```
# Mesh Generation:
a, b, c, nx, ny, nz = 1.0079, 0.6283, 1.0, 20, int(0.6283/1.0 * 20), 20
mesh2d = RectangleMesh(nx, ny, a, b, quadrilateral=True) # Rectangular 2D mesh
mesh = ExtrudedMesh(mesh2d, nz)
bottom, top, left, right, front, back = "bottom", "top", 1, 2, 3, 4

# ------------------------------------------------------------------------------------------------------------
# Initial condition and constants:
Told.interpolate(0.5*(erf((1-X[2])*4)+erf(-X[2]*4)+1) + 0.2*(cos(pi*X[0]/a)+cos(pi*X[1]/b))*sin(pi*X[2]))
Ra = Constant(3e4) # Rayleigh number
k = Constant((0, 0, 1)) # Unit vector (in direction opposite to gravity).

# ------------------------------------------------------------------------------------------------------------
# Stokes Equation Solver Parameters:
stokes_solver_parameters = {
    "mat_type": "matfree",
    "snes_type": "ksponly",
    "ksp_type": "preonly",
    "pc_type": "fieldsplit",
    "pc_fieldsplit_type": "schur",
    "pc_fieldsplit_schur_type": "full",
    "fieldsplit_0": {
        "ksp_type": "cg",
        "ksp_rtol": 1e-7,
        "pc_type": "python",
        "pc_python_type": "firedrake.AssembledPC",
        "assembled_pc_type": "gamg",
        "assembled_pc_gamg_threshold": 0.01,
        "assembled_pc_gamg_square_graph": 100,
    },
    "fieldsplit_1": {
        "ksp_type": "fgmres",
        "ksp_rtol": 1e-6,
        "pc_type": "python",
        "pc_python_type": "firedrake.MassInvPC",
        "Mp_ksp_rtol": 1e-5,
        "Mp_ksp_type": "cg",
        "Mp_pc_type": "sor",
    } }

# Energy Equation Solver Parameters:
energy_solver_parameters = {
    "mat_type": "aij",
    "snes_type": "ksponly",
    "ksp_type": "gmres",
    "ksp_rtol": 1e-7,
    "pc_type": "sor", }

# ------------------------------------------------------------------------------------------------------------
# Set up boundary conditions:
bcvfb = DirichletBC(Z.sub(0).sub(1), 0, (front, back))
bcvlr = DirichletBC(Z.sub(0).sub(0), 0, (left, right))
bcvbt = DirichletBC(Z.sub(0), 0, (bot,top))
bctb, bctt = DirichletBC(Q, 1.0, bot), DirichletBC(Q, 0.0, top)

# ------------------------------------------------------------------------------------------------------------
# Generating near_nullspaces for GAMG:
x_rotV = Function(V).interpolate(as_vector((0, X[2], -X[1])))
y_rotV = Function(V).interpolate(as_vector((-X[2], 0, X[0])))
z_rotV = Function(V).interpolate(as_vector((-X[1], X[0], 0)))
nns_x = Function(V).interpolate(Constant([1., 0., 0.]))
nns_y = Function(V).interpolate(Constant([0., 1., 0.]))
nns_z = Function(V).interpolate(Constant([0., 0., 1.]))
V_near_nullspace = VectorSpaceBasis([nns_x, nns_y, nns_z, x_rotV, y_rotV, z_rotV])
V_near_nullspace.orthonormalize()
Z_near_nullspace = MixedVectorSpaceBasis(Z, [V_near_nullspace, Z.sub(1)])

# ------------------------------------------------------------------------------------------------------------
# Updated solve setup:
stokes_problem = NonlinearVariationalProblem(F_stokes, z, bcs=[bcvbt, bcvfb, bcvlr])
stokes_solver = NonlinearVariationalSolver(stokes_problem, solver_parameters=stokes_solver_parameters, appctx={"
    mu": mu}, nullspace=p_nullspace, transpose_nullspace=p_nullspace, near_nullspace=Z_near_nullspace)
energy_problem = NonlinearVariationalProblem(F_energy, Tnew, bcs=[bctb, bctt])
energy_solver = NonlinearVariationalSolver(energy_problem, solver_parameters=energy_solver_parameters)

# ------------------------------------------------------------------------------------------------------------
# Updated diagnostics:
nusselt_number_top = -1. * assemble(dot(grad(Tnew), n) * ds_t) * (1./assemble(Tnew * ds_b))
```