A, B, C having the following compositions in wt percent. Write a program to determine A, B & C for 500kg of blend production.

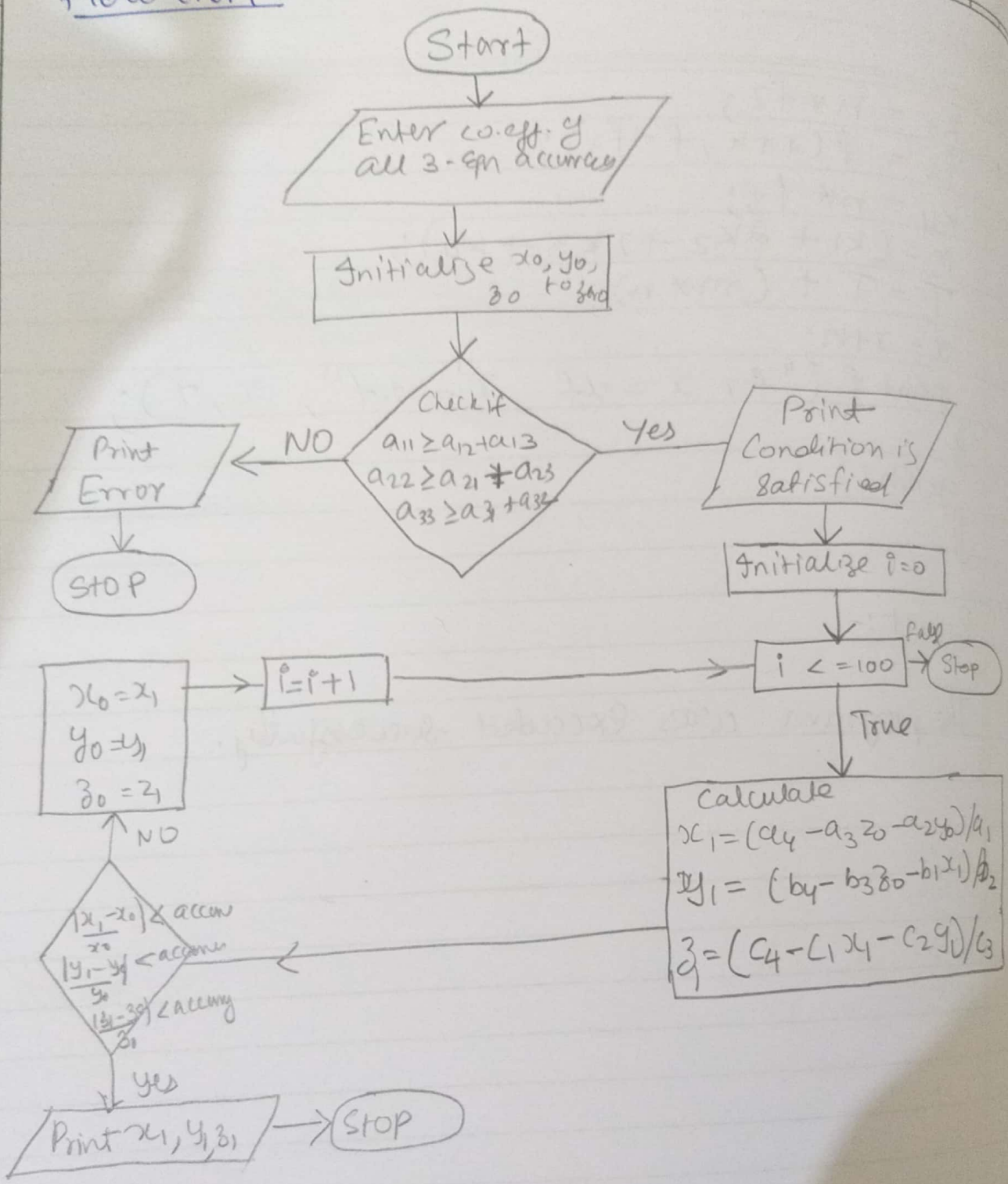
Component	A	B	C	Blend
1	55	35	5	30
2	35	50	40	30
3	10	15	55	40

Algorithm :-

1. Start
2. Declare the variables and initialize with required values
3. Ask the user to Enter the accuracy
4. Input accuracy.
5. Initialise flag variable f = 0 & initial values x_0, y_0, z_0 to 0.
6. Calculate values of x, y, z using Gauss Seidel formula :

$$x = (a_4 - (a_2 * y_0) - (a_3 * z_0))/a_1$$
$$y = (b_4 - (b_1 * x_0) - (b_3 * z_0))/a_2$$
$$z = (c_4 - (c_1 * x_0) - (c_2 * y_0))/a_3$$

Flow chart :-



Output :-

Enter coefficient of 3 equations.

