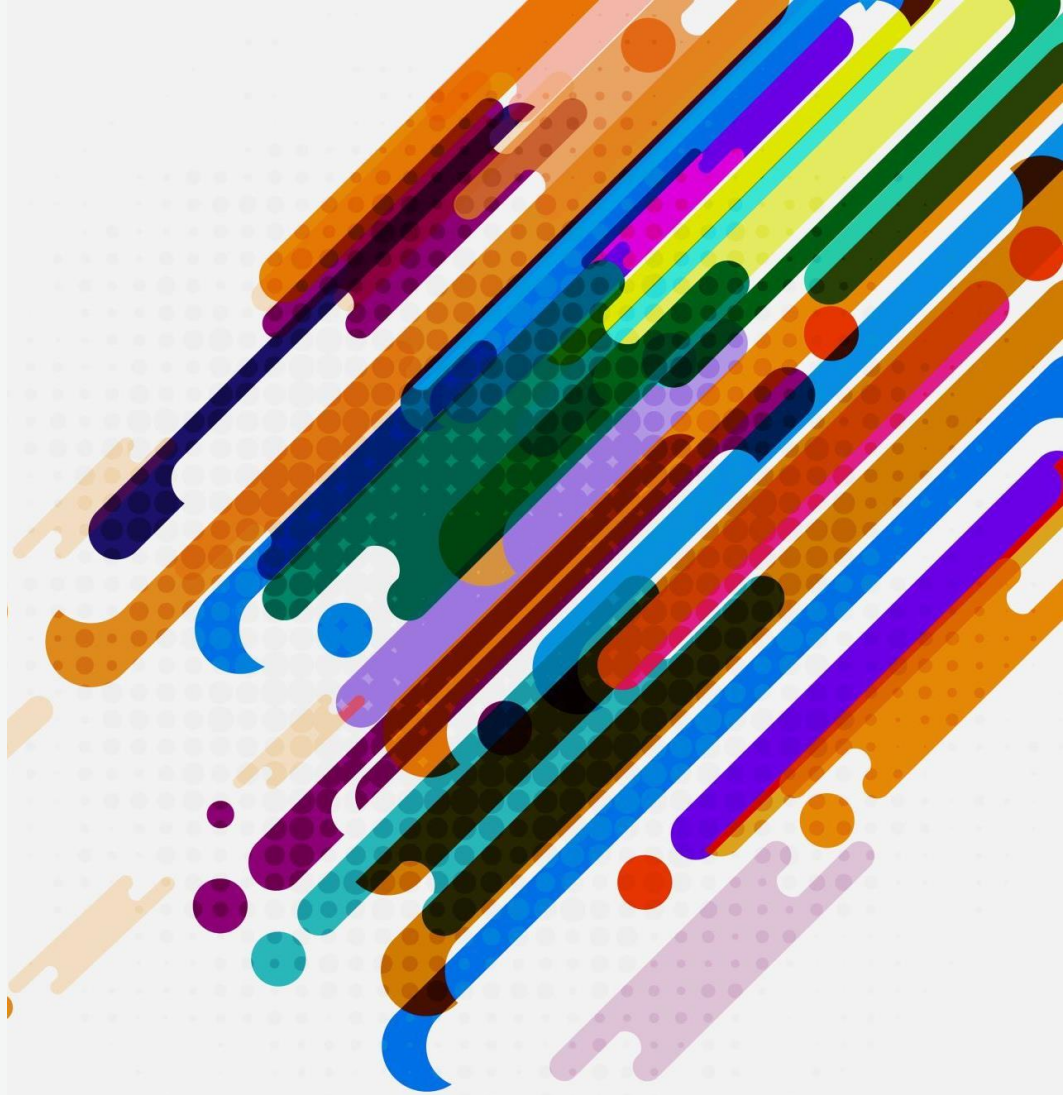
OpenFOAM Catalyst

Ali Shayegh

TensorFields, University College Dublin



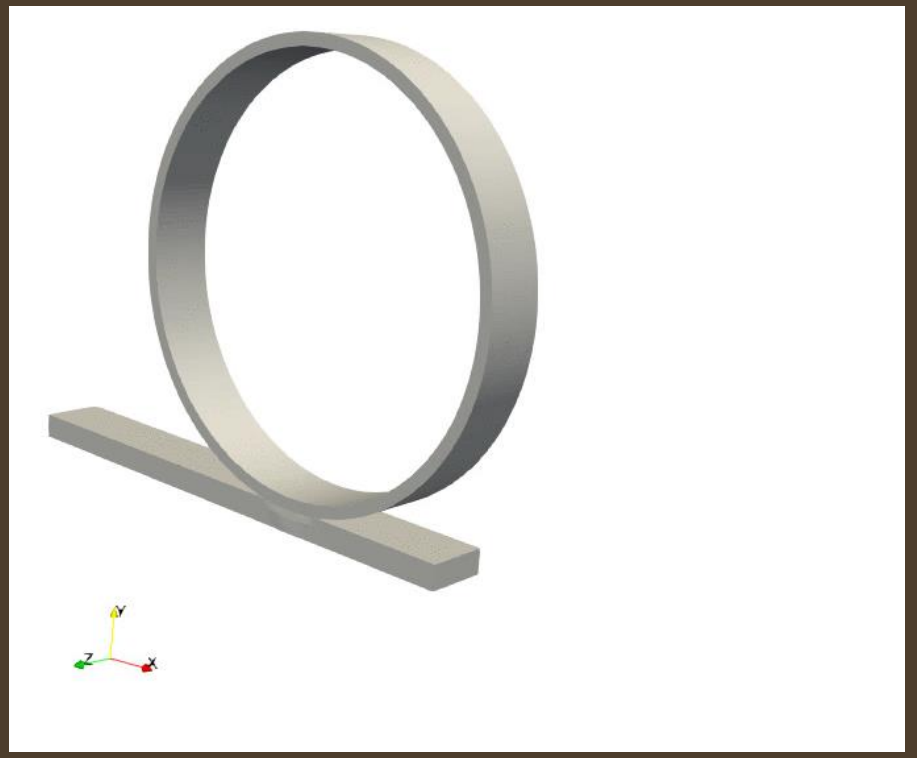
University College Dublin

Tensorfields

ABOUT US

Ali Shayegh

- Researcher at **UCD**
- Seven years of **OpenFOAM** experience
- Senior member on **cfd-online**



Goal

- Create engineering-standard applications.

Goal

- Create engineering-standard applications.

Agenda

- Write a plain bullet point plan for solving a **PDE**,

Goal

- Create engineering-standard applications.

Agenda

- Write a plain bullet point plan for solving a **PDE**,
- Re-write the plan using **C++** syntax,

Goal

- Create engineering-standard applications.

Agenda

- Write a plain bullet point plan for solving a **PDE**,
- Re-write the plan using **C++** syntax,
- Develop necessary **libraries**,

Goal

- Create engineering-standard applications.

Agenda

- Write a plain bullet point plan for solving a **PDE**,
- Re-write the plan using **C++** syntax,
- Develop necessary **libraries**,
- Glue everything together: Compile your **solver**,

Goal

- Create engineering-standard applications.

Agenda

- Write a plain bullet point plan for solving a **PDE**,
- Re-write the plan using **C++** syntax,
- Develop necessary **libraries**,
- Glue everything together: Compile your **solver**,
- Re-write the solver using **core OpenFOAM libraries**,

Goal

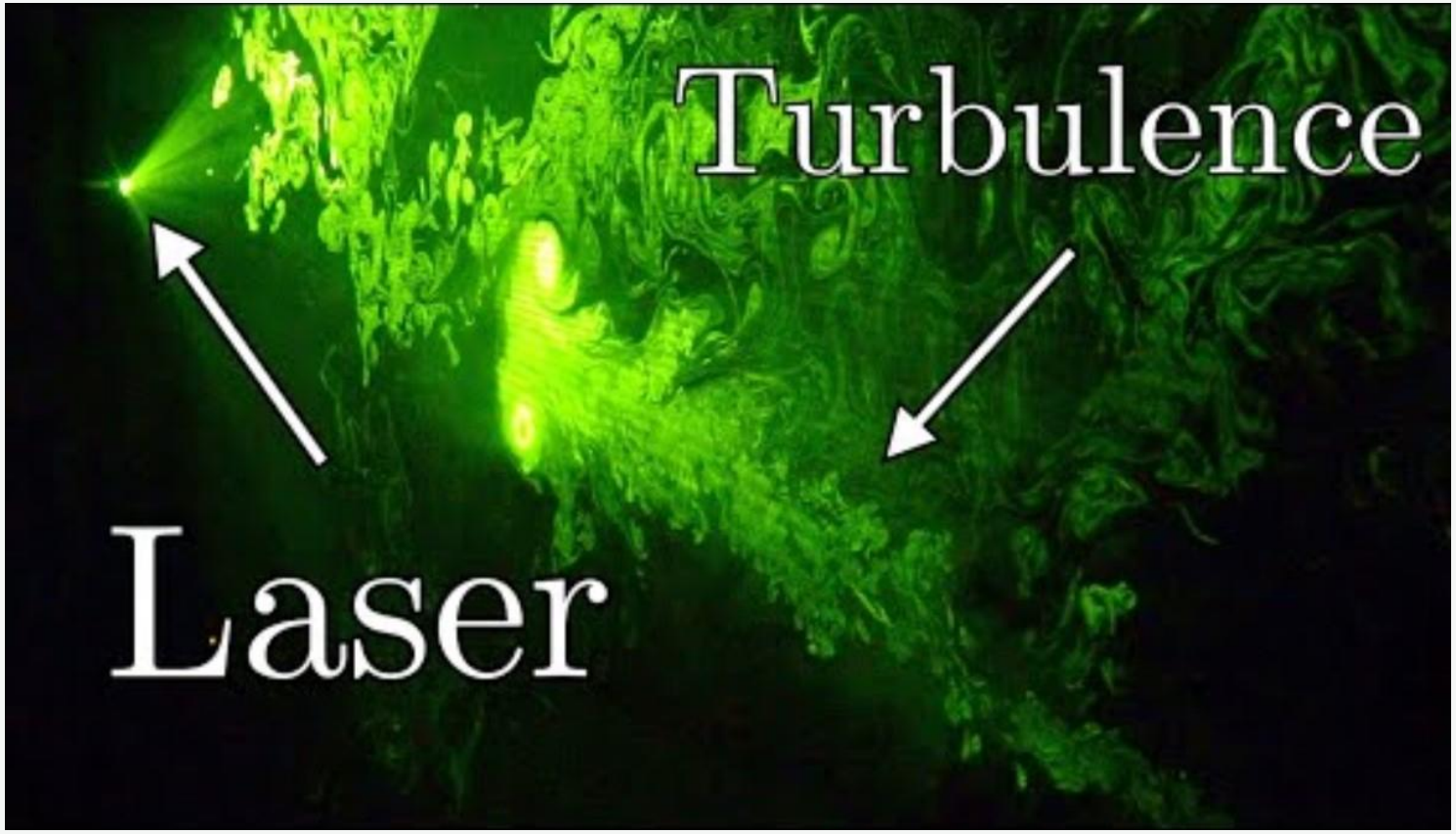
- Create engineering-standard applications.

Agenda

- Write a plain bullet point plan for solving a **PDE**,
- Re-write the plan using **C++** syntax,
- Develop necessary **libraries**,
- Glue everything together: Compile your **solver**,
- Re-write the solver using **core OpenFOAM libraries**,
- Build on the top of OpenFOAM: **New Solver and Boundary Conditions**

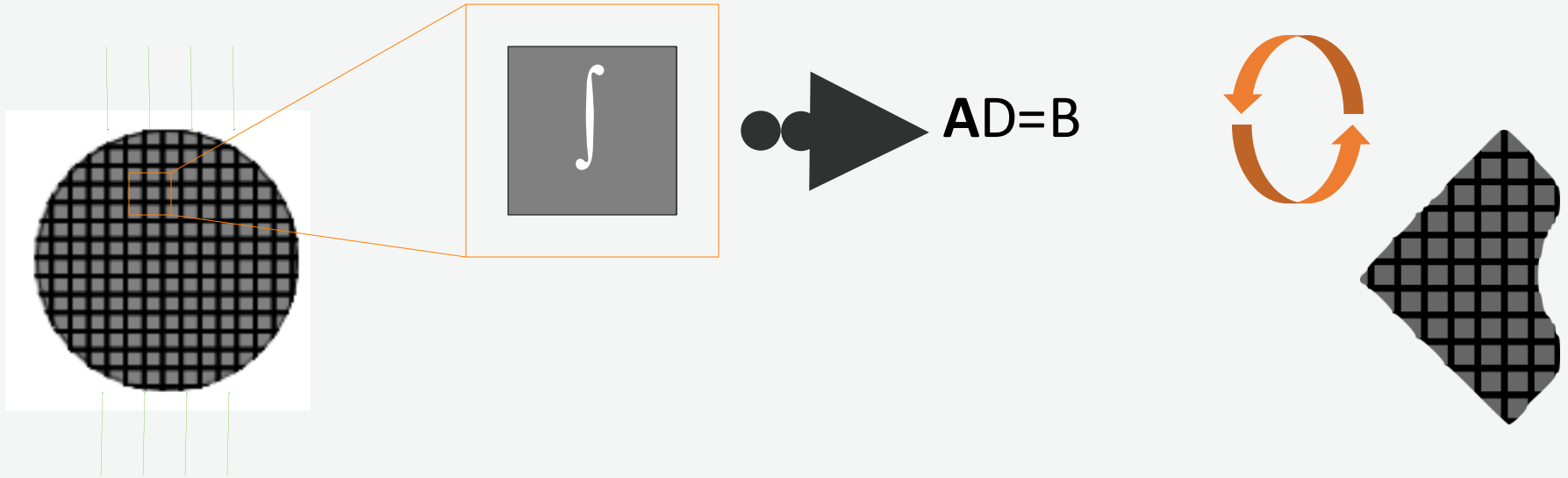
OpenFOAM; what is that?!

What is a PDE?



