Übungsblatt 7: Wärmeleitung in 2D (Dirichlet RBn)

Aufgabe 1: Dirichlet-Randbedingungen

1.1 Implementieren Sie die Funktion

```
function [K, r] = applyDirichletBCs(dofs, K, r)
```

mit der das lineare Gleichungssystem so modifiziert wird, dass die Lösungen für die Freiheitsgrade in `dofs` gleich null sind.

Zur Kontrolle:

```
[K, r] = applyDirichletBCs([2, 4], magic(5), (1:5)');
```

```
K = 5x5                                    r = 5x1
    17    24     1     8    15                1
     0     1     0     0     0                0
     4     6    13    20    22                3
     0     0     0     1     0                0
    11    18    25     2     9                5
```

1.2 Erzeugen Sie mit Gmsh ein FE-Netz für ein kreisförmiges Gebiet mit dem Radius $r = 2.2\,\mathrm{m}$. Die Elementgröße soll $h = 0.2\,\mathrm{m}$ betragen. Es trägt zur Übersichtlichkeit bei, wenn Sie die Gmsh-Dateien in einen Unterordner `gmsh` legen.

Zur Kontrolle: Sie sollten rund 500 Knoten und 1000 Elemente erhalten.

1.3 Berechnen Sie in der Datei **'heat_plate.mlx'** die Näherungslösung der Temperaturverteilung für die Kreisplatte (Parameter λ und w frei gewählt). Vergleichen Sie die maximale Temperatur mit der exakten Lösung

$$w_{\max} = \frac{w \cdot r^2}{4 \cdot \lambda}.$$

Tipp: Die Funktion `assembleKr` von Übungsblatt 4 sollte auch für das Wärmeleitungsproblem funktionieren. Mit den Attributen `Nn` und `Ne` der `Mesh`-Klasse lässt sich der Code ggf. noch übersichtlicher schreiben.

1.4 Erstellen Sie die Funktion

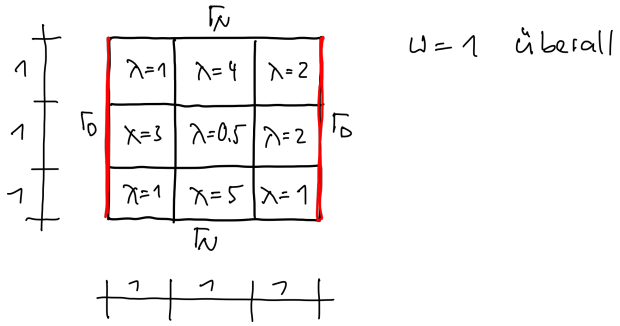
```
function plotNodalSolution(mesh, values)
```

um auf dem Netz `mesh` die Näherungslösung in `values` zu plotten. Verwenden Sie die Matlab-Funktion `patch` (In der Dokumentation gibt es ein Beispiel, das Sie (fast) 1:1 übernehmen können).

Plotten Sie damit die Temperaturverteilung der aus Aufgabe 1.4.

Zusatzaufgabe: Mithilfe der Funktion der Temperaturverteilung (Notizen zur Vorlesung) können Sie die exakte Lösung in den Knotenpunkten berechnen und damit den Diskretisierungsfehler plotten.

1.5 Berechnen Sie die Wärmeverteilung für das dargestellte Wärmeleitungsproblem.



Sie müssen hierfür die Funktion `assembleKr` dahingehend anpassen, dass die Funktionen `keFunc` und `reFunc` für alle Elemente aller physikalischen Gruppen aufgerufen werden, falls diese definiert sind.

Tipp: Mit `isa(a, 'function_handle')` können Sie überprüfen, ob die Variable `a` ein Funktionshandle ist.

1.6 Mit `K = sparse(K)` können Sie die Matrix K so umwandeln, dass nur die von null verschiedenen Einträge gespeichert werden. Vergleichen Sie die Zeit, die zum Lösen des LGS benötigt wird für die Ausgangsmatrix und die konvertierte Matrix.

Tipp: Mit `tic` und `toc` können Sie in Matlab die benötigte Rechenzeit messen.