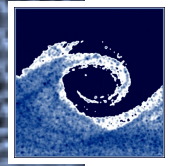


Assignments and laboratory tasks

Simulate a laminar pipe flow with 2 mm pipe inner diameter and 6 mm length.

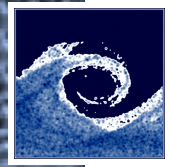
Create a 2D axisymmetric model of the flow using uniform 1/15 mm cell size. Use uniform 1 m/s inlet velocity and study entrance lenght effect. Develop an OpenFOAM utility to create "parabolic" inlet velocity profile. Re-run simulation with parabolic inlet velocity profile. Determine how the entrance length changes.

Optionally keep track of file changes using git.

Debug the "parabolic" utility using GDB.

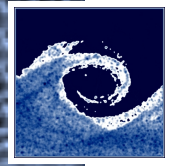
Help with blockMeshDict (1/2)

```
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

convertToMeters 1e-3;

vertices
(
    (0 0 0)
    (0 1 -0.04)
    (0 1  0.04)
    (6 0 0)
    (6 1 -0.04)
    (6 1  0.04)
);

blocks
(
    hex (0 3 4 1 0 3 5 2) (90 15 1) simpleGrading (1 1 1)
);
```



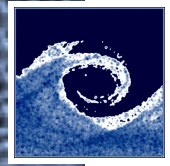
Help with blockMeshDict (2/2)

```
boundary
(
    inlet
    {
        type patch;
        faces
        ((0 1 2 0));
    }
    outlet
    {
        type patch;
        faces
        ((3 4 5 3));
    }
    wall
    {
        type wall;
        faces
        ((1 4 5 2));
    }
    axis
    {
        type empty;
        faces
        ((0 3 3 0));
    }
    front
    {
        type wedge;
        faces
        ((0 3 4 1));
    }
    back
    {
        type wedge;
        faces
        ((0 3 5 2));
    }
);                                                                    //
```



Help with parabolic.C (1/2)

```
#include "fvCFD.H"

int main(int argc, char *argv[])
{
    argList::addOption("Ubar", "scalar", "average velocity [m/s]");
    argList::addOption("R", "scalar", "profile radius [m]");
    argList::validArgs.append("patchName");
    #include "setRootCase.H"
    #include "createTime.H"
    #include "createMesh.H"

    scalar Ubar(args.optionRead<scalar>("Ubar"));
    scalar R(args.optionRead<scalar>("R"));
    vector axis(1, 0, 0);
    const word patchName = args[1];

    Info<< "Time = " << runTime.timeName() << nl << endl;

    Info<< "Reading field U" << endl;
    volVectorField U
    (
        IOobject
        (
            "U",
            runTime.timeName(),
            mesh,
            IOobject::MUST_READ,
            IOobject::NO_WRITE
        ),
        mesh
    );                                                                    //
```



Help with parabolic.C (2/2)

```
    const fvPatchList& patches = mesh.boundary();
    label pi = mesh.boundaryMesh().findPatchID(patchName);
    if (pi < 0)
    {
            FatalError
                    << "Unable to find patch "<< patchName << nl
                    << exit(FatalError);
    }
    const fvPatch& currPatch = patches[pi];

    scalarField r(mag(currPatch.Cf() - ((currPatch.Cf() & axis)/magSqr(axis)) * axis));
    vectorField velocityProfile(2*Ubar*(1-sqr(r/R)) * -currPatch.nf());

    forAll(U.boundaryField()[pi], i)
    {
        U.boundaryField()[pi][i] = velocityProfile[i];
    }

    Info<< "Writing U\n" << endl;
    U.write();

    Info<< "End\n" << endl;

    return 0;
}
```