0.55 0.33 0.05 150
015 0.55 0.40 150
030 0.15 0.55 200

Condition is satisfied.

Enter accuracy.
0.01

$x = 227.2727$ kg
$y = 45.4544$ kg
$z = 227.2727$ kg

Iteration = 20

Experiment No. :..... Date :.....

7. Check if the input accuracy is obtained for x, y, z.

if yes,
print values of x, y, z.
update flag variable ; i=1
if no,
update initial values as:
$x_0 = x$
$y_0 = y$
$z_0 = z$.

8. Check the value of flag variable ; i:
If i=0,
Go to step 6.
if i=1;
Exit.

9. Stop.

Source Code:-

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
main()
{float acc,x1,y1,z1,x0,y0,z0,a1,a2,a3,a4,b1,b2,
      b3,b4,c1,c2,c3,c4,errx,erry,errz;
int i;
printf("Enter Coefficient of 3 Equations");
```

Experiment No. : ..... Date : .....

```
Scanf("%f %f %f %f %f %f %f %f %f %f %f %f", &a1, &a2, &a3,
      &a0, &b1, &b2, &b3, &b4, &c1, &c2, &c3, &c4);

x0 = 0;
y0 = 0;
z0 = 0;
if((a1>=(a2+a3)) && (b2>=(b1+b3)) && (c3>=(c2+c1))
printf("condition is satisfied");
else
{ printf("Error");
  Exit(0)
}
printf("Enter accuracy");
Scanf("%f", &acc);
// Gauss-Seidel method.
for(i=0 i<=100; i++)
{ x1 = (a4 - (a3*z0) - (a2*y0))/a1;
  y1 = (b4 - (b3*z0) - (b1*x1))/b2;
  z1 = (c4 - (c1*x1) - (c2*y1))/c3;
  errx = fabs(x1-x0)/x0;
  Erry = fabs(y1-y0)/y0;
  Errz = fabs(z1-z0)/z0;
  if((errx && erry && Errz) < acc)
{
  printf("\n x = %f", x1);
  printf("\n y = %f", y1);
  printf("\n z = %f", z1);
  Exit(0);
}
```

Experiment No. :..... Date :.....

