```cpp
#include "GenericDomain.hpp"
#include "GenericMesh.hpp"
#include "MeshImport_Triangle.hpp"
#include <iostream>
#include "TriangleRefiner.hpp"

using namespace GeDiM;
using namespace Eigen;

int main(int argc, char *argv[])
{
        /// PARAMETRI
        double cellsize;
        int percentuale;
        if (argc != 3)
        {
                cerr << "Uso corretto: progetto.bin <cellsize> <percentuale>\n";
                exit(EXIT_FAILURE);
        }
        else
        {
                cellsize = stod(argv[1]);
                percentuale = stoi(argv[2]);
        }

        /// CREATE DOMAIN
        const unsigned int numDomainVertices = 4;
        GenericDomain2D domain(0, numDomainVertices);
        vector<Vector3d> vertexCoords(numDomainVertices);
        vertexCoords[0] << 0.0, 0.0, 0.0;
        vertexCoords[1] << 1.0, 0.0, 0.0;
        vertexCoords[2] << 1.0, 1.0, 0.0;
        vertexCoords[3] << 0.0, 1.0, 0.0;
        for (unsigned int i = 0; i < numDomainVertices; i++)
        {
                domain.AddVertex(vertexCoords[i]);
                domain.AddEdge(i, (i + 1) % numDomainVertices);
        }
        domain.Initialize();

        /// MESH DOMAIN
        MeshImport_Triangle meshCreator;
        meshCreator.SetMaximumCellSize(cellsize);
        meshCreator.CreateTriangleInput(domain);
        meshCreator.CreateTriangleOutput(domain);
        GenericMesh mesh;
        meshCreator.CreateMesh(domain, mesh);
        Output::PrintGenericMessage("Triangle ha prodotto una mesh contenente %d triangoli, %d nodi e %d lati", true, mesh.NumberOfCells(), mesh.NumberOfPoints(), mesh.NumberOfEdges());

        /// INPUT MESH TO MATLAB SCRIPT FOR VISUALIZATION
        ofstream file("plotTriangleMesh.m", ofstream::out);
        file << "nodesBefore = [";
        for (unsigned int i = 0; i < mesh.NumberOfPoints(); i++)
                file << mesh.Point(i)->Coordinates()(0) << "," << mesh.Point(i)->Coordinates()(1) << ";" << endl;
        file << "];" << endl;

        file << "trianglesBefore = [";
        for (unsigned int i = 0; i < mesh.NumberOfCells(); i++)
        {
                file << mesh.Cell(i)->Point(0)->Id() + 1 << "," << mesh.Cell(i)->Point(1)->Id() + 1 << "," << mesh.Cell(i)->Point(2)->Id() + 1 << ";" << endl;
        }
        file << "];" << endl;
        file << "figure;trimesh(trianglesBefore, nodesBefore(:,1), nodesBefore(:,2));" << endl;

        /// REFINE MESH
        TriangleRefiner refiner(mesh);
        srand(1);
        for (unsigned i = 0; i < mesh.NumberOfCells(); i++)
                if (rand() % 100 < percentuale)
                        refiner.PrepareTriangle(i);

        refiner.RefineMesh();
        refiner.AggiornaInformazioniPunti();

        /// OUTPUT MESH TO MATLAB SCRIPT FOR VISUALIZATION
        mesh.CleanInactiveTreeNode();
        file << "nodesAfter = [";
        for (unsigned int i = 0; i < mesh.NumberOfPoints(); i++)
                file << mesh.Point(i)->Coordinates()(0) << "," << mesh.Point(i)->Coordinates()(1) << ";" << endl;
        file << "];" << endl;

        file << "trianglesAfter = [";
        for (unsigned int i = 0; i < mesh.NumberOfCells(); i++)
        {
                file << mesh.Cell(i)->Point(0)->Id() + 1 << "," << mesh.Cell(i)->Point(1)->Id() + 1 << "," << mesh.Cell(i)->Point(2)->Id() + 1 << ";" << endl;
        }
        file << "];" << endl;
        file << "figure;trimesh(trianglesAfter, nodesAfter(:,1), nodesAfter(:,2));" << endl;
        file << "figure;"
                 << "hold on;" << "trimesh(trianglesAfter, nodesAfter(:,1), nodesAfter(:,2),'color', 'r');"
                 << "trimesh(trianglesBefore, nodesBefore(:,1), nodesBefore(:,2),'color', 'b');" << "hold off;" << endl;
        file.close();
}

```