```cpp
#include "/src/Definitions.h"
#include "/src/ElementTests.h"

#include "MaterialModelBar1D.h"
#include "TwoNodeBar.h"
#include "ExternalForces.h"
// Typedef based on MaterialModel1DBar
typedef MaterialModels::MaterialModel1DBar      MaterialModel;

// Typedef's based on FiniteBar3D
typedef Elements::FiniteBar3D<MaterialModel>    Element;
typedef Elements::Properties                    ElementProperties;
typedef Element::Node                           Node;
typedef Element::Vector                         Vector;
typedef Element::Stress                         Stress;
typedef Element::Strain                         Strain;

// Typedef based on the ConstantBodyForce-element
typedef Elements::ExternalForce::ConstantBodyForce<Element> ConstantBodyForce;

int main(int arc, char *argv[]) {

  ignoreUnusedVariables(arc,argv);
  array<Node, 2> exampleNodes;

  // 1D Bar Material Model initialisation
  const double youngsModulus = 1.0;
  MaterialModel     materialModel(youngsModulus);

  // Two Node Bar Element initialisation
  const double area = 1.0;
  const double density = 1.0;
  ElementProperties elementProperties(area,density); // discuss with Dennis


  // Test 1 : TwoNodeBar Test
  Vector exampleNodePosition0 = Vector::Random();
  Vector exampleNodePosition1 = Vector::Random();

  exampleNodes[0] = Node(0,exampleNodePosition0);
  exampleNodes[1] = Node(1,exampleNodePosition1);

  Element finiteKinematicsElement(exampleNodes, elementProperties, &materialModel);
  Elements::testElementDerivatives(finiteKinematicsElement);

  // Test 2 : ConstantBodyForce Test
  Vector gravityForceVector = Vector::Random();

  ConstantBodyForce gravityElement(finiteKinematicsElement,gravityForceVector);
  Elements::testElementDerivatives(gravityElement);

  return 0;
}

```