```
}
Else
{
  x0 = x1;
  y0 = y1;
  z0 = z1
}
}
}
```

Result:-

Program was successfully executed.

Experiment No. : 005

Date : 18/12/2020

Curve fitting : Method of Least Squre

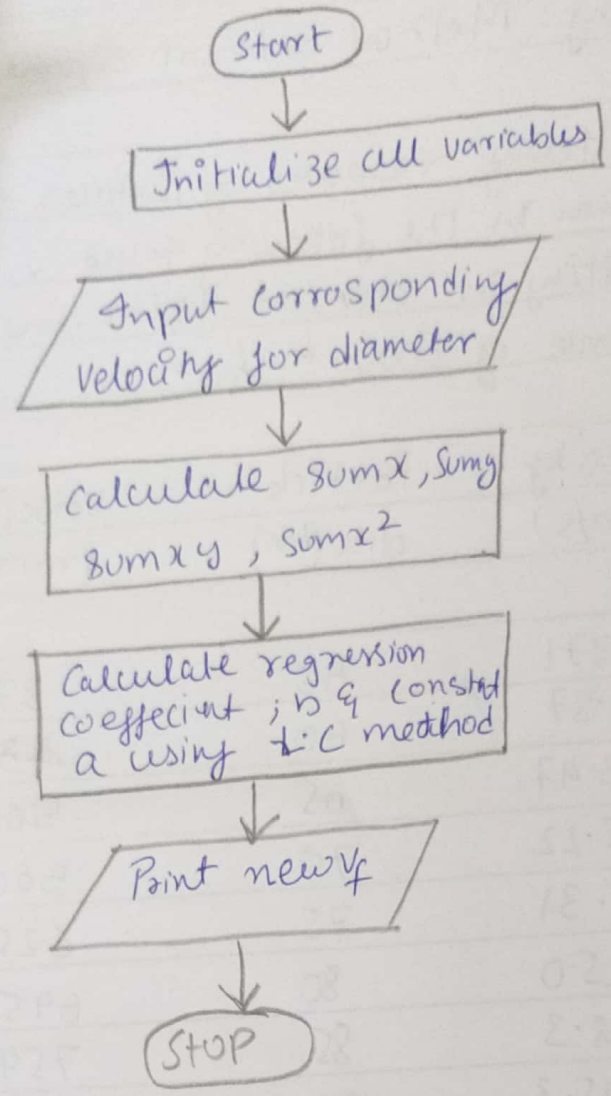
Aim : The free Setteling velocites of Particles of diffenut sizes are given in the following table . The data refers to Settling in air at 1atm and 20°C at a density difference of 5000 kg/m³

Particle dia (μm)	Velocity (mm/s)	Particle dia (μm)	Velocity (mm/s)
5	3.871	55	378
10	14.87	60	~~439~~
15	33.47	65	500
20	58.22	70	560
25	59.31	75	625
30	125.0	80	695
35	168.3	85	759
40	215.3	90	823
45	266.4	95	890
50	320	100	957

Algorithm :-

1. Start
2. Declare variables and initializing with requirred values
3. Ask the user to Enter Corrosponding velocities in mm/s for the displayed diameter.
4. Convert given relation, $V_f = a(D_p)^b$ into $y = mx + c$ from taking log, $\log V_f = b \log D_p + \log a$

Flow chart :-



Experiment No. :..... Date :.....

5. Vary loop variable, i for given diameter range & increment
Calculate $\log V_f$ & $\log D_p$ values Calculate ~~$\log V_f$~~
Sum of x terms, xy, y, x^2 terms using forms.
$\Sigma x = \Sigma \log D_p$.
$\Sigma y = \Sigma \log V_f$
$\Sigma xy = \Sigma (\log V_f \log D_p)$
$\Sigma x^2 = \Sigma \log D_p^2$

6. Calculate regression coeffecient, b & Constant a,
$b = (M * \Sigma xy) - (\Sigma x \Sigma y) / (M * \Sigma x^2 - (\Sigma x)^2)$

7. Print entered old velocity & new velocity found using $V_f = a(D_p)^b$

8. Stop.

Source Code :-

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
Void main()
{
    Clset();
    inti, d=5;
    float u, unew, a, b, x, y, s, t, Sum1=0,
    Sum2=0, Sum3=0 Sum4=0;
    for (i=0; i<20; i++)
```

Experiment No. :..... Date :.....

```
{
    printf ("\n Diameter (mm): %d", d);
    printf ("\t Velocity (mm/s): ");
    scanf ("%f", &u);
    x = log(d); printf ("\t log dia = %f", x);
    y = log(u); printf ("\t log Velocity: %f", y);
    s = x*y; printf ("\n\t xy = %f", s);
    t = x*x; printf ("\t x^2 %f \n", t);
    sum1 = sum1 + x;
    sum2 = sum2 + y;
    sum3 = sum3 + s;
    sum4 = sum4 + t;
    d = d+5;

}
b = (20*sum3 - (sum1*sum2))/(20*sum4 - (pow
    (sum1, 2)));

a = exp((sum2 - b*sum1)/20);
printf ("\n Value of a = %f \t b= %f", a, b);
d = 5;

for (i=0; i<20; i++)
{
  printf ("diameter (mm) : %d ", d)
  Unew = a* pow (d, b);
  printf ("New Velocity (mm/s) : %f", Unew);
  d = d+5;
}
getch ();
}
```

Experiment No. :..... Date :.....

Result :- Program was Executed Successfully using C++ program.

Experiment No. : 06 Date : 18/12/2020.

Dew Temperature Calculations

Aim :- Write a Matlab program to find temperature, t & mole fraction x_i for acetone (1) / acetonitrile (2) / nitromethane (3) System the Antoine's Equations are.

