C++ code 10.9.6: Initialization of sparse Galerkin matrix $\mathbf{A}$

```
Eigen::SparseMatrix<double> initializeA(unsigned int M) {
  // For the sake of efficiency the use of Eigen's sparse matrix data type is
  // essential.  The matrix is stored in CCS format.
  Eigen::SparseMatrix<double> A(M * M, M * M);
  // We already know that the matrix has at most 9 non-zero entries per row
  // and column.  This information is passed to Eigen via the reserve() member
  // function.
  A.reserve(Eigen::VectorXi::Constant(M * M, 9));
  // Iterate over all interior nodes of the mesh and apply the stencil and
  // initialize the matrix in column-wise order, from top to bottom in every
  // column, which is most efficient for the CCS storage format.
  for (int i = 0; i < M; ++i) {    // "vertical" loop
    for (int j = 0; j < M; ++j) {  // "horizontal" loop
      // Index of the current node
      const int k = i * M + j;
      // Self-interaction weight
      A.insert(k, k) = 16.0 / 6;
      // Interaction term with the node below to the left
      if (i > 0 && j > 0) {
        A.insert(k - M - 1, k) = -2.0 / 6;
      }
      // Interaction term with the node below
      if (i > 0) {
        A.insert(k - M, k) = -2.0 / 6;
      }
      // Interaction term with the node below to the right
      if (i > 0 && j < M - 1) {
        A.insert(k - M + 1, k) = -2.0 / 6;
      }
      // Interaction term with the node to the left
      if (j > 0) {
        A.insert(k - 1, k) = -2.0 / 6;
      }
      // Interaction term with the node to the right
      if (j < M - 1) {
        A.insert(k + 1, k) = -2.0 / 6;
      }
      // Interaction term with the node above to the left
      if (i < M - 1 && j > 0) {
        A.insert(k + M - 1, k) = -2.0 / 6;
      }
      // Interaction term with the node above
      if (i < M - 1) {
        A.insert(k + M, k) = -2.0 / 6;
      }
      // Interaction term with the node above to th right
      if (i < M - 1 && j < M - 1) {
        A.insert(k + M + 1, k) = -2.0 / 6;
      }
    }
  }
  A.makeCompressed();
  return A;
```