```
//============================== EQUATION OF STATE
#include "global.h"
#include <stdio.h>

double conduct(double p);  // declare function

void eos(const int W, double E[][W+1], double T[][W+2], double p[]
[W+1], double Fx[][W+1], double Fy[][W+1]){
        // Nodes
        for(int i = 1; i <= M; i++){
                for(int j = 1; j <= M; j++){
                        if(E[i][j] < 0){
                                //if (i==i){printf("im solid\n");}
                                T[i][j] = Tm + E[i][j]/(rho*Cs);
                                p[i][j] = 0;
                        }else if(E[i][j] >= 0 && E[i][j] <= rho*L){
                                //printf("im mushy\n");
                                T[i][j] = Tm;
                                p[i][j] = E[i][j]/(rho*L);
                        }else if(E[i][j] > rho*L){
                                //printf("im liquid\n");
                                T[i][j] = Tm + (E[i][j] - rho*L)/
(rho*Cl);
                                p[i][j] = 1;
                                //printf("%i frac=%f",i, p[i]);
                        }
                /*if (i ==i){
                        printf("Tn+1=%f en+1=%f\n\n", T[1], E[0]);
                }*/
                }
        }

        /*printf("enafterloop=%f\n", E[0]);
        E[0] = E0;
        if(p[1] == 1){p[0] = 1;}*/
        // Boundaries
        double k, R;
                // LEFT AND RIGHT
        for(int j = 1; j <= M; j++){
                k = conduct(p[1][j]);
                R = dx/(2*k);
                T[0][j] = Fx[0][j]*R + T[1][j];
                        // RIGHT
                k = conduct(p[M][j]);
                R = dx/(2*k);
                //T[M+1] = (R*h*Tinf - T[M])/(R*h - 1);
                T[M+1][j] = T[M][j] - Fx[M][j]*R;
                //T[M+1][j] = T0;
        }
```

```
                // DOWN AND UP
        for(int i = 1; i <= M; i++){
                k = conduct(p[i][1]);
                R = dx/(2*k);
                T[i][0] = Fy[i][0]*R + T[i][1];
                        // RIGHT
                k = conduct(p[i][M]);
                R = dx/(2*k);
                //T[M+1] = (R*h*Tinf - T[M])/(R*h - 1);
                T[i][M+1] = T[i][M] - Fy[i][M]*R;
                //T[i][M+1] = T0;
        }

// CORNERS
T[0][0] = (T[0][1] + T[1][0])/2.;
T[0][M+1] = (T[0][M] + T[1][M+1])/2.;
T[M+1][M+1] = (T[M][M+1] + T[M+1][M])/2.;
T[M+1][0] = (T[M][0] + T[M+1][1])/2.;

}
```