$\ln P_1^s = 14.5463 - 2940.46/(t+237.22)$

$\ln P_2^s = 14.2724 - 2945.467/(t+224)$

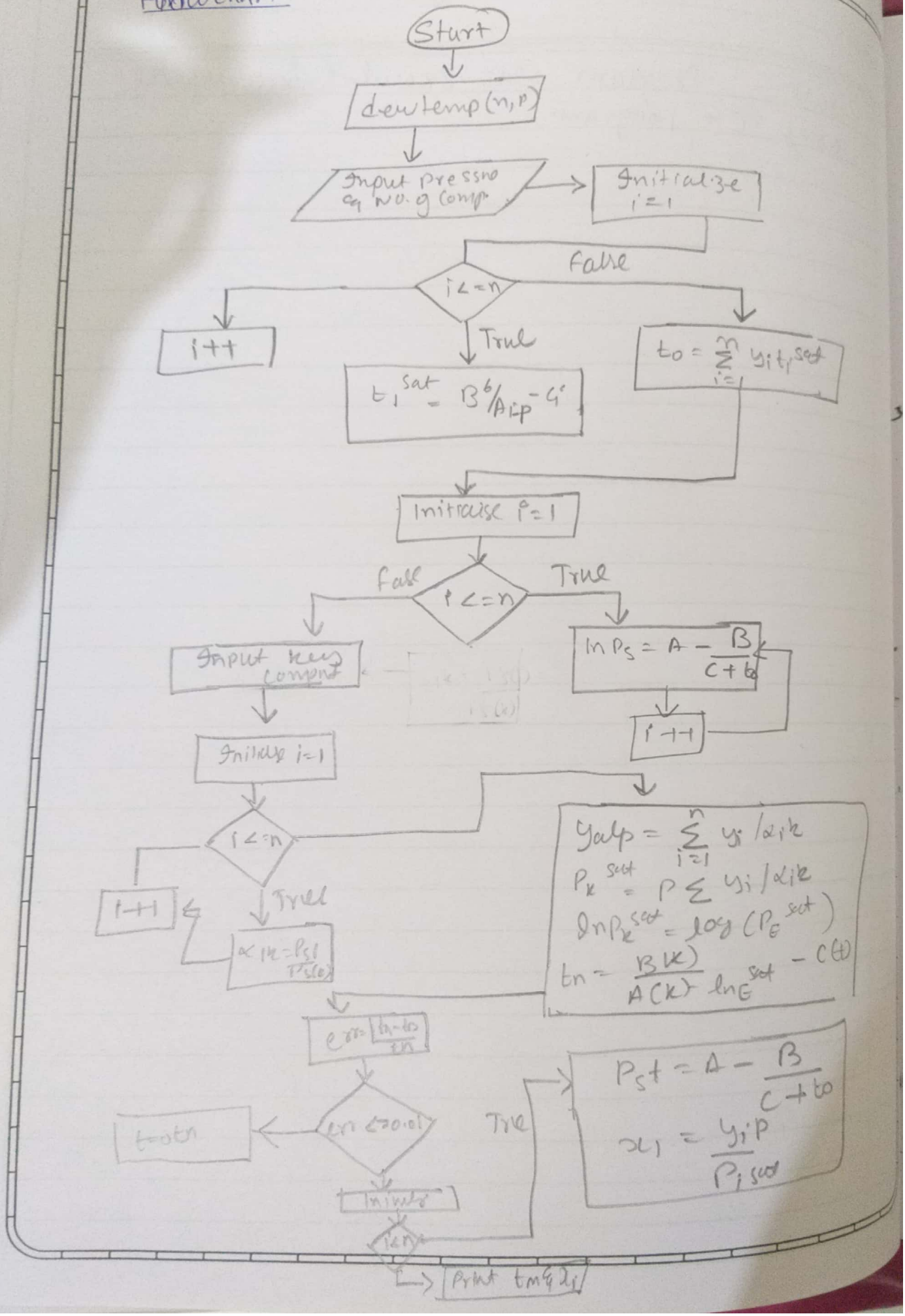
$\ln P_3^s = 14.2043 - 2972.64/(t+209)$; t(°C), P(kPa) Given, P = 90kPa, $y_1 = 0.6$; $y_2 = 02$ $y_3 = 0.2$

Liquid mixtures do not have temperature. During vapourization the temperatures of liquid increases. During condensation dew temperature is at which the first drop of condensate forms.

Algorithm :-

1. Start
2. Input no. of components
3. Input pressure, y_i Antoine's constants.
4. Calculate t^{sat} for i = n using P in Antoine Eqn.
   $\ln P^{sat} = \{ B_i / [A_i - \ln P_i^{sat}] \} - C_i$
5. Initial given guess for temperature, $t_0 = \sum_{i=1}^{n} y_i t_i^{sat}$
6. Calculate P_i^{sat} using Antoine Equation at t_0
7. Taking k' as the key compond Calculate $\alpha_{ik} = P_i^{sat} / P_k^{sat}$ for i = 1 to n.