3Blue1Brown, Youtube

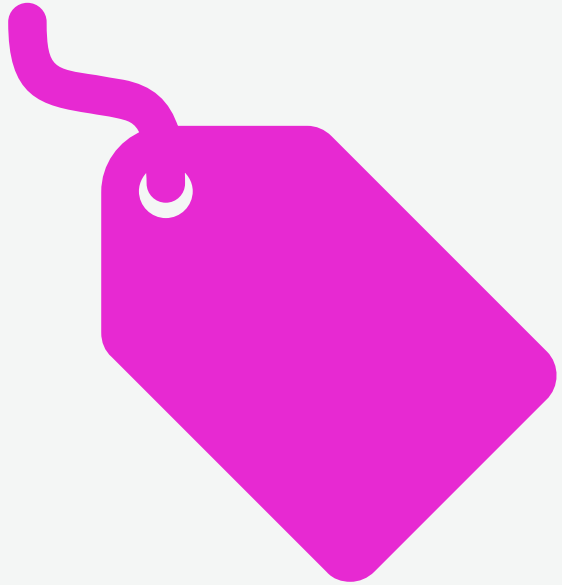
- A software to solve PDEs using Finite Volume Method



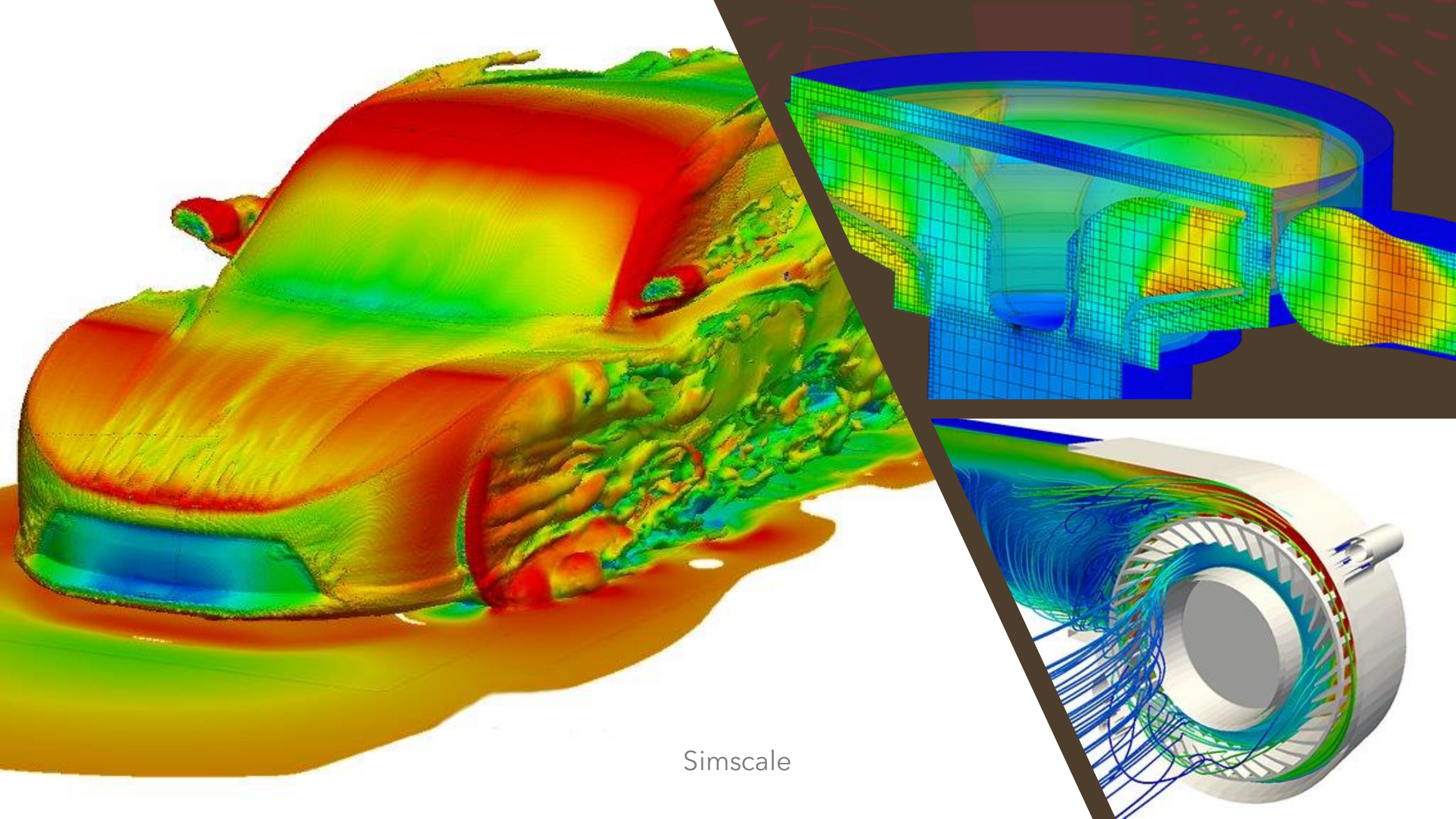
$$\rho \frac{d^2u}{dt^2} = \nabla . T + \rho f$$

Advantages

- Free price tag
- Open-source



Is It Industry-Standard?



Simscale

Any Downsides?

Any Downsides?

REVENUE STREAM

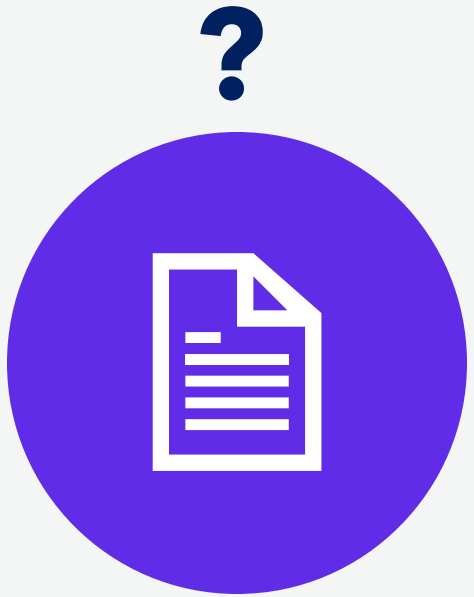
Any Downsides?

REVENUE STREAM

DOCUMENTATION

Any Downsides?

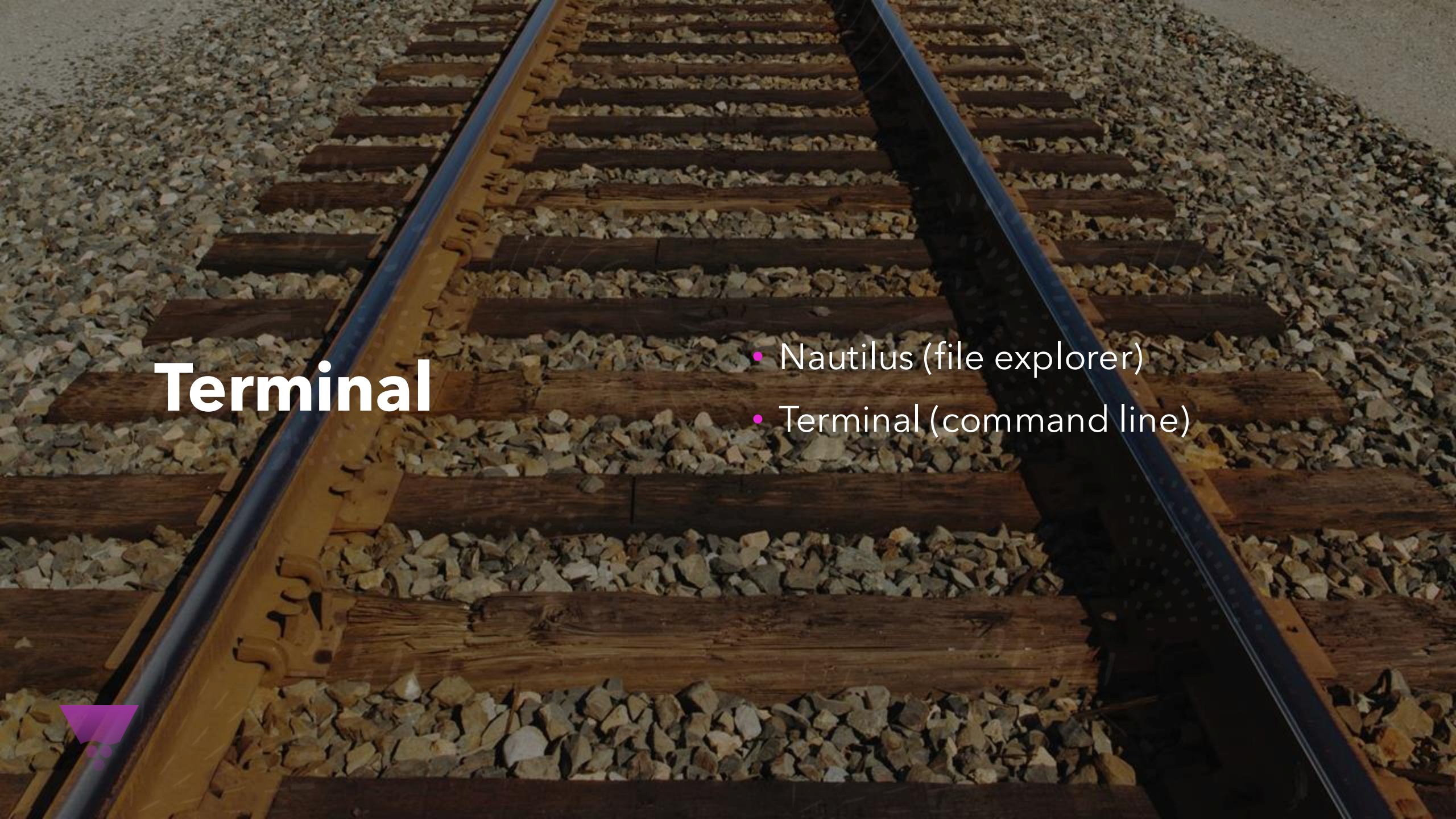
REVENUE STREAM



DOCUMENTATION

Linux

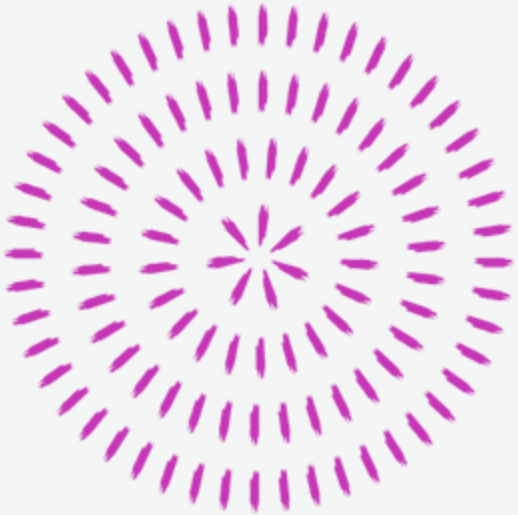
- Multiple distros
- Each distro releases new versions



Terminal

- Nautilus (file explorer)
- Terminal (command line)

Installation



A Quick Review of C++

main()

Juicer :

```
lemonde
```

main()

Juicer :

```
lemonde juicer(lemons)
```

main()

Juicer :

```
lemonde juicer(lemons)
{
    peel(lemons);
```

main()

Juicer :

```
lemonde juicer(lemons)
{
    peel(lemons);
    cut(lemons);
```

main()

Juicer :

```
lemonde juicer(lemons)
{
    peel(lemons);
    cut(lemons);
    ...
```

main()

Juicer :

```
lemonde juicer(lemons)
{
    peel(lemons);
    cut(lemons);
    ...
    return (lemonade);
}
```

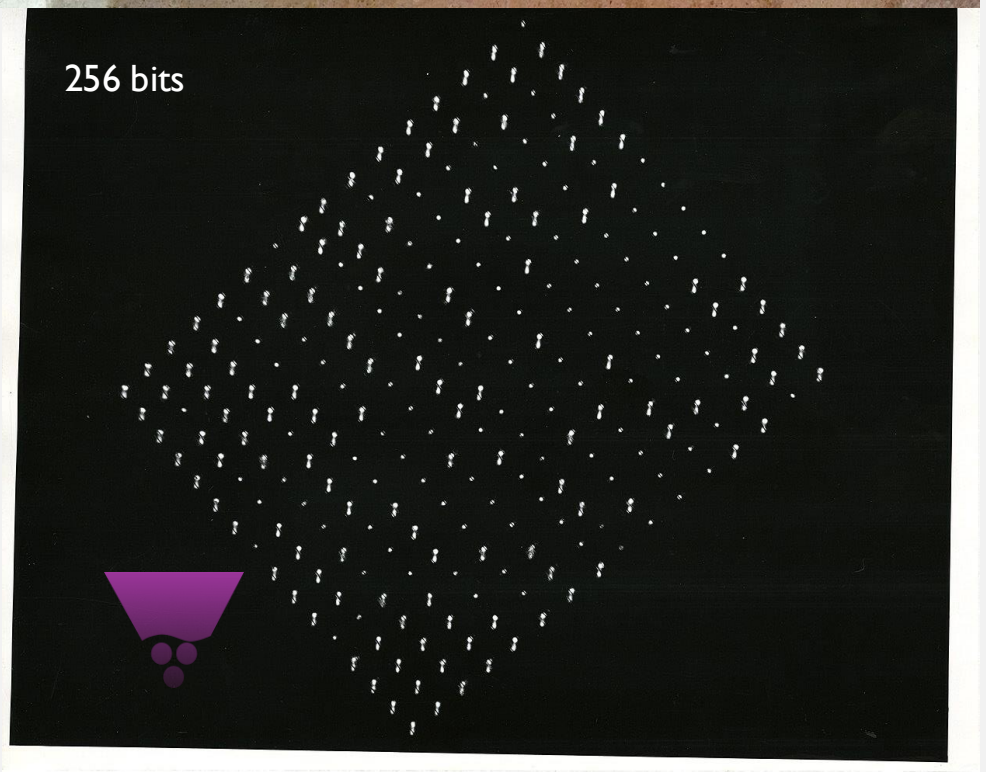
main()

Juicer :

```
lemonde juicer(lemons)
{
    peel(lemons);
    cut(lemons);
    ...
    return (lemonade);
}
```

file.C :

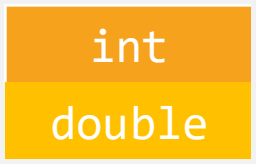
```
int main() {}
```



main()

