C++ code 2.9.6: Sub-problem (2-9.h): Implementation of `convertDOFsLinearQuadratic()` ↬ GitLab

```cpp
Eigen::VectorXd convertDOFsLinearQuadratic(
    const lf::assemble::DofHandler &dofh_Linear_FE,
    const lf::assemble::DofHandler &dofh_Quadratic_FE,
    const Eigen::VectorXd &mu) {
  if (dofh_Linear_FE.Mesh() != dofh_Quadratic_FE.Mesh()) {
    throw "Underlying meshes must be the same for both DOF handlers!";
  }
  std::shared_ptr<const lf::mesh::Mesh> mesh =
      dofh_Linear_FE.Mesh();  // get the mesh
  // coefficient vector for returning the result
  Eigen::VectorXd zeta(dofh_Quadratic_FE.NumDofs());
  // Play safe:  always set zero if you're not sure to set every entry
      later
  // on for us this shouldn't be a problem, but just to be sure
  zeta.setZero();
  for (const auto *cell : mesh->Entities(0)) {
    // check if the spaces are actually linear and quadratic
    if (dofh_Linear_FE.NumLocalDofs(*cell) != 3 ||
        dofh_Quadratic_FE.NumLocalDofs(*cell) != 6) {
      throw std::runtime_error(
          "dofh_Linear_FE must have 3 dofs per cell and dofh_Quadratic_FE 6!");
    }
    // get the global dof indices of the linear and quadratic FE spaces,
        note
    // that the vectors obey the LehrFEM++ numbering, which we will make
        use of
    // lin_dofs will have size 3 for the 3 dofs on the nodes and
    // quad_dofs will have size 6, the first 3 entries being the nodes
        and
    // the last 3 the edges
    std::span<const lf::assemble::gdof_idx_t> lin_dofs =
        dofh_Linear_FE.GlobalDofIndices(*cell);
    std::span<const lf::assemble::gdof_idx_t> quad_dofs =
        dofh_Quadratic_FE.GlobalDofIndices(*cell);
    for (std::size_t l = 0; l <= 2; ++l) {
      // Let $p_1,p_2,p_3$ be the nodes of the triangle and $p_4,p_5,p_6$ the
      // edge midpoints.  Let $\lambda_1,\lambda_2,\lambda_3$ be the local basis
      // functions of $\mathcal{S}_1^0$ and $b_1,..,b_6$ for $\mathcal{S}_2^0$.  The values of
      // $\lambda_i$ in the interpolation nodes $p_1,..,p_6$ are the
      // coefficients of writing $\lambda_i$ as lin.  comb of $b_1,..b_6$.
      // From 2-9.a we know $\lambda_l(p_l) = 1;\ l = 1,2,3$.
      // Hence we simply copy the coefficient of $\lambda_l$ to $b_l$
      // for $l = 1,2,3$.
      zeta(quad_dofs[l]) = mu(lin_dofs[l]);
      // And $\lambda_l(p_l+3) = 0.5,\ \lambda_{(l+1)mod3}(p_l+3) = 0.5;$
      // $l = 1,2,3$.  Hence we copy 0.5 of the respective coefficients!
      zeta(quad_dofs[l + 3]) =
          0.5 * (mu(lin_dofs[l]) + mu(lin_dofs[(l + 1) % 3]));
    }
  }
  return zeta;
}
```