Flowchart :-



output

demo temp (3,90)

y = 0.6 0.2 0.2

A = 14.5463 14.2724 14.2043

B = 2900.46 2445.467 2972.64

C = 237.22 224 209

t_1^{sat} = −39.5748 −22.7248 −42223

to = −29.0144

P_s = 162.3103 79.7476 40.0342

k = 2

alp = 2.0353 1 0.502

galp = 0.8932

P_{sk} = 80.3869

ln P_{sk} = 4.3869

tm = 73.0571

err = 0.0033

P_{s1} = 163.5019 80.3875 40.3939

x = 0.3303 0.2239 0.4456

temp = 73.9570°C

Experiment No. :..... Date :.....

8. Calculate $P_d = P_k^{Sat} / \Sigma y_i / dck$

9. Calculate 'tnew' using Antoine Equation.

10. Check if $|(tnew - to)/tnew| \geq eps$.

11. Calculate P_i^{Sat} at 'tnew' using Antoine Equation

12. Calculate $x_i = y_i P / P_i^{sat}$.

13. Print 'tnew' & x_i for $i = 1$ to n

14. Stop.

Source Code :-

```
function den = dewtemp(n,P)
P = log(P)
y = input('Enter y')
A = input('Enter Ai')
B = input('Enter Bi')
C = input('Enter Ci')
for i = 1:n
    tis(i) = [B(i)/A(i) - P] - C(i)
end
to = Sum(y * tis)
for i = 1:100
  for P = 1:n
    Ps(i) = exp(A(i) - ((B(i)/(C(i) + to))))
  end
```

Experiment No. :..... Date :.....

```
k = input('Enter key component')
for i = 1:n
alp(i) = Ps(i)/Ps(k)
end
yalp = sum(y./alp)
Psk = P* yalp
lnPsk = log(Psk)
tm = (B(k)/(A(k) - lnPsk)) - C(k)
err = abs((tn-to)/tn)
if (err <= 0.01)
    for i = 1:n
    Ps1(i) = exp(A(i) -(B(i)/(C(i) + tm)))
    x(i) = (y(i)*P)/Ps1(i)
end
fprintf('temperature = %.f c', tn)
Pause
else
to = tn
end
end.
```

Result :- Program was successfully Executed. in matlab.

Experiment No. : 07 Date : 18/12/2020

Bubble Temperature Calculation :-

* Aim :- Write a matlab program to find T' y_i for acetone / acetonitrile / nitromethane, system, hp Antoine's Equations for these are

$\ln P_1^s = 14.5463 - 2940.46/(t + 23722)$;
$\ln P_2^s = 14.2724 - 2945.46/(t + 224)$;
$\ln P_3^s = 14.2043 - 2972.64/(t + 209)$
$t(°C)$, $P(kPa)$. Assuming Raoults is valid
Given $P = 80\ kPa$, $x_1 = 0.3$, $x_2 = 0.45$, $x_3 = 0.25$