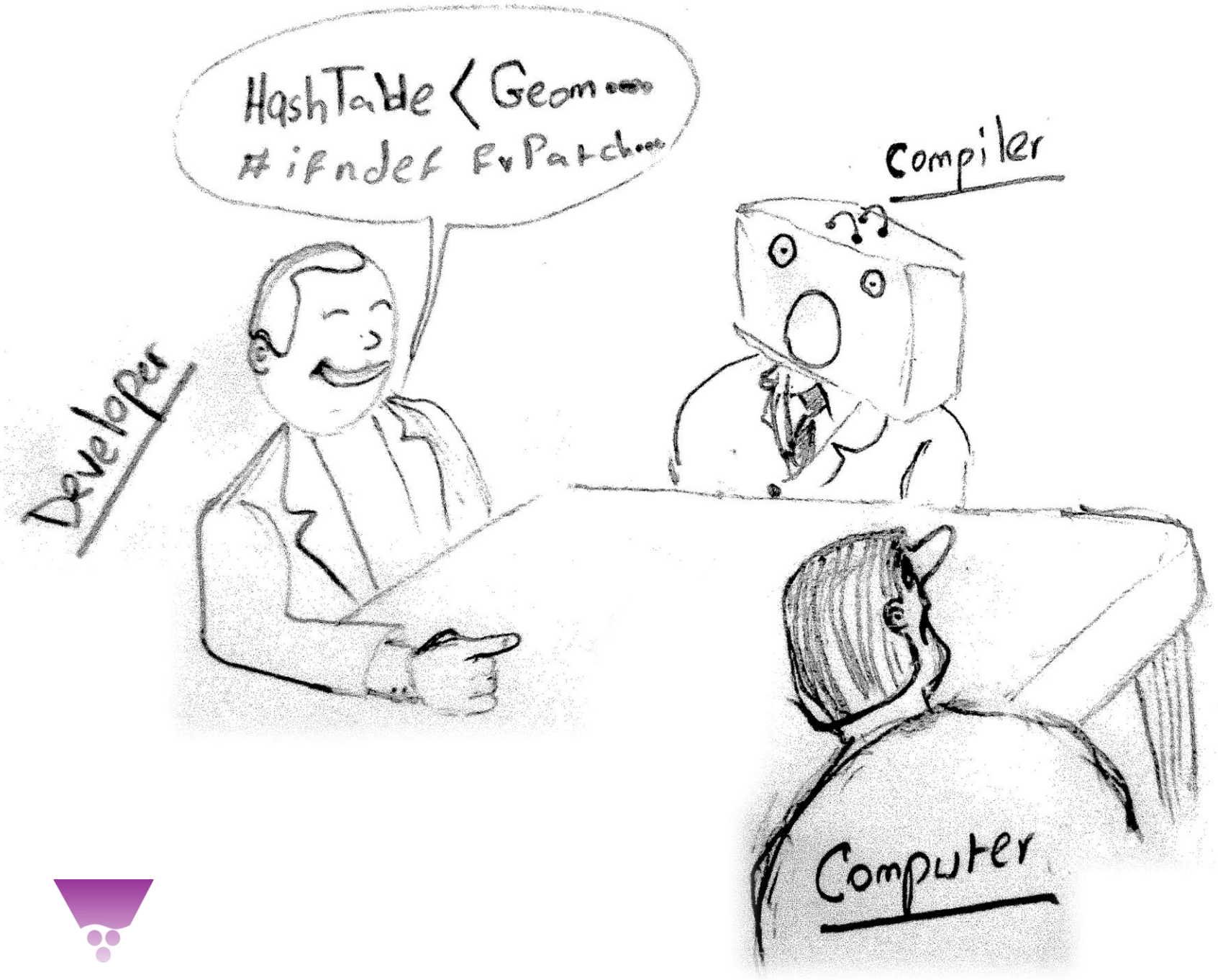
file.C :

```
int main() {}
```



* https://cplusplus.com/doc/tutorial/variables

Figures: https://en.wikipedia.org/wiki/Williams_tube

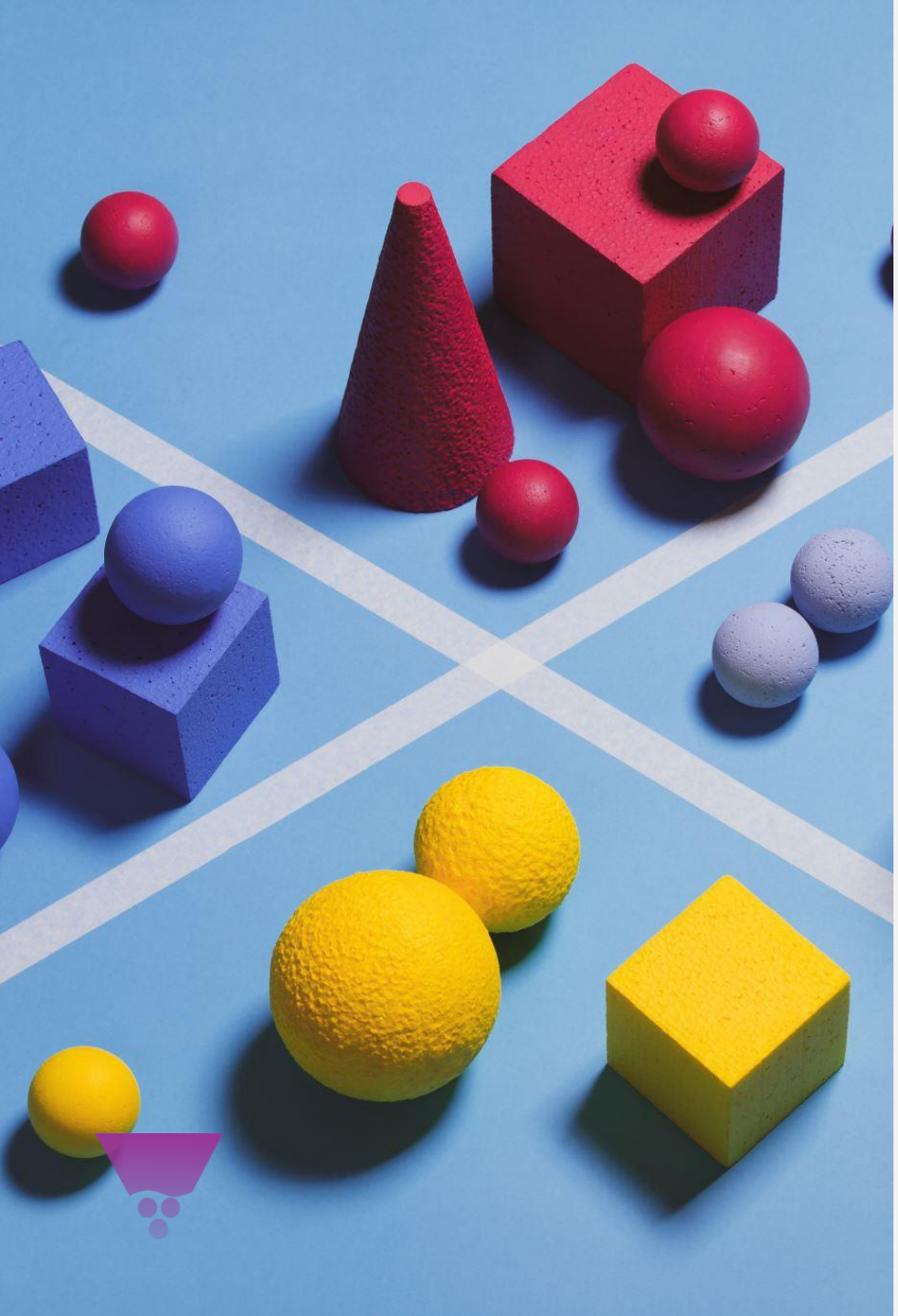


COMPILING: BASIC

simplest.C :

```
int main() {}
```

```
g++ -o file.o file.C
```



COMPILING: BASIC

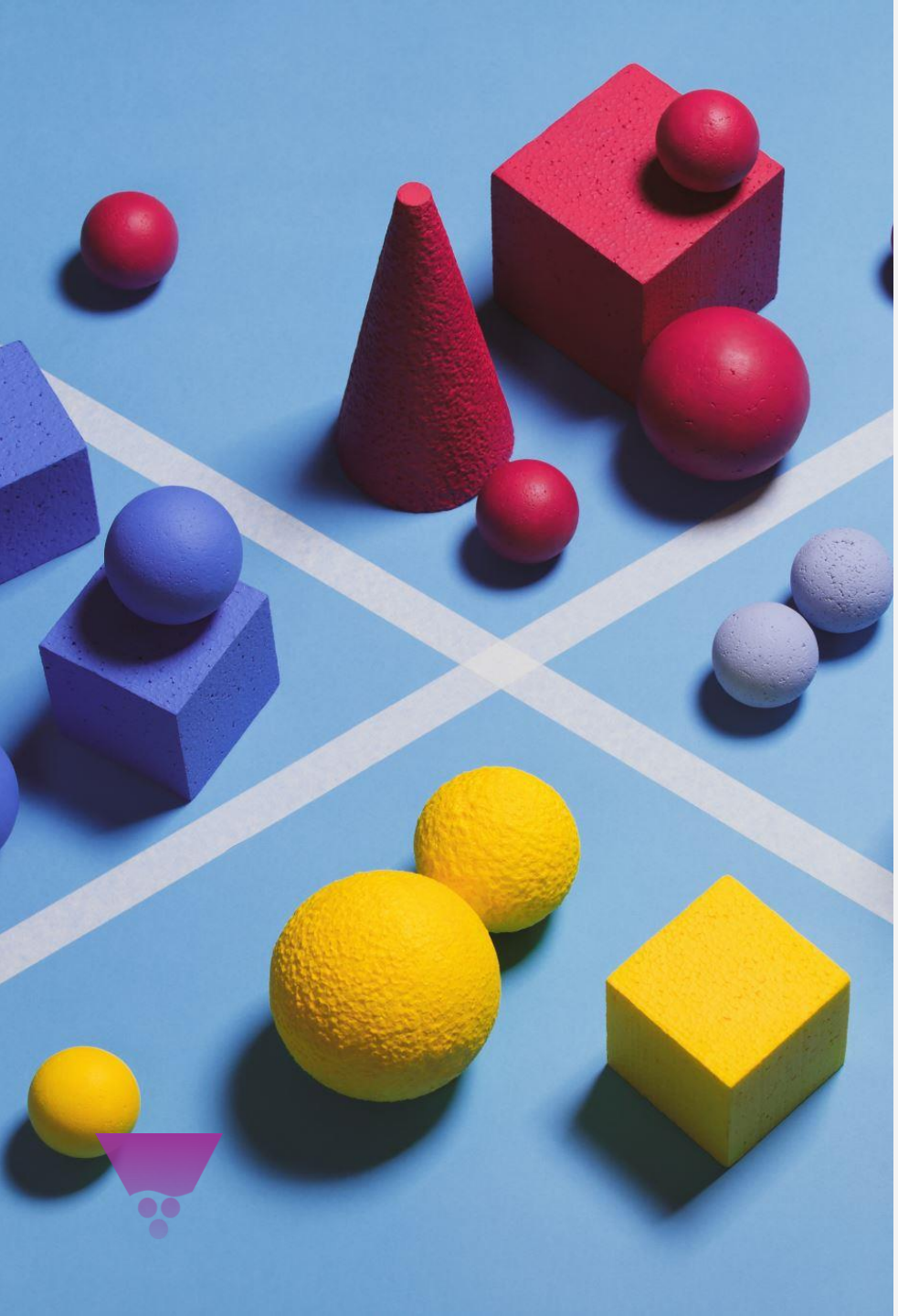
simplest.C :

```
int main() {}
```

```
g++ -o file.o file.C
```

Object code

Source code



COMPILING: BASIC

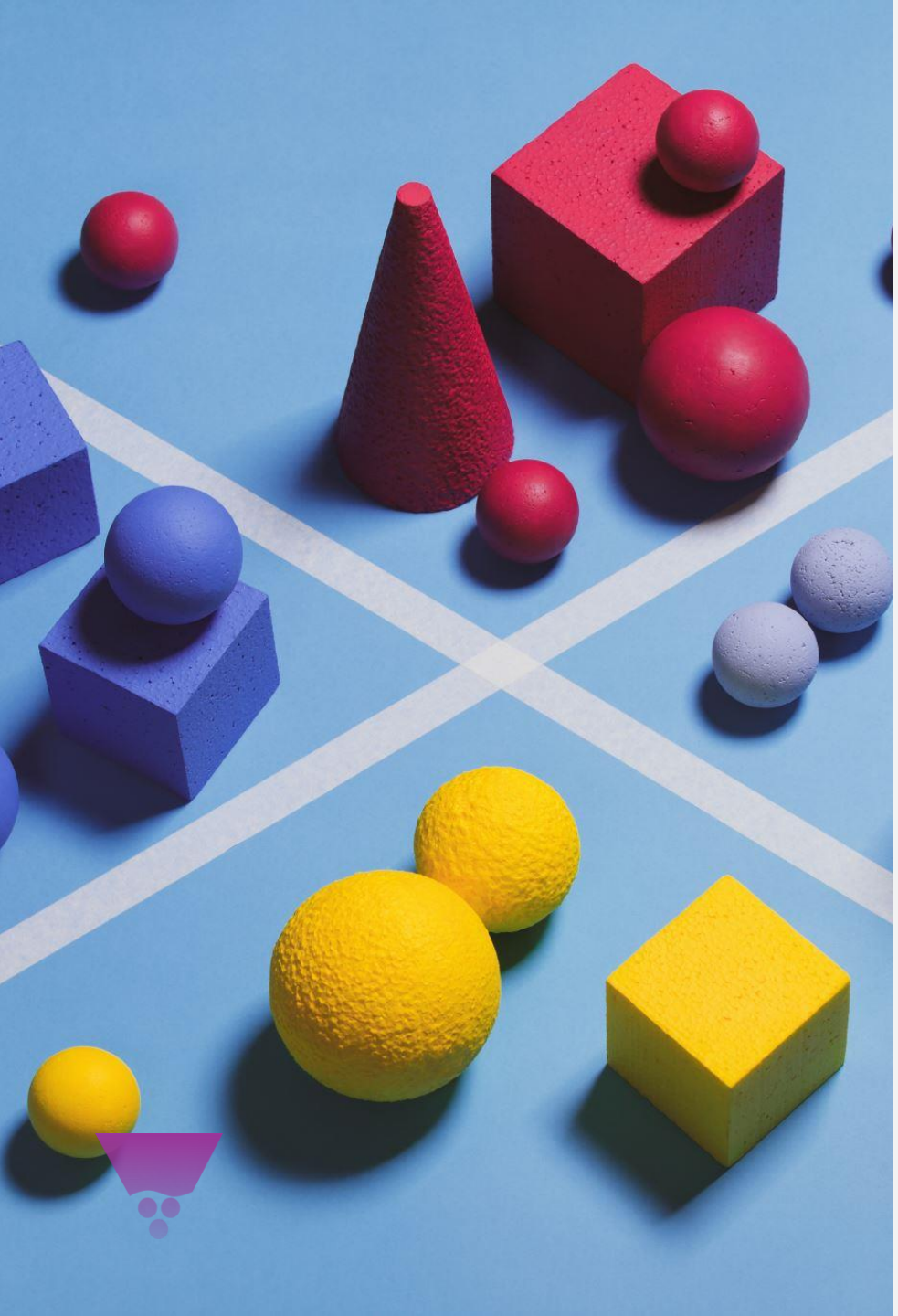
simplest.C :

```
int main() {}
```

```
g++ -o file.o file.C
```

