TU Berlin, Scientific Computing, WS19/20
Homework assignment #2

Please return this assignment by Thursday, Jan. 16, 2020 Send it in form of a compressed (zip) file which unpacks into a subdirectory by e-mail to juergen.fuhrmann@wias-berlin.de. Please prefix file and subdirectory names with your last names, e.g. "MuellerNguyenHW02.zip". The file should contain a subdirectory with

- the commented Julia code in form of a module according to this template:

```
module MuellerNguyenHW02
function task1(;optional_parameters)
end
function task2(;optional_parameters)
end
....
end
```

- a pdf describing your answers. Hint: you can use the package Literate.jl to create this pdf from your source (e.g. via Jupyter notebook or via markdown + pandoc)

For the following tasks, you can start with the code provided in the jupyter notebook of lecture 17.

1 Mesh generation

Write Julia code to define a polygon approximating a circle of radius 1 with the possibility to choose the number of approximation points. Use this polygon as input to the triangulate() method provided by the Julia package Triangulate.jl and generate discretization meshes of the disk Ω described by the circle with maximum triangle areas of $0.1 \cdot 4^{-n}$ for $n = 1 \ldots 7$.

Plot the coarsest grids, and provide a table showing some characteristics of the different grids: the refinement level, the smallest edge length, the number of triangles and the number of vertices.

If possible, characterize the error of approximation of the disc by the polygonal domain.

2 Finite element simulation

For the disk Ω given above, find an exact solution to the homogeneous Dirichlet problem

$$\begin{aligned} -\Delta u &= 1 \quad \text{in} \quad \Omega \\ u &= 0 \quad \text{on} \quad \partial\Omega \end{aligned}$$

Using the P1 finite element method on the discretization meshes created in task 1, calculate approximate solutions and compare them to the exact solution obtained. Provide graphs of the solution and report the accuracy of approximation on the different meshes in a table and a graph. Discuss the results.

If possible, compare the time for matrix assembly for a standard Julia SparseMatrixCSC and an ExtendableSparseMatrix from the Julia package ExtendableSparse.jl.