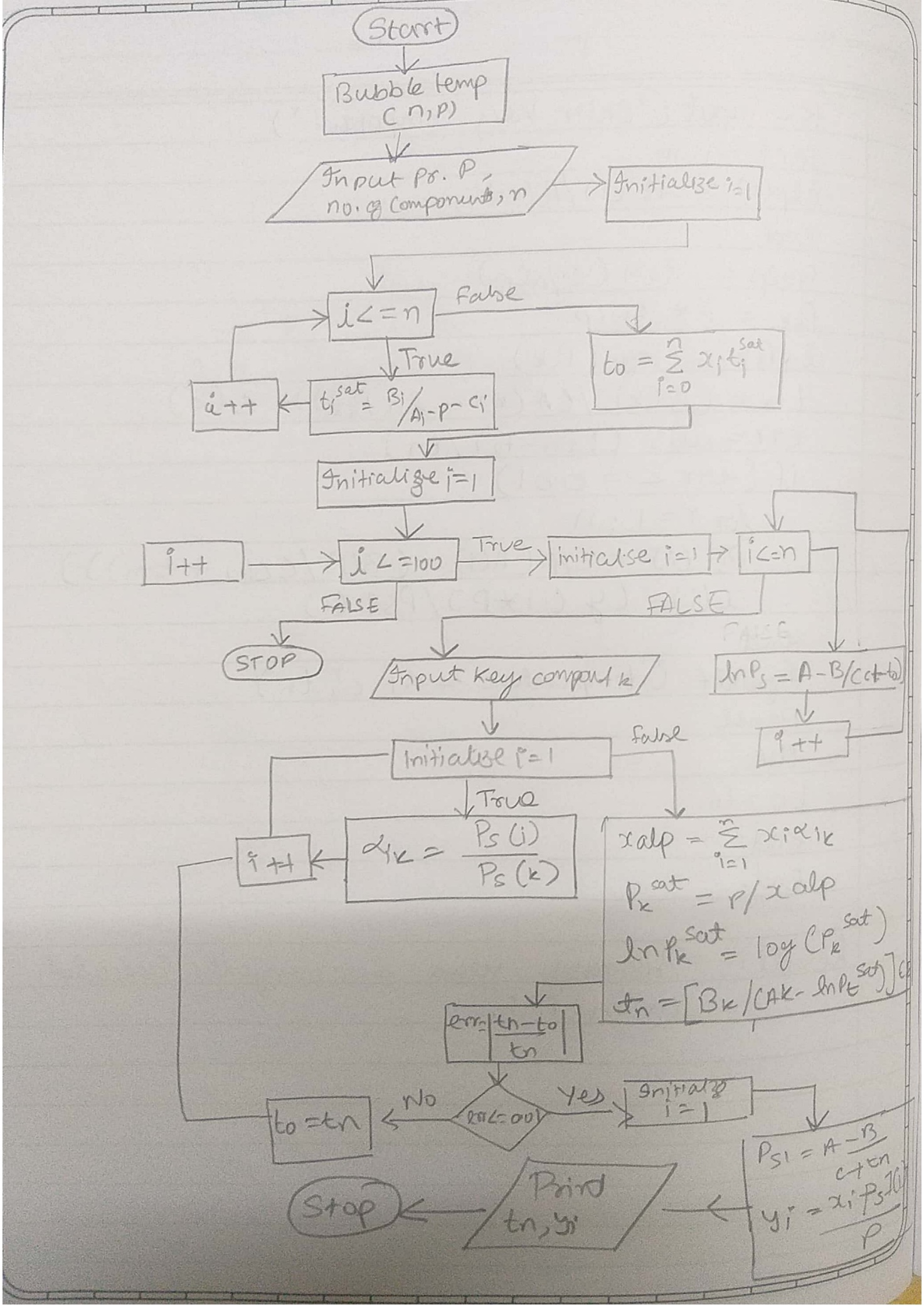
* Theory :-
Bubble - Point ~~Pressure~~ - $T_{x,y}$ given - P_{y} Unknown.
This is a trial & Error procedure must be followed ; where T is assumed, the vapour pressures Calculated, and then se if the correct total pressure is obtained.

$$\sum y_i = 1 = \sum \frac{P_i^{sat} x_i}{P}$$

* Algorithm :-

1. Start
2. Input pressure P and no. of Component, n,
3. Input x_i, Antonie's Constant A_i, B_i, C_i for $i = 1$ ton
4. Calculate t_i^{sat} at P using Antoine's Equation
5. Initial guess for temperature, $t_0 = \sum_{i=1}^{n} x_i t_i^{sat}$
6. Using t_0 in Antoine's Equation, Calculate P_i^{sat} for i 1 ton
7. Take a key component 'k'
8. Calculate $\alpha_{ik} = P_i^{sat} / P_k^{sat}$ for $i = 1:n$

Flow chart :-



output:-

bubble temp (3,80)

x = [0.3 0.45 0.25]

A = [14.5463 14.2724 14.2043]

B = [2940.46 2945.467 2972.64]

C = [237.22 224 209]

tis = [52.073 73.811 93.642]

t0 = 72.2480

Ps = [139.0618 67.3452 33.1282]

k = 2

alp = [2.0649 1.00 0.4919]

x.alp = 1.1925

Psk = 67.0886

lnPsk = 4.2060

tn = 68.6042

err = 0.0016

PsL = [138.5776 67.0886 32.9881]

y = [0.5797 0.3774 0.1031]

temperature = 68.604227

Experiment No. :..... Date :.....