file.cc
file.cp
file.cxx
file.cpp
file.CPP
file.c++
File.C

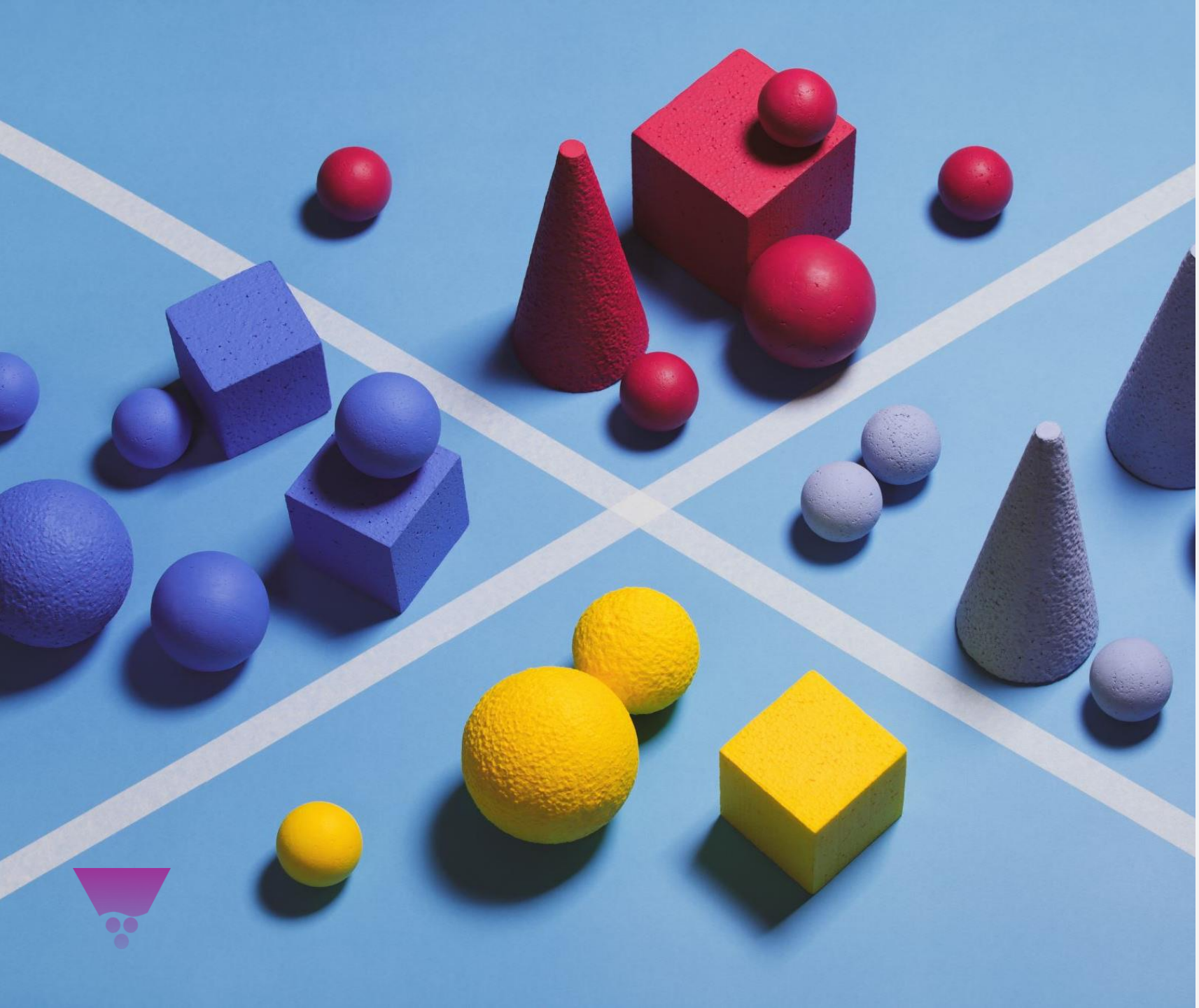
(from `man g++`)



COMPILING: BASIC

simplest.C :

```
int main()
{
  std::cout << "Hello\n" << std::endl;
}
```



OBJECTS

OOP

Objects

Brayan

Yanuar

Jawad

OOP

Attributes

Student No.
Modules
Library actions

Objects

Brayan

Yanuar

Jawad

OOP

Class

student

Attributes

Student No.
Modules
Library actions

Objects

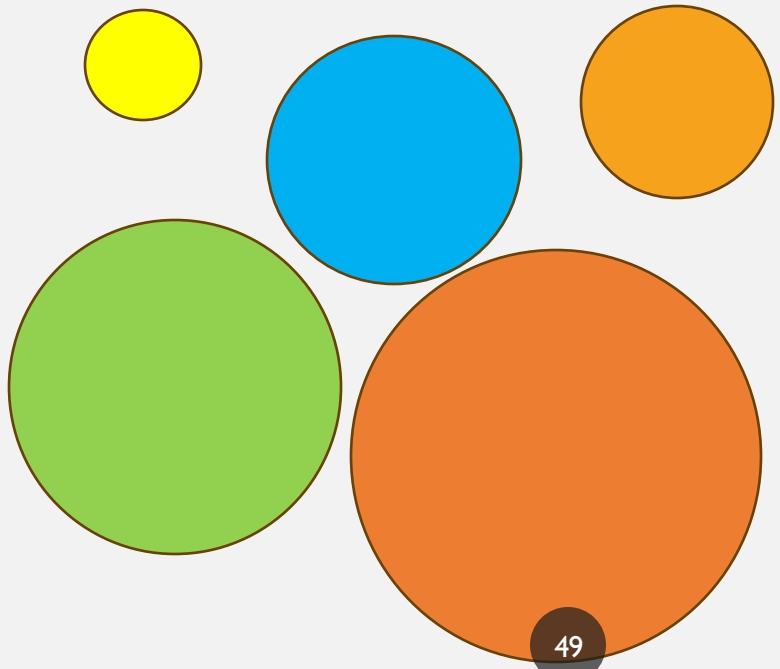
Brayan

Yanuar

Jawad

OOP

Objects



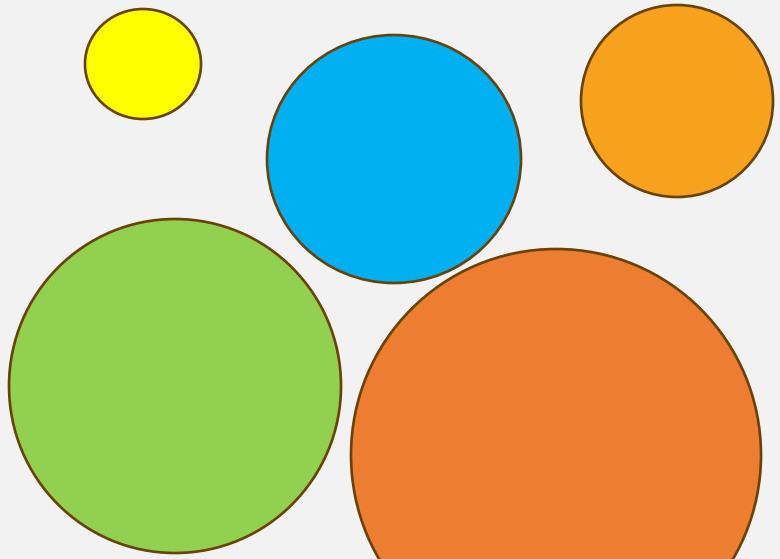
OOP

Attributes

Data
Diameter
Color

Methods
Area
Perimeter

Objects



OOP

Class

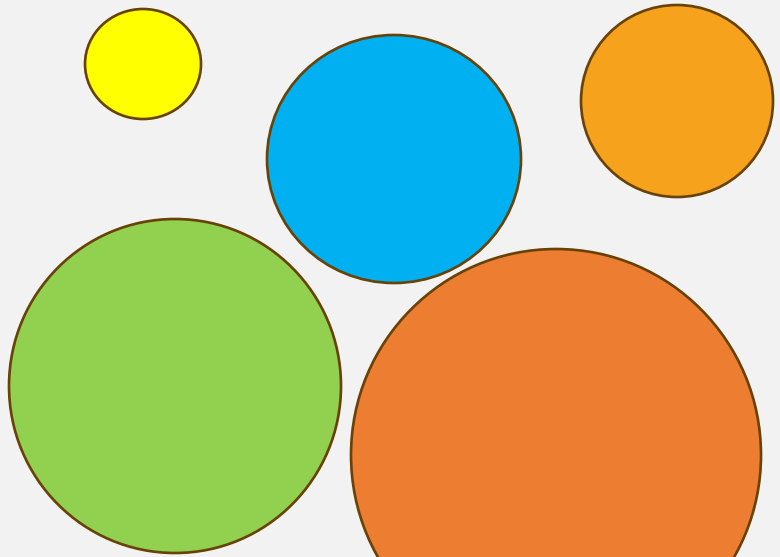
circle

Attributes

Data
Diameter
Color

Methods
Area
Perimeter

Objects



OOP

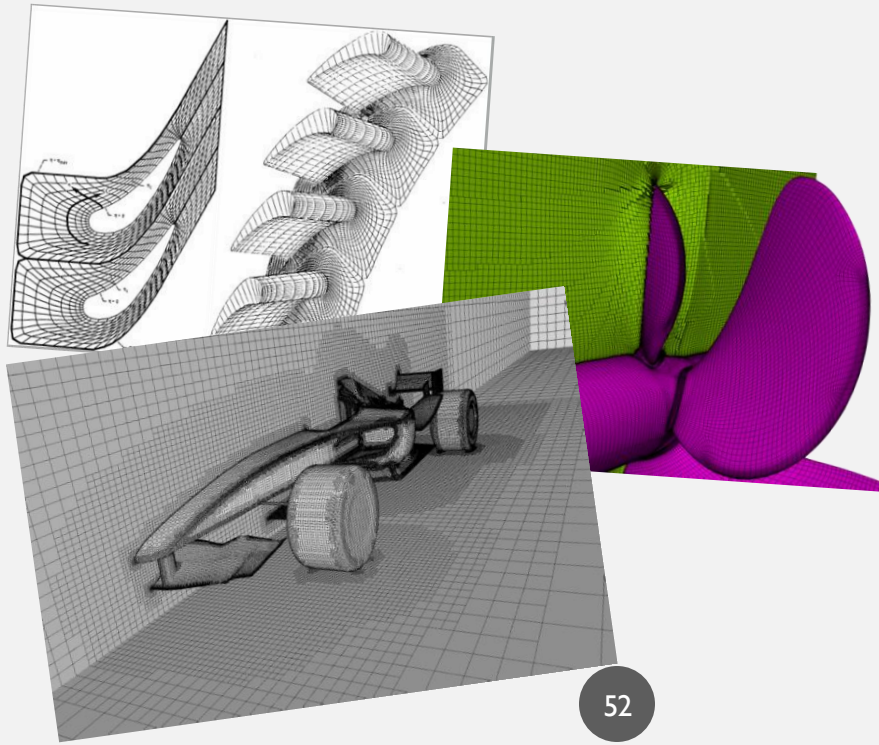
Class

fvMesh

Attributes

Data
points
faces
...
Methods
movePoints
addPatches

Objects



DRIVER

Calculate the area(*A*) of a circle with *diameter(D)=d* unit: *area.C*

```
int main()
{
    circle c1(1);
    std::cout << c1.area() << std::endl;

    return 0;
}
```

circle CLASS

Calculate the area(*A*) of a circle with *diameter(D)=d* unit: *area.C*

```
class circle
{
    // member data
    double d_;

    public:

    // member function(s)
    double area();

    // Constructor(s)
    circle(double);
};
```

circle CLASS

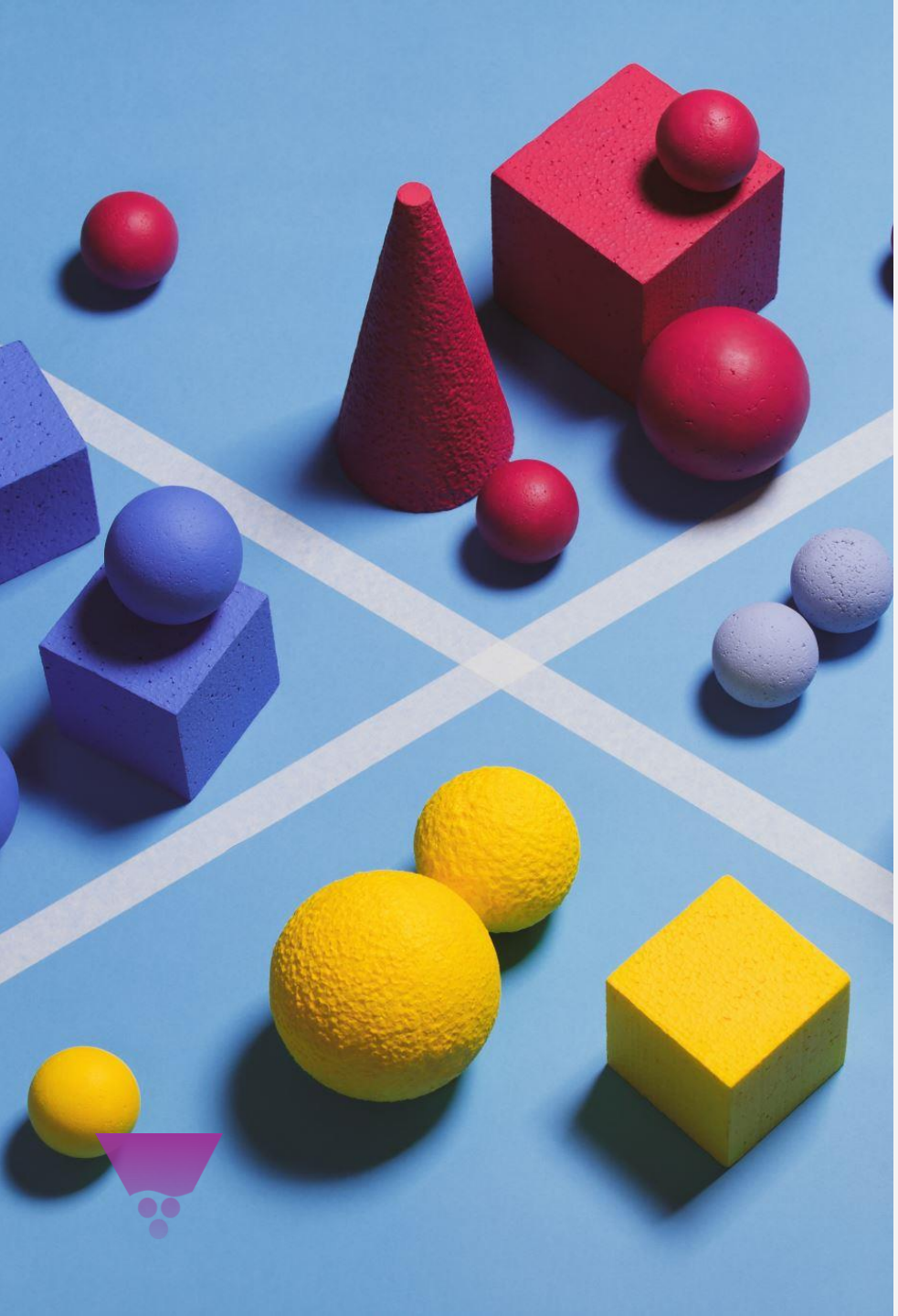
Calculate the area(*A*) of a circle with *diameter(D)=d* unit: *area.C*

```
circle::circle(double d):
d_(d)
{}

double circle::area()
{
   return 3.14* d_ * d_ / 4;
}
```

`circle` CLASS

Calculate the area(*A*) of a circle with *diameter(D)=d* unit:

Run



COMPILING: `wmake`

```
.
├── Make
│   ├── files
│   └── options
└── sayHello.C
```

Make/files:

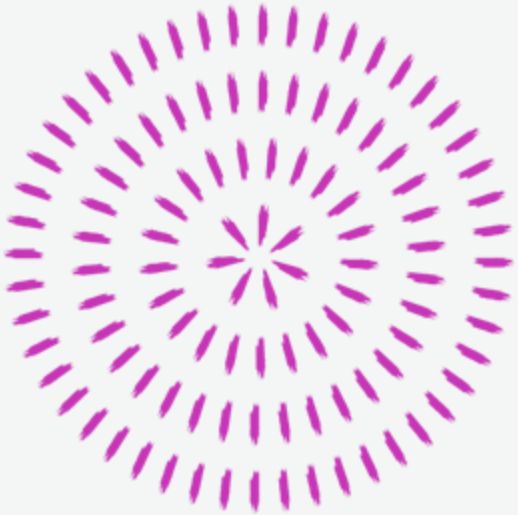
```
sayHello.C

EXE=$(FOAM_USER_APPBIN)/sayHello
```

QUIZZ

Create a class called **`rectangle`**;

Create a rectangle object;

Show its perimeter on the terminal.



