C++ code 11.8.28: Sub-problem (11-8.p): function `findCFLthreshold()` ⇝ GitLab

```
template <typename VECTORFIELD>
int findCFLthreshold(const lf::assemble::DofHandler &dofh, VECTORFIELD &&beta,
                     double T) {
  // Set upper and lower limit
  int M_upper = int((T / computeHmin(dofh.Mesh()))) + 1);
  int M_lower = 1;

  // Compute cell normals
  std::shared_ptr<lf::mesh::utils::CodimMeshDataSet<
      Eigen::Matrix<double, 2, Eigen::Dynamic>>>
      normal_vectors = AdvectionFV2D::computeCellNormals(dofh.Mesh());

  // Compute adjecent cells
  std::shared_ptr<lf::mesh::utils::CodimMeshDataSet<
      std::array<const lf::mesh::Entity *, 4>>>
      adjacentCells = AdvectionFV2D::getAdjacentCellPointers(dofh.Mesh());

  // Initialize a vector for the result and
  // randomly initialize a vector for the initial condition
  int num_dof = dofh.NumDofs();
  Eigen::VectorXd u0_h = Eigen::VectorXd::Random(num_dof);

  // Shift upper and lower bound until contition below isn't satisfied
  while ((M_upper - M_lower) > 2) {
    // Perform a simulation at M_middle
    // If it succeeds, set the upper bound to M_middle
    // Otherwise set the lower bound to M_middle
    // Repeat ...
    int M_middle = (M_upper + M_lower) / 2;
    try {
      solveAdvection2D(dofh, beta, u0_h, adjacentCells, normal_vectors, T,
                       M_middle);
      M_upper = M_middle;
    } catch (const std::exception &e) {
      M_lower = M_middle;
    }
  }

  int thres = M_upper;
  return thres;
}
```