```
  //Outer loop over vertex (test) functions:
  for i = 1,2 do {
    //If $>$ -1, $m_1$ is index of a test function $v_{m_1} \in V_{h,p}$,
    //i.e., row in $\boldsymbol{S}$:
    m1 := Elem[m].vert_dof[i];
    if (m1 > -1) then {
      //Inner loop over vertex (basis) functions:
      for j = 1,2 do {
        //If $>$ -1, $m_2$ is index of a basis function $v_{m_2} \in V_{h,p}$,
        //i.e., column in $\boldsymbol{S}$:
        m2 := Elem[m].vert_dof[j];
        if (m2 > -1) then
          S[m1][m2] := S[m1][m2] + a1*MESI[i][j]/Elem[m].jac
          + a0*Elem[m].jac*MEMI[i][j];
      } //End of inner loop over vertex functions
      //Inner loop over bubble (basis) functions:
      for j = 1,2,...,Elem[m].p-1 do {
        m2 := Elem[m].bubb_dof[j];
        S[m1][m2] := S[m1][m2] + a1*MESI[i][j+2]/Elem[m].jac
        + a0*Elem[m].jac*MEMI[i][j+2];
      } //End of inner loop over bubble functions
      //Contribution of the vertex test function $v_{m_1}$
      //to the right-hand side $\boldsymbol{F}$:
      F[m1] := F[m1] + $\int_{K_a} |J_{K_m}|\tilde{f}^{(m)}(\xi)\varphi_i(\xi)\,\mathrm{d}\xi$;
    } //End if (m1 > -1)
  } //End of outer loop over vertex functions
  //Outer loop over bubble (test) functions:
  for i = 1,2,...,Elem[m].p-1 do {
    m1 := Elem[m].bubb_dof[i];
    //Inner loop over vertex (basis) functions:
    for j = 1,2 do {
      m2 := Elem[m].vert_dof[j];
      if (m2 > -1) then
        S[m1][m2] := S[m1][m2] + a1*MESI[i+2][j]/Elem[m].jac
        + a0*Elem[m].jac*MEMI[i+2][j];
    } //End of inner loop over vertex functions
    //Inner loop over bubble (basis) functions:
    for j = 1,2,...,Elem[m].p-1 do {
      m2 := Elem[m].bubb_dof[j];
      S[m1][m2] := S[m1][m2] + a1*MESI[i+2][j+2]/Elem[m].jac
      + a0*Elem[m].jac*MEMI[i+2][j+2];
    } //End of inner loop over bubble functions
    //Contribution of the bubble test function $v_{m_1}$
    //to the right-hand side $\boldsymbol{F}$:
    F[m1] := F[m1] + $\int_{K_a} |J_{K_m}|\tilde{f}^{(m)}(\xi)\varphi_{i+2}(\xi)\,\mathrm{d}\xi$;
  } //End of outer loop over bubble functions
} //End of element loop
```

In Algorithm 2.5 we used the notation $\tilde{f}^{(m)}(\xi) = f(x_{K_m}(\xi))$. If the operator L is space- or time-dependent (for example, if the coefficient functions a_1 and a_0 in the model problem (2.20) are not constant), the precomputed `MESI` and `MEMI` arrays cannot be used. Instead, appropriate numerical quadrature must be performed each time the `MESI` or `MEMI` arrays in Algorithm 2.5 are accessed.

Efficient implementation of Algorithm 2.5 For the sake of transparency, large portion of Algorithm 2.5 (the application of a given test function to all vertex and bubble basis functions) was repeated two times with minor changes. This part of the code can be moved to a separate subroutine. Moreover, it is not necessary to store the full `Elem[m].bubb_dof`