OpenFOAM Directory Tree

LIBRARY VS APPLICATION

```
.
├── application
└── src
```

DIRECTORY TREE

LIBRARY VS APPLICATION

```
.
├── application
│   └── area
│       ├── area.C
│       └── Make
│           ├── files
│           └── options
└── src
    └── circle
        ├── circle.C
        ├── circle.H
        └── Make
            ├── files
            └── options
```

Application/ Executable

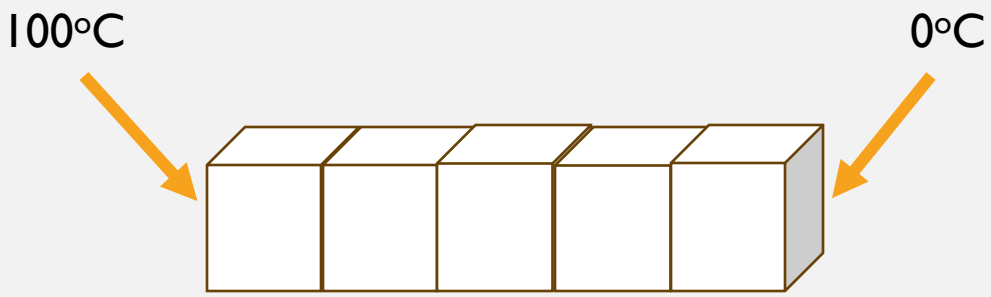
Library

A SOLVER'S SKELETON: MINIFOAM

main

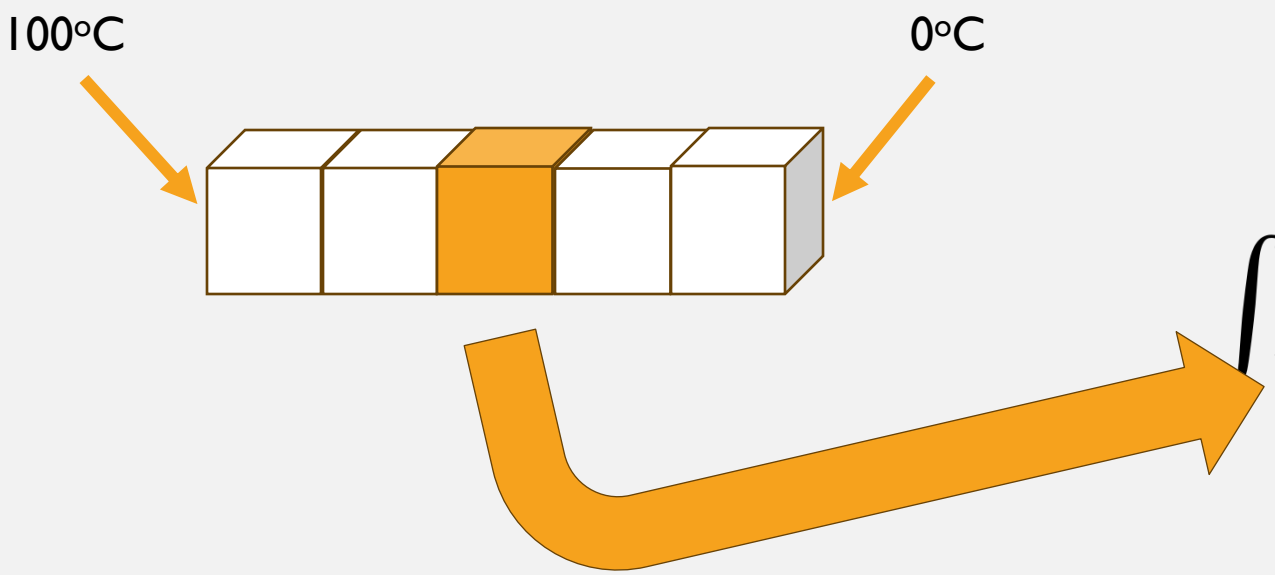
$$\frac{\partial^2 T}{\partial x^2} = 0$$

main



$$\frac{\partial^2 T}{\partial x^2} = 0$$

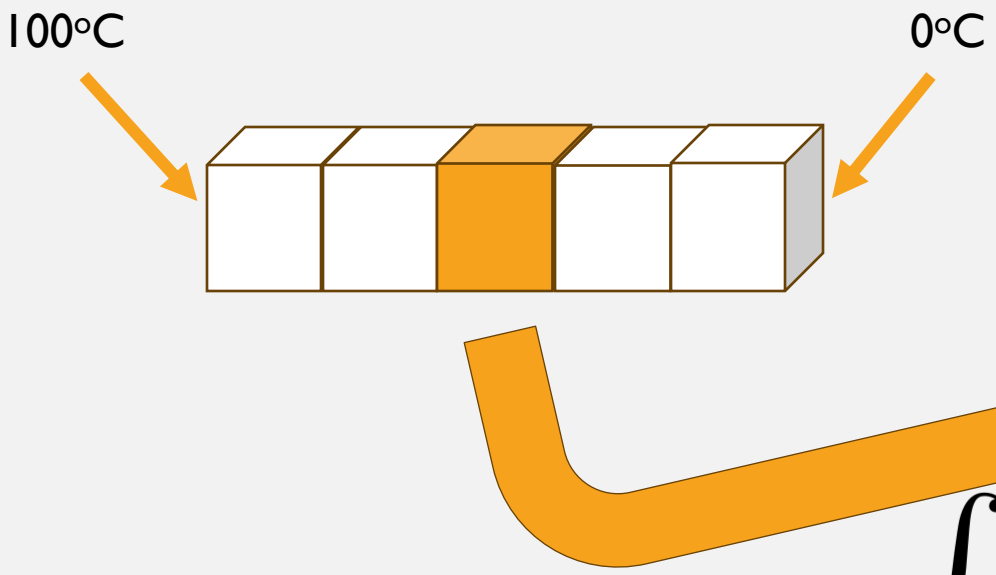
main



$$\frac{\partial^2 T}{\partial x^2} = 0$$

$$\int_{\mathcal{V}} \frac{\partial^2 T}{\partial x^2} \mathrm{d}V = 0$$

main

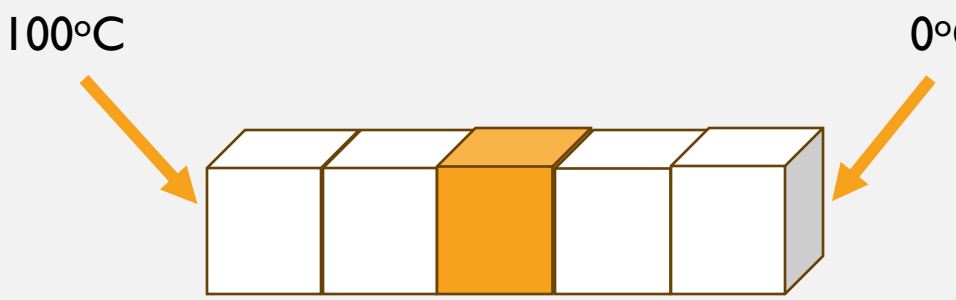


$$\frac{\partial^2 T}{\partial x^2} = 0$$

$$\int_{\mathcal{V}} \frac{\partial^2 T}{\partial x^2} \mathrm{d}V = 0$$

$$\int_{\mathcal{V}} \frac{\partial}{\partial x}\left(\frac{\partial T}{\partial x}\right) \mathrm{d}V = 0$$

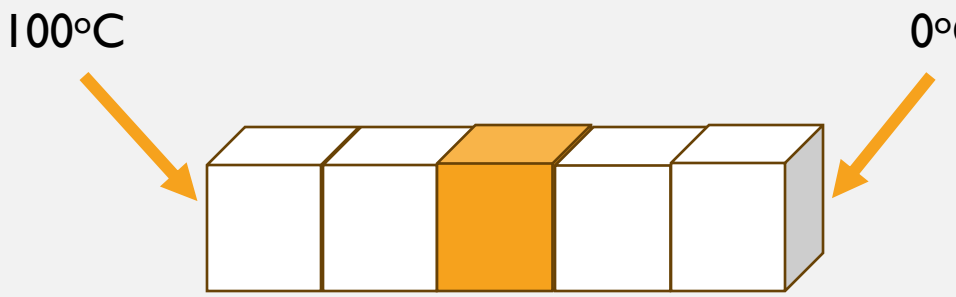
main



$$\int_{\mathscr{V}} \frac{\partial}{\partial x}\left(\frac{\partial T}{\partial x}\right)\mathrm{d}V = 0$$

$$\int_{\mathscr{F}} n_i \partial_i T \mathrm{d}S = 0$$

main

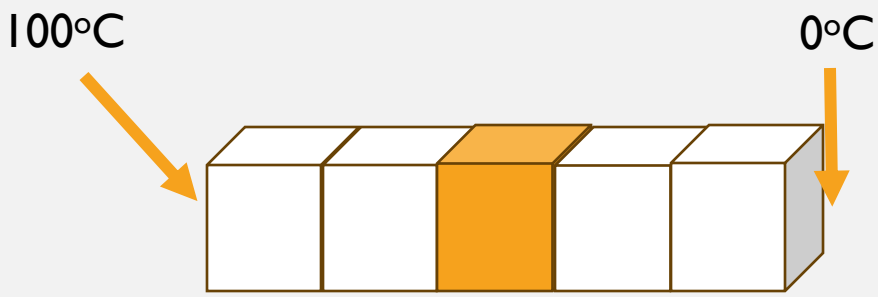


$$\int_{\mathscr{V}} \frac{\partial}{\partial x}\left(\frac{\partial T}{\partial x}\right)\mathrm{d}V = 0$$

$$\int_{\mathscr{F}} n_i \partial_i T \mathrm{d}S = 0$$

$$\int_{\mathscr{F}_\mathrm{E}} n_i \partial_i T \mathrm{d}S + \int_{\mathscr{F}_\mathrm{W}} n_i \partial_i T \mathrm{d}S = 0$$

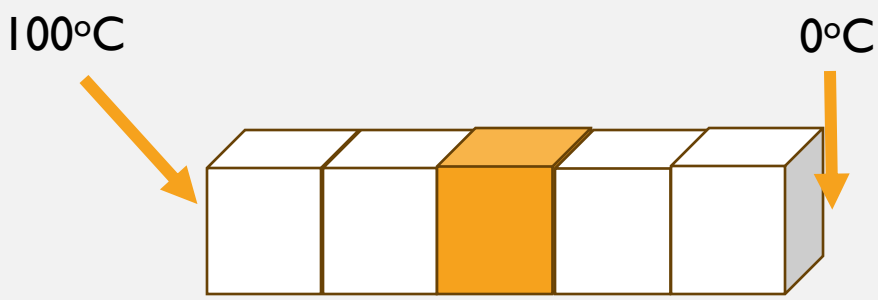
main



$$\int_{\mathscr{F}_\mathrm{E}} n_i \partial_i T \mathrm{d}S + \int_{\mathscr{F}_\mathrm{W}} n_i \partial_i T \mathrm{d}S = 0$$

$$(n_i \partial_i T S)_\mathrm{E} + (n_i \partial_i T S)_\mathrm{W} = 0$$

main

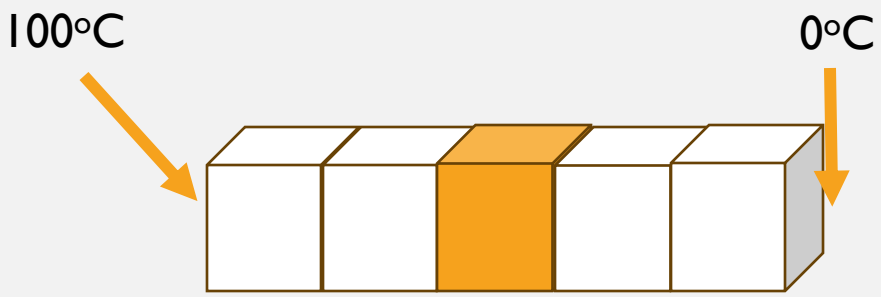


$$\int_{\mathscr{F}_\mathrm{E}} n_i \partial_i T \mathrm{d}S + \int_{\mathscr{F}_\mathrm{W}} n_i \partial_i T \mathrm{d}S = 0$$

$$(n_i \partial_i T S)_\mathrm{E} + (n_i \partial_i T S)_\mathrm{W} = 0$$

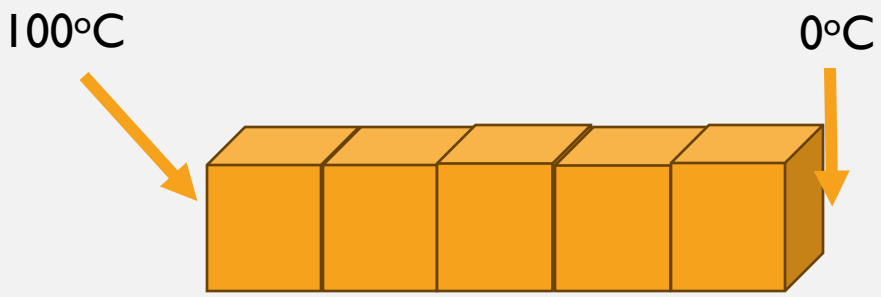
$$\frac{T_\mathrm{E} - T_\mathrm{P}}{\Delta x} S_\mathrm{E} + \frac{T_\mathrm{P} - T_\mathrm{W}}{\Delta x} S_\mathrm{W} = 0$$

