Flash Program Pseudo-Code John Hillert

1. Need structure for fluids
2. Need arrays for z, y, x, phi_l, phi_v, a, b, Ki
3. Set psi equal to .5
4. Pick T
5. Pick P
6. Designate all molar z compositions of 5 fluids
7. Pick equation of state
    a. if RK or RKS then u = 1, w = 0
    b. if PR then u = 2, w= -1
    c. calculate ai and bi arrays

----------this is where great loop starts----------

8. Set y values and x values each equal to z values
9. Solve for a_mx and b_mx
10. Solve for a_my and b_my
11. Solve for Ay, Ax, By, Bx
12. Use A, B, u, w to solve for Z in cubic
13. Solve for phi_l[i], phi_v[i] using bi and ai, bm, am, Z, u, w
14. Solve for K, K = phi_l/phi_v
15. Solve for new xi, and yi by replacing variables in array with
    a. xi=zi/(Ki+phi(1-K)
    b. yi=zi/(xi*Ki)
16. sum up all (xi-yi)
17. if sum(xi-yi)<0.001 then you're done
18. if not then psi = psi – sum(zi(Ki-1)/(Ki+psi(1-Ki))/sum(zi(Ki-1)^2/(Ki+psi(1-Ki))^2)

----------this is where loop ends----------

19. output final psi
20. output xi and yi arrays

Required functions:
1.
Algorithm

Stucture "props"
1. take T_c, P_c, Omega

Functions
1. a_sum
    a. input: T, P, z mole fractions
    b. output: a_sum, b_sum
2. b_sum
    a. Input: T,P, z mole fractions
    b. Output: a_sum, b_sum
3. cubic
    a. input: a,b
    b. outputs: Zl or Zv
4. p(a,b,c)

input: a, b, c coefficients to cubic
output: p

5. q(a,b,c)

input: a,b,c coeffiecients to cubic
output: q

6.

Includes:
1. math.h
2. iostream.h
3. string.h

Main
Double T, P
int choice
int array z[4]
    y[4]
    x[4]
props Ethane, propane, 1-propanol, pentane, n-heptane
double psi =.5, psi_vap, psi_liq
double = Ax, Ay, amx, bmx, aiy, biy

// input data
Cout >> enter T
Cin << T
Cout >> enter P
Cin << enter P
Cout >> enter z of Ethane
Cin >> z[0]
Cout >> enter z of Propane
Cin >> z[1]
Cout >> enter z of 1-propanol
Cin >> z[3]
Cout >> enter z of pentane
Cin >> z[4]
Cout >> enter z of n-heptane
Cin >> z[5]

Loop 0-4
Set x[I]=z[I]
Set y[I]=z[I]

Cout >> enter EOS >> endl >> 1. RK >> endl >> 2. RKS >> endl >> 3. Peng-Robinson >> endl;
Cin << choice

while(statement about convergence)
//A and B
Ax = a_sum()*P/(R^2*T^2)
Ay = a_sum()*P/(R^2*T^2)

Bx = a_sum()
By = b-sum()
//put into cubic

//calculating z factors
Zl = Cubic(Ax,Bx)

Zv = Cubic(Ay,By)

//calculating psi_vap and psi_liq
Psi_vap = Psi(Ay, By, Zv, w,

Function Bodies
Cubic
p(a,b,c)
q(a,b,c)