```
Input: vector x, vector y
Output: coefitiens, Matrix A

begin cubicSpline

    if x or y has duplicates:
        return error
    end if

    if lenght of x is not equals to lenght of y:
        return error
    end if

    set n = lenght of x
    set m = (n - 1) * 4
    set A = matrix[m][m]
    set B = vector[m]
    set A[0][0] = x[0] ^3
    set A[0][1] = x[0] ^2
    set A[0][2] = x[0]
    set A[0][3] = 1
    set B[0] = y[0]
        #interpolation conditions
    For i = 0,..., n      1
        set A[i+1][4*(i+1)-4] = math.pow(x[i+1], 3)
        set A[i+1][4*(i+1)-3] = math.pow(x[i+1], 2)
        set A[i+1][4*(i+1)-2] = x[i+1]
        set A[i+1][4*(i+1)-1] = 1
        set B[i+1] = y[i+1]
    #continuity conditions
    for i = 1,..., n      1
        set A[n-1+i][4*i-4] = math.pow(x[i], 3)
        set A[n-1+i][4*i-3] = math.pow(x[i], 2)
        set A[n-1+i][4*i-2] = x[i]
        set A[n-1+i][4*i-1] = 1
        set A[n-1+i][4*i] = -math.pow(x[i], 3)
        set A[n-1+i][4*i+1] = -math.pow(x[i], 2)
        set A[n-1+i][4*i+2] = -x[i]
        set A[n-1+i][4*i+3] = -1
        set B[n-1+i] = 0
    #softness condition
    for i = 1,..., n      1
        set A[2*n-3+i][4*i-4] = 3 * math.pow(x[i], 2)
        set A[2*n-3+i][4*i-3] = 2 * x[i]
        set set A[2*n-3+i][4*i-2] = 1
        set A[2*n-3+i][4*i-1] = 0
```

```
        set A[2*n–3+i][4*i]   = –3 * math.pow(x[i], 2)
        set A[2*n–3+i][4*i+1] = –2 * x[i]
        set A[2*n–3+i][4*i+2] = –1
        set A[2*n–3+i][4*i+3] = 0
        set B[2*n–3+i] = 0
    #concavity conditions
     for i = 1,..., n       1
        set A[3*n–5+i][4*i–4] = 6 * x[i]
        set A[3*n–5+i][4*i–3] = 2
        set A[3*n–5+i][4*i–2] = 0
        set A[3*n–5+i][4*i–1] = 0
        set A[3*n–5+i][4*i]   = –6 * x[i]
        set A[3*n–5+i][4*i+1] = –2
        set A[3*n–5+i][4*i+2] = 0
        set A[3*n–5+i][4*i+3] = 0
        set B[n+5+i]= 0
    #boundary conditions
     set A[m–2][0] = 6 * x[0]
     set A[m–2][1] = 2
     set A[m–1][m–4] = 6 * x[n–1]
     set A[m–1][m–3] = 2
     x = solveSystem(A, B)
     return x, A

end cubicSpline
```