main



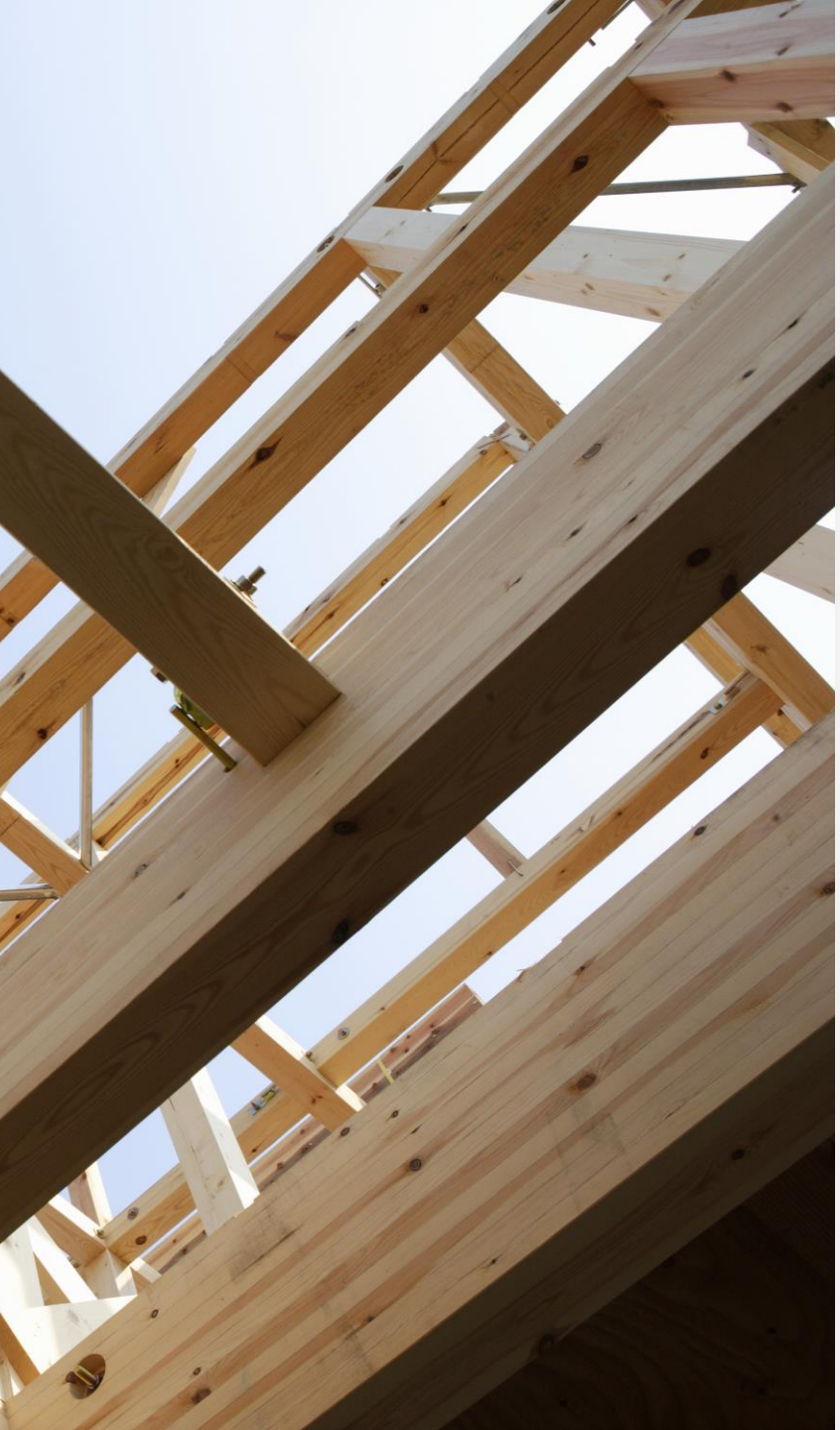
$$\frac{S}{\Delta x}T_{\mathrm{W}} + \frac{S}{\Delta x}T_{\mathrm{E}} - \frac{2S}{\Delta x}T_{\mathrm{P}} = 0$$

main



$$\frac{S}{\Delta x}T_{\mathrm{W}} + \frac{S}{\Delta x}T_{\mathrm{E}} - \frac{2S}{\Delta x}T_{\mathrm{P}} = 0$$

$$\begin{bmatrix} \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & a_{\mathrm{W}} & a_{\mathrm{P}} & a_{\mathrm{E}} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} T_0 \\ \vdots \\ T_{\mathrm{P}} \\ \vdots \\ T_{\mathrm{N}-1} \end{bmatrix} = \begin{bmatrix} b_0 \\ \vdots \\ b_{\mathrm{P}} \\ \vdots \\ b_{\mathrm{N}-1} \end{bmatrix}$$



`main`

Mesh
- Discretise the domain

PDE
- Discretise the PDE

Solution
- Solve the system

A SOLVER'S SKELETON: OPENFOAM

main

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

HEAT EQUATION

Field: `volScalarField T`

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

HEAT EQUATION

Field: `volScalarField T`

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

Time: `Time runTime`

HEAT EQUATION

Scalar: `dimensionedScalar alpha`

Field: `volScalarField T`

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

Time: `Time runTime`

HEAT EQUATION

Scalar: `dimensionedScalar alpha`

Field: `volScalarField T`

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

Time: `Time runTime`

Space: `fvMesh mesh`

HEAT EQUATION

Scalar: `dimensionedScalar alpha`

Field: `volScalarField T`

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

Time: `Time runTime`

Space: `fvMesh mesh`

Other required objects:

```
argList args
```

HEAT EQUATION

Scalar: `dimensionedScalar alpha`

Field: `volScalarField T`

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

Time: `Time runTime`

Space: `fvMesh mesh`

Other required objects:

```
argList args
IOdictionary dict
```

TIME AND SPACE

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

```
int main(int argc, char* argv[])
{
    argList args(argc, argv);
    Time runTime("controlDict", args);
    fvMesh mesh
    (
    IOobject
    (
        fvMesh::defaultRegion,
        runTime.timeName(),
        runTime,
        IOobject::MUST_READ));
...
```

OBJECTS

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

```
volScalarField T
(
    IOobject
    (
    "T",
    runTime.timeName(),
    mesh,
    IOobject::MUST_READ,
    IOobject::AUTO_WRITE
    ),
    mesh
);
```

OBJECTS

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

```
IOdictionary dict
( IOobject
    (
    "transportProperties",
    runTime.constant(), runTime,
    IOobject::MUST_READ, IOobject::NO_WRITE) );

dimensionedScalar k
(
"k",
pow(dimLength,2)/dimTime,
dict.lookup("k"));
```

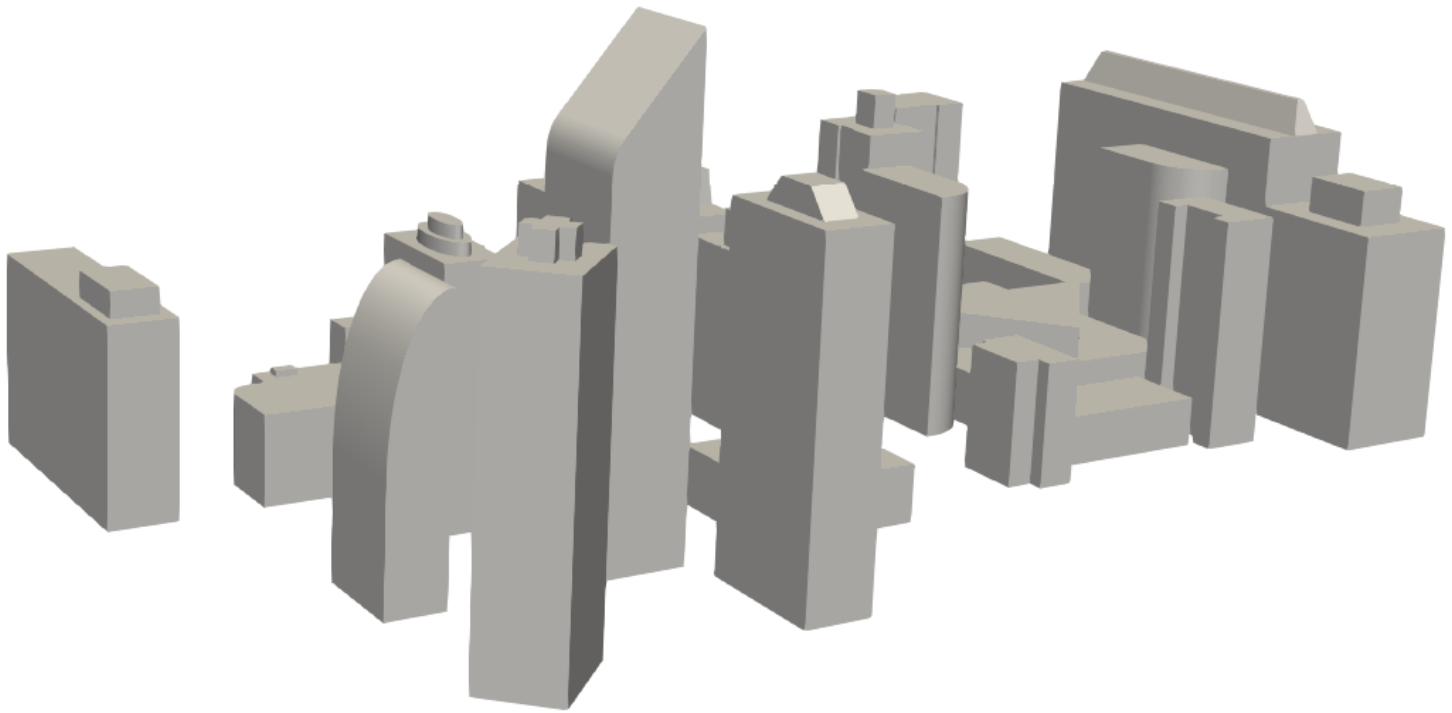
More dimensions in dimensionSets.H

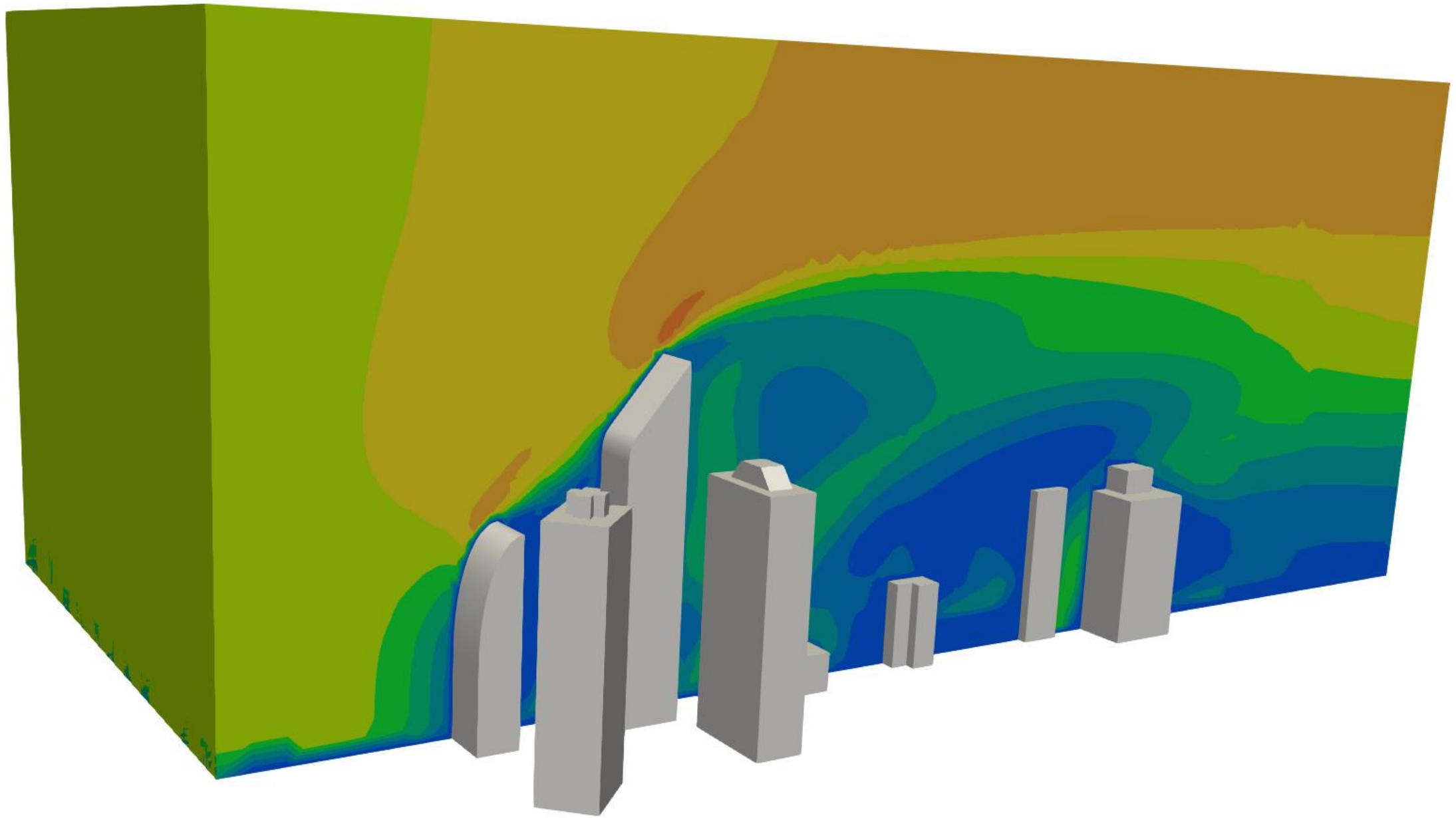
OBJECTS

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x_i x_i}$$

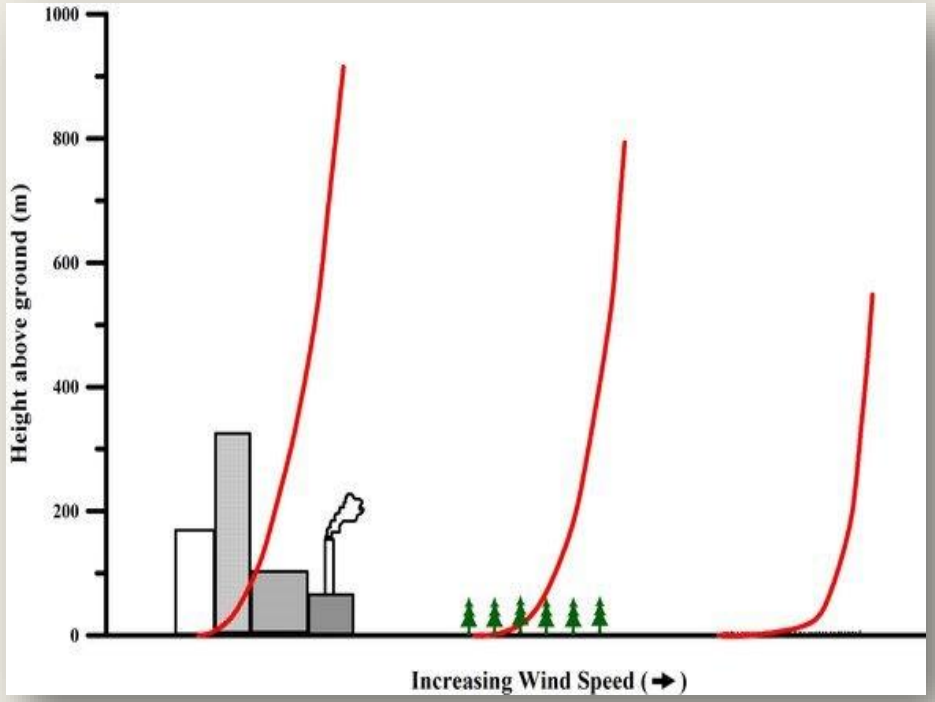
```
while (runTime.loop())
{
    fvScalarMatrix pEqn
    (
        fvm::ddt(p) -
        fvm::laplacian(k, p) == geometricZeroField()
    );
    pEqn.solve();

    runTime.write();
}
```