9. Calculate $P_k^{sat} = P / \sum_{i=1}^{n} x_i \alpha_{ik}$

10. Using P_k^{sat} in Antoine's Equation, Calculate new temperature 'tnew'.

11. Check $\left| \frac{tnew - to}{tnew} \right| \leq eps$, if no then go to Step 6 & to tnew else go to next Step.

12. Calculate P_i^{Sat} at tnew using Antone's Equation for i=1 to n.

13. Calculate $y_i^{\circ} = x_i P_i^{Sat} / P$

14. Point temperature and y_i

15. Stop.

* Source Code :-

```
function btc = bubble temp (n,P)
    P = log(P)
    x = input ('Enter xi Values')
    A = input ('Enter Ai Values')
    B = input ('Enter Bi Values')
    C = input ('Enter Ci Values')
for i = 1:n
  tis(i) = B(i)/( A(i)-P)) - C(i)
end
```

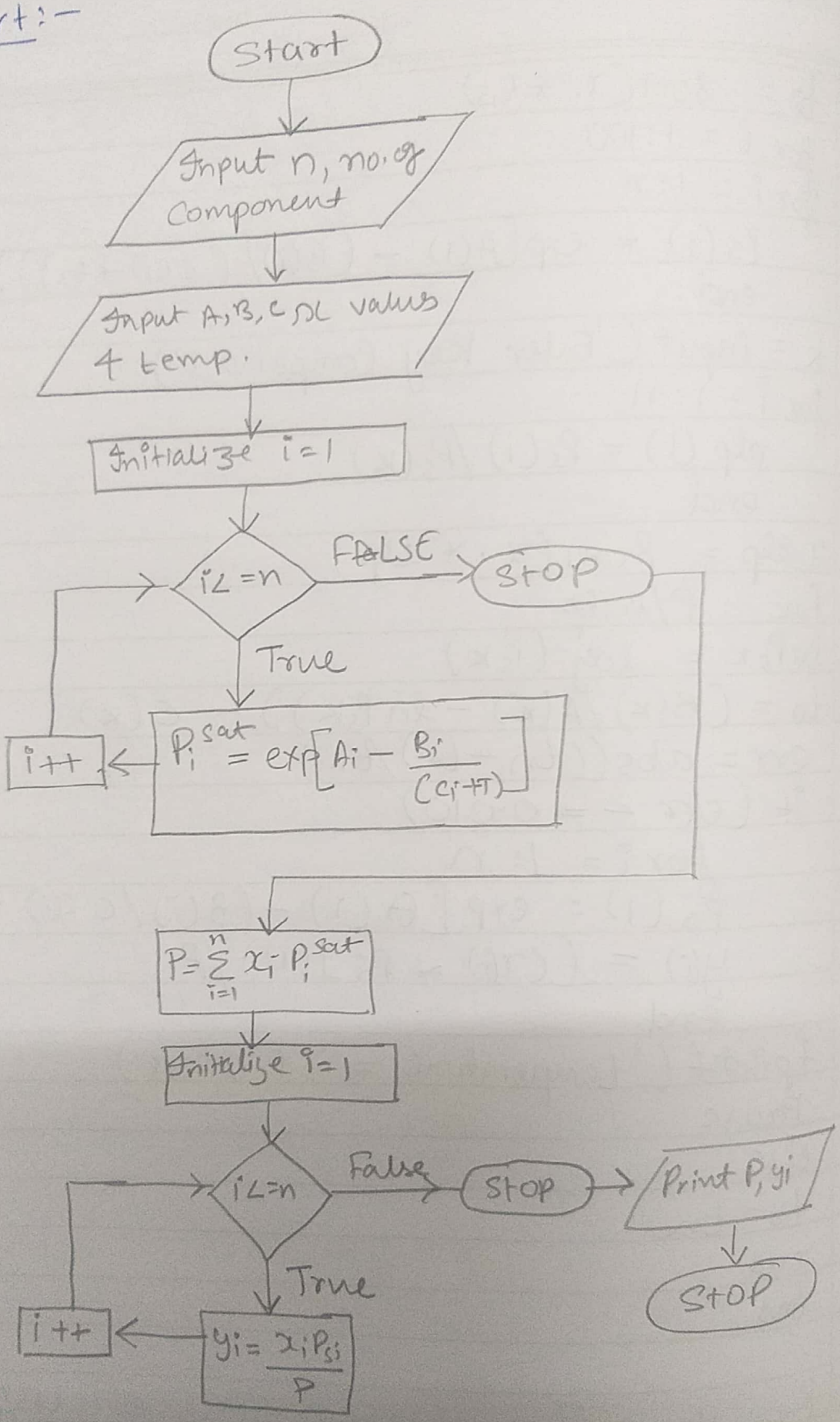
Experiment No. :.....