Boundary Conditions



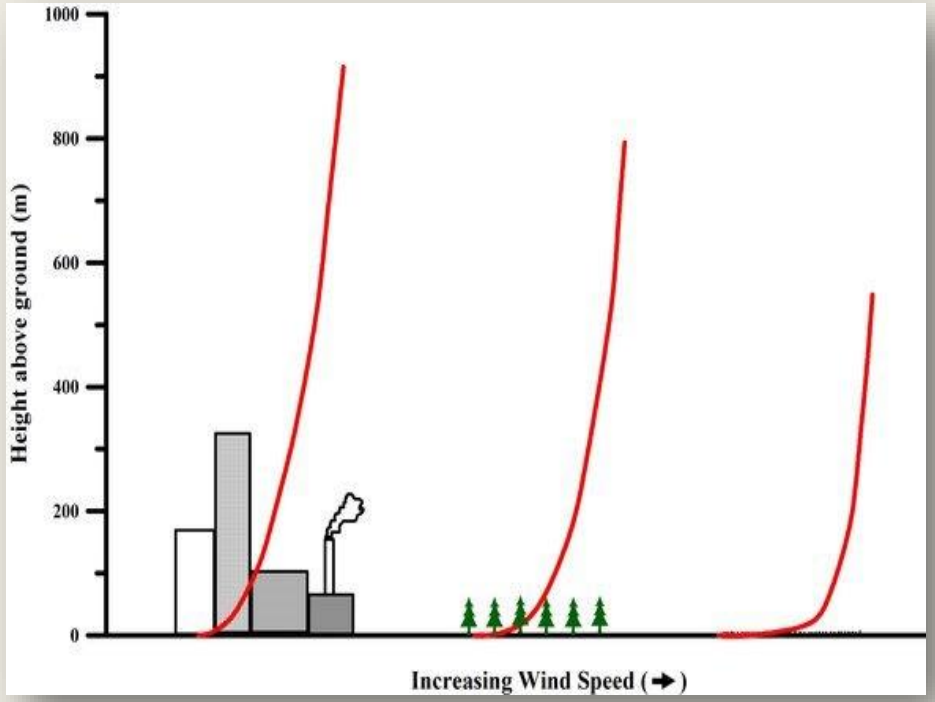


Wind Velocity Profile



From A surface roughness parameterization study near two proposed windfarm locations in Southern Ontario

Wind Velocity Profile



From A surface roughness parameterization study near two proposed windfarm locations in Southern Ontario

$$U_x = a\ln(bz + c)$$

Wind Velocity Profile

$$U_x = a\ln(bz + c)$$

```
foamNewBC -f -v wind
```

Wind Velocity Profile

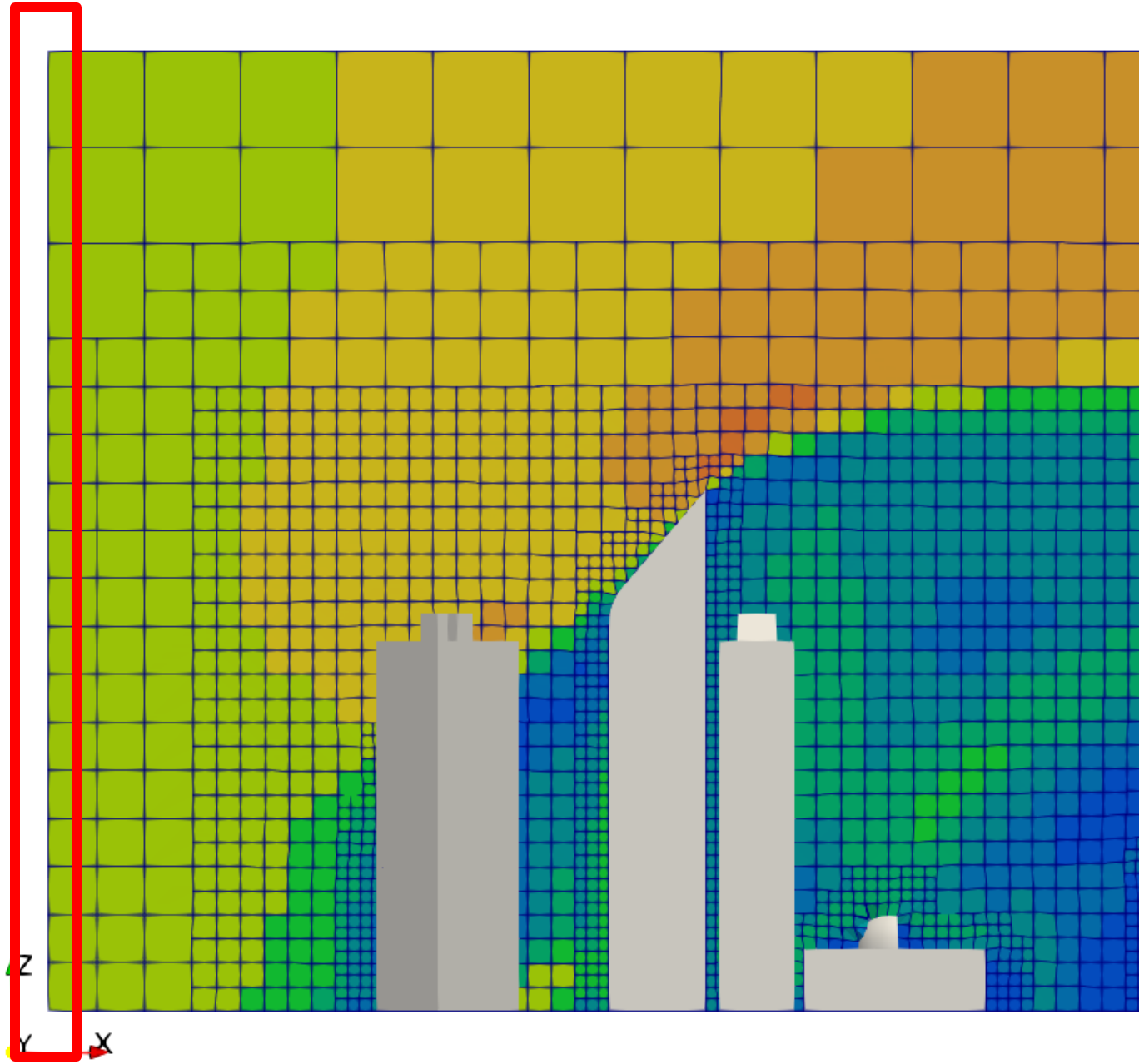
$$U_x = a\ln(bz + c)$$

```
foamNewBC -f -v wind
```

```
updateCoeffs
```

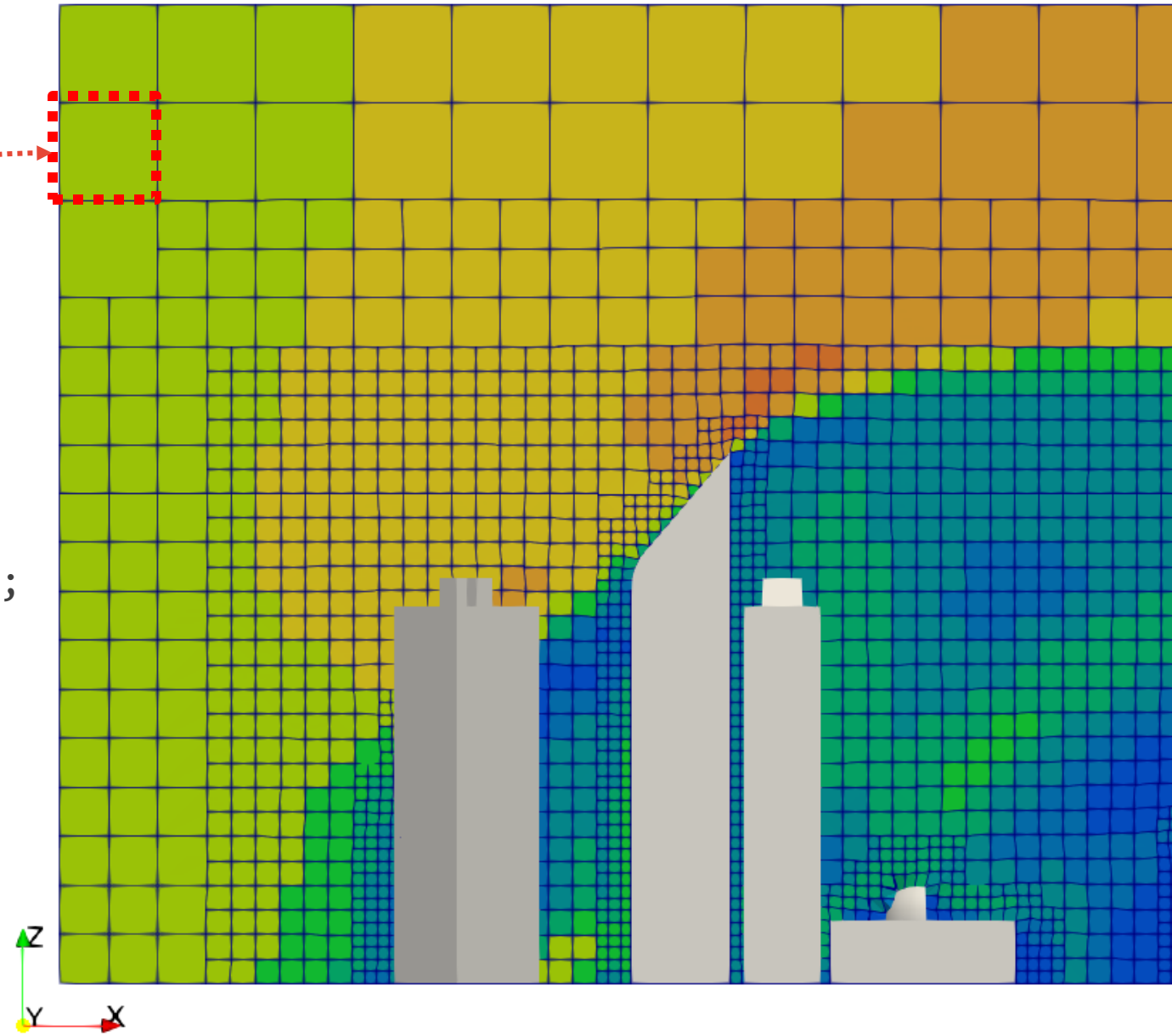
```
dict
```

$$U_x = a\ln(bz + c)$$



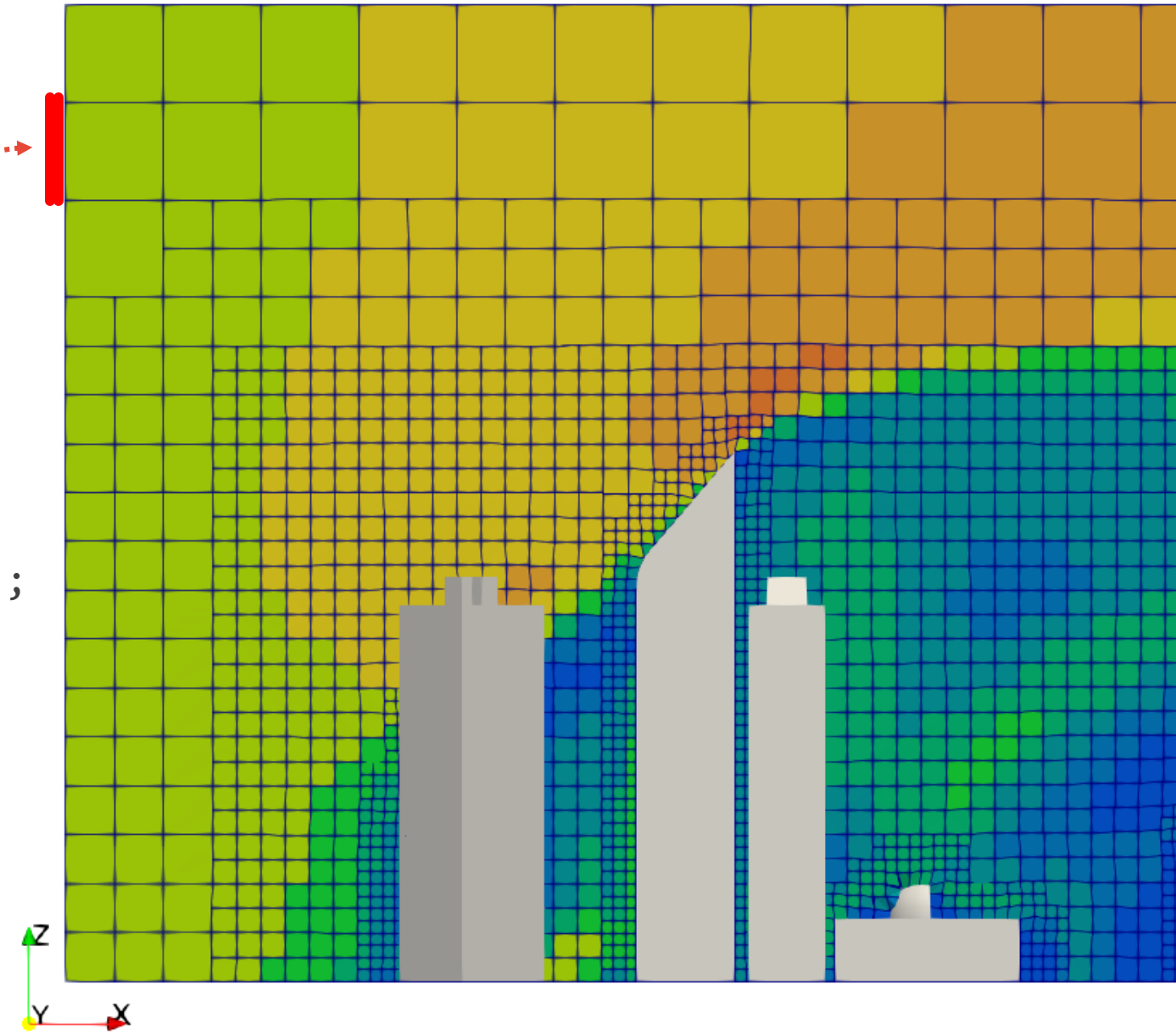
$$U_x = a\ln(bz + c)$$

```
U[i][0] =
    a * log(b*patch().Cf()[i][2] + c);
```



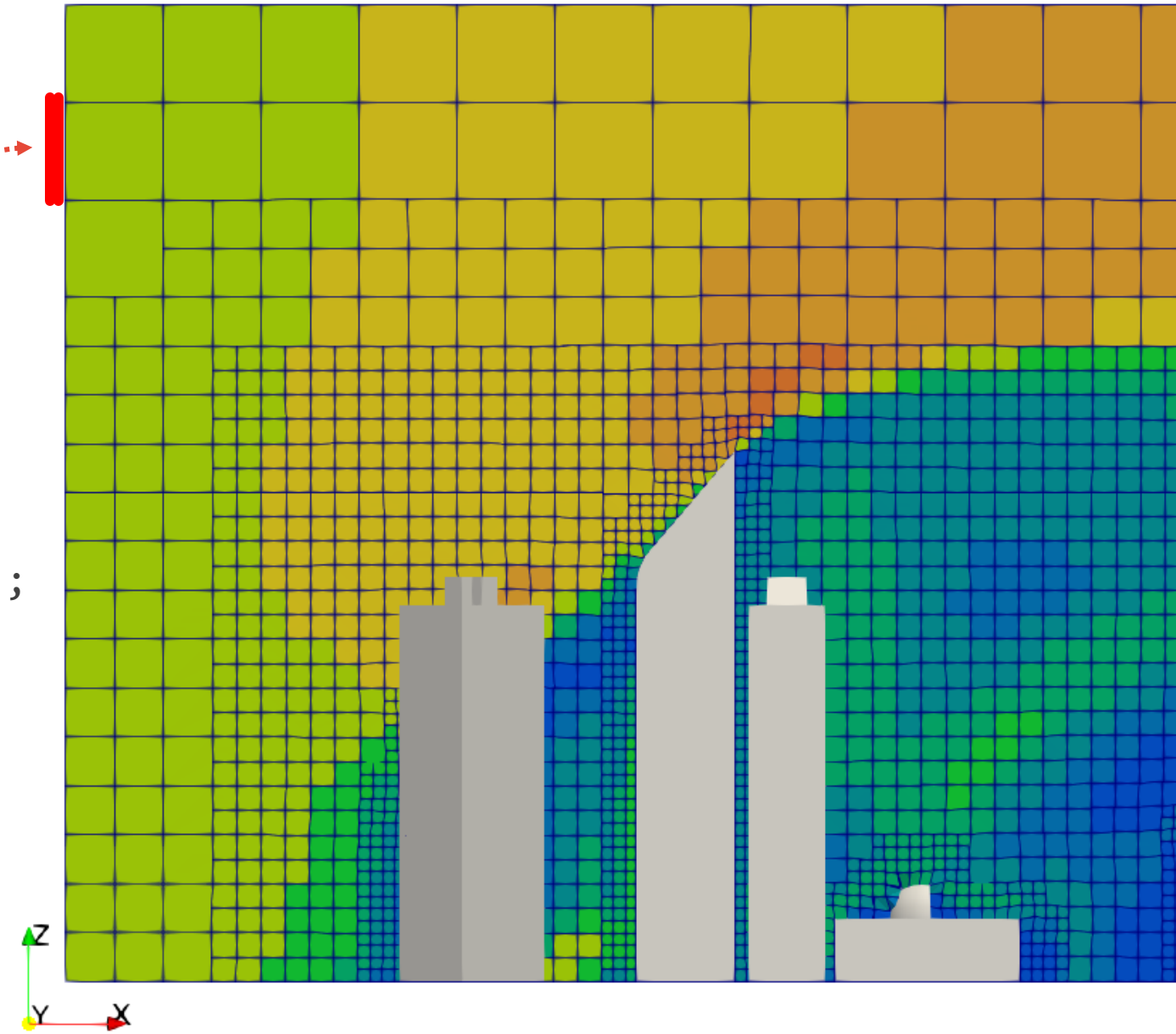
$$U_x = a\ln(bz + c)$$

```
U[i][0] =
    a * log(b*patch().Cf()[i][2] + c);
```



$$U_x = a\ln(bz + c)$$

```
U[i][0] =
    a * log(b*patch().Cf()[i][2] + c);
```

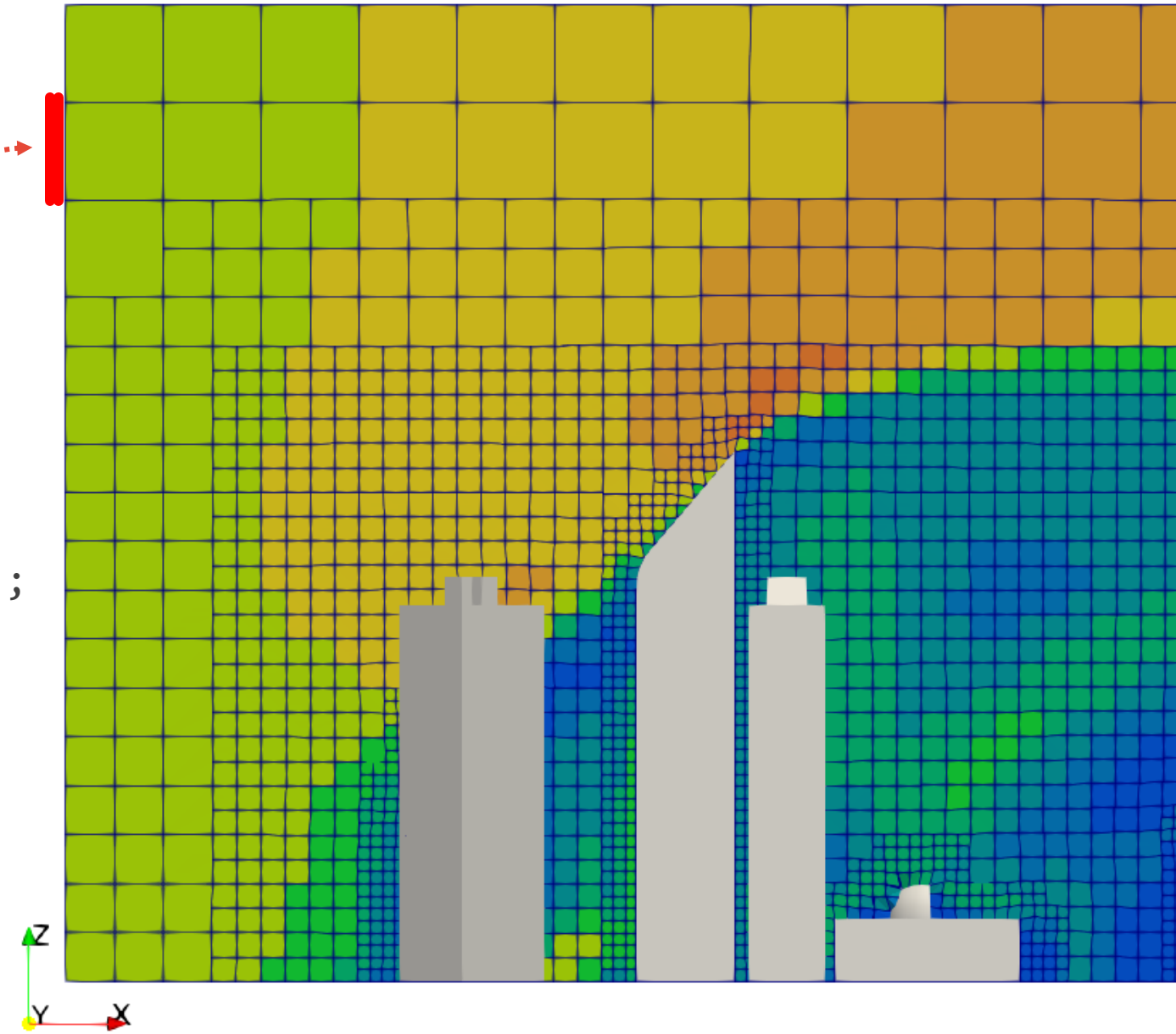


$$U_x = a\ln(bz + c)$$

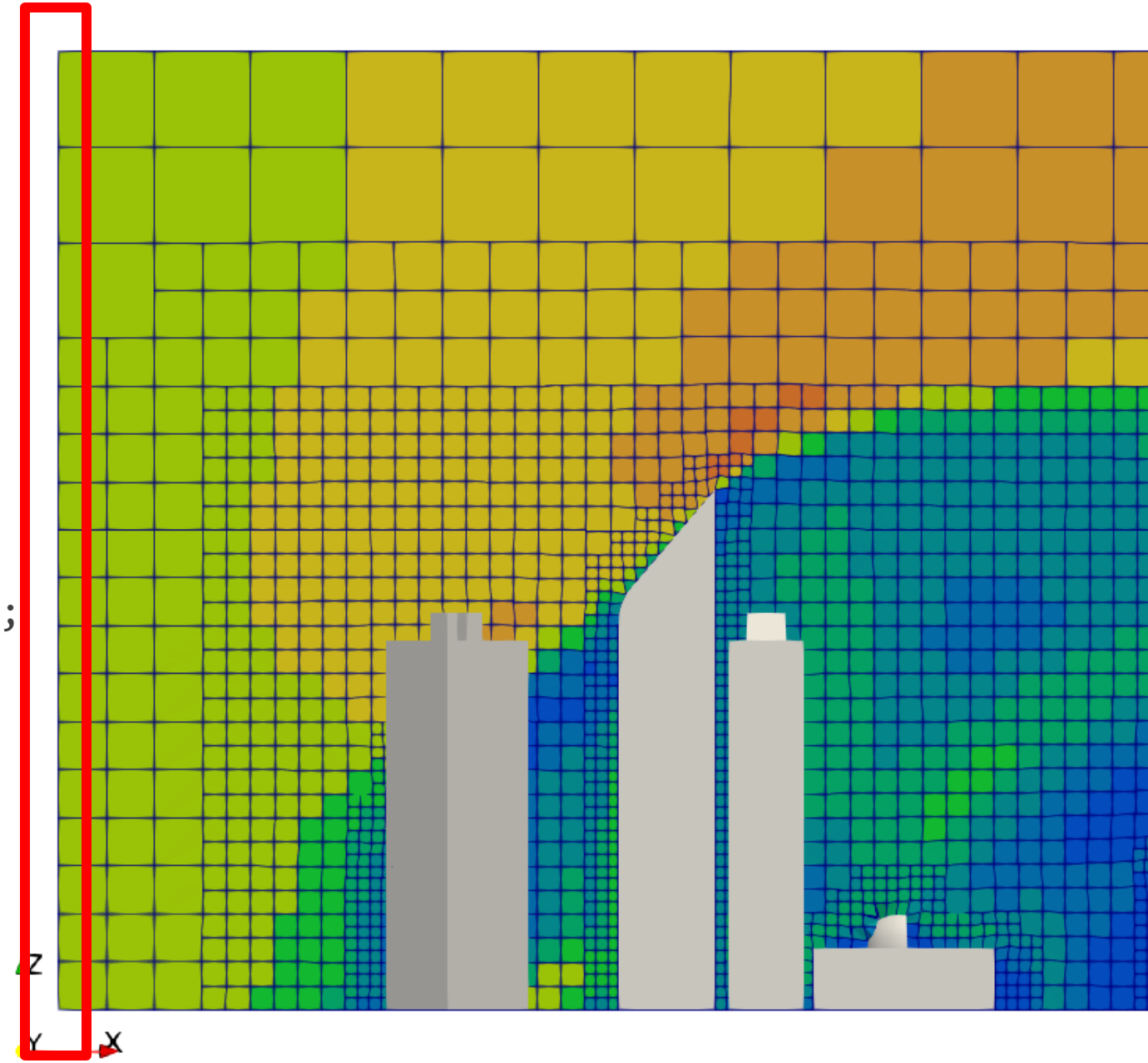
```
U[i][0] =
    a * log(b*patch().Cf()[i][2] + c);
```

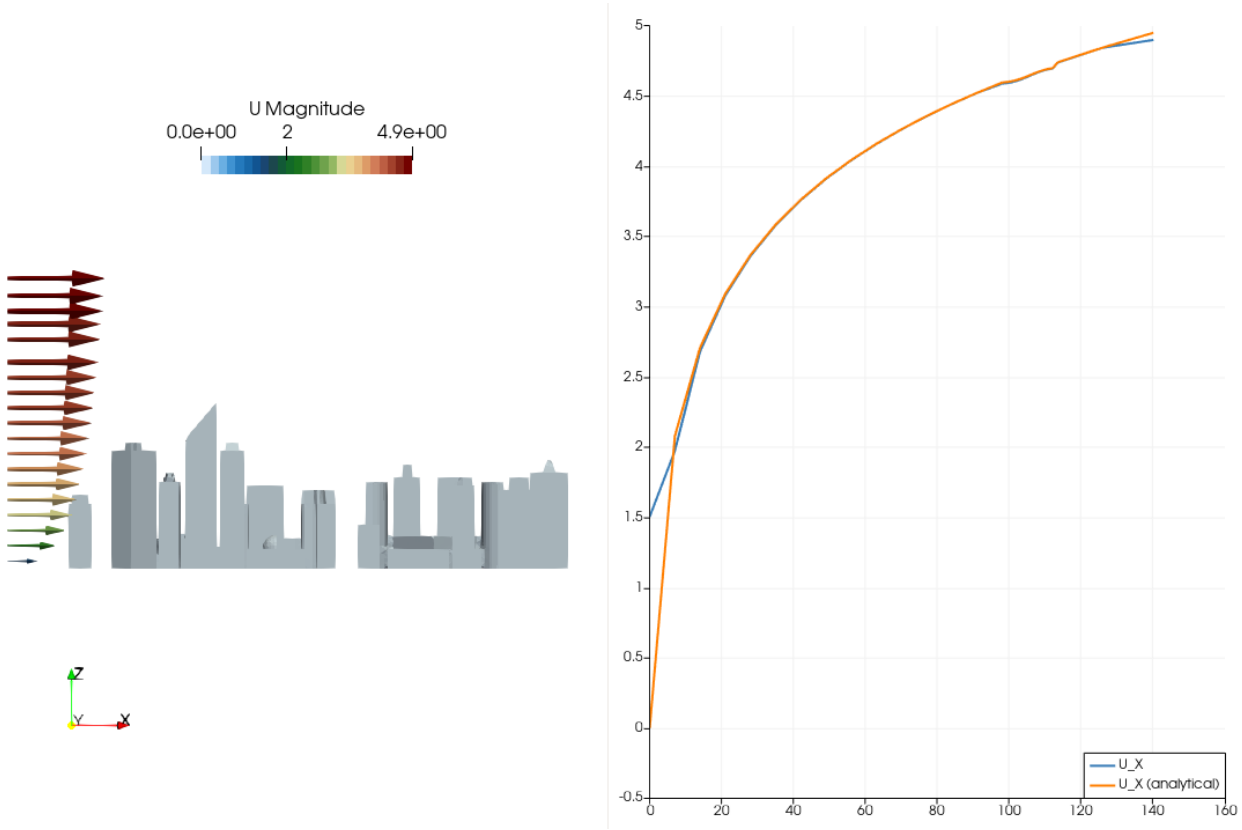
`patch()` returns an `fvPatch`.

See `fvPatch.H`



```
forAll (patch().Cf(), i)
{
U[i][0] =
    a * log(b*patch().Cf()[i][2] + c);
}
```





Quizz

- Compile the logarithmic BC;
- Run "windAroundBuildings" tutorial using the new BC at the inlet;

Hint: add the following to the *controlDict*:

```
libs    ("libwind.so");
```

Thank You!