Date :.....

```
to = sum(x. * tis)
for i = 1:100
for i = 1:n
    Ps(i) = exp[A(i) - (B(i)/(c(i)+to)))
    end
k = input('Enter key Component')
for i = 1:n
    alp(i) = Ps(i)/Ps(k)
    end
xalp = sum(x.* alp)
Psk = P/xalp
lnPsk = log(Psk)
tn = (B(k)/A(k) - lnPsk)) - C(k)
err = abs((tn-to)/tn)
if (err <= 0.010)
    for i = 1:n
    Ps(i) = exp[A(i) - (B(i)/c c(i) +tn))]
    y(i) = ((x(i) * Ps1(i))/P
    end
fprintf('temperature = %f', tn)
Pause
else
to = tn
end
end
```

Result:- The program was successfully Executed on Mathab.

Flowchart :-



Experiment No. : 08 Date : 28/12/2020

Bubble & Dew Pressure Calculations (MATLAB)

* Aim :- Write a program to find dew Pressure and x_i for acetone (1) acetonitride (2) nitromethane (3) System The Antoine's Equations are -

$$\ln P_1^s = 14.5463 - \frac{2940.46}{t + 237.22}$$

$$\ln P_2^s = 14.2724 - \frac{2945.467}{t + 224}$$

$$\ln P_3^s = 14.2043 - \frac{2972.64}{t + 209}$$

Assume Roult's law is valid. Calculate P & x_i ;
$t = 80^\circ C$; $y_1 = 0.5$; $y_2 = 0.3$; $y_3 = 0.2$

* Algorithm :-

1) Start
2) Accept the values of y_i, T and n.
3) Accept the Antoine's constant A_i, B_i, C_i for i = 1 to n.
4) Calculate $P_i^{sat} = \exp[A_i - B_i/[C_i + T]]$ for i = 1 to n.
5) Calculate $P = 1/\sum_{i=1}^{n} [y_i/P_i^{sat}]$ for i = 1 to n.
6) Calculate $x_i = y_i P / P_i^{sat}$ for i = 1 to n.
7) Print P and x_i for i = 1 to n.
8) Stop.

Output :-

```
A = [14.3463, 14.3724, 14.2043];
B = [2940.26, 2945.46, 2997.64];
C = [ 25722 224 209];

Y = [05 03 02];
n = 3;
T = 30;
for i = 1:
```

```
B =
   1.0e+03
   2940.5  2945.5  2997.6
C =
   237.2200  224.0000  209.0000
y =
   0.5000   0.3000   0.2000
T =
   80
n = 3
Ps = 195.7142
Ps =
   195.7142  97.8430
Ps =
   195.7142  97.8430  6.8097
M =
   128.5894
P =
   0.0078
```

```
x =
   1.2586e+04
x = 3.7745e+03
x =
   175.1311
P = 0.007777kpa = 17513
y = 175.113112
```

Experiment No. :..... Date :.....

Source Code :-

```
A=[14.5463 14.2724 14.2043]
B=[2940.46 2945.467 2972.64]
C=[237.22 224 209]
y=[0.5, 0.3, 0.2]
T=80
n=3
for i=1:n
Ps(i) = exp(A(i)-B(i)/(C(i)+T)))
end
m=sum(y*Ps)
P=1/m
for i=1:n
x(1)=(y(i)*Ps(i))/P
end
fprintf('P=%f kpa', P)
for i=1:n
fprintf('y=%f', x(1))
end
```

Result :-

The program was succesfully executed using MATLAB.

Experiment No. : 09 Date : 23/12/2020

Flash Vapourization :-

* Aim :- A system acetone(1) - acetonitrile (2) - nitromethane (3) at 80°C & 110kPa has overall composition. $Z_1 = 0.45$; $Z_2 = 0.35$; $Z_3 = 02$ write MATLAB code to determine L, V, x_i, y_i if misture is flash vapourized.

Component	A	B	C
1	14.5463	2940.46	237.22
2	14.2724	2945.47	224
3	14.2043	2972.64	209

Flash vapourization is a single stage operation where in a liquid mixture is partially vapourized the vapour allowed to come equlibrium with residual liquid & resulting vapour & liquid phase are seperated & removed.

Algorithm :-

1. Start.
2. Input no. of Componets ; n
3. Input Z_i ; A_i, B_i, C_i, P.
4. Calculate P_i^{sat} (Antonie's Eqn)
5. Calculate $P_b = \sum Z_i P_i^{sat}$ & $P_d = \frac{1}{\sum Z_i / P_i^{sat}}$
6. Check $P_d < P < P_b$ if yes goto next step else flash is not Possible.