# Your first metric graph

## Creating a metric graph

Let us first create a combinatorial star graph `G` with 5 vertices and 4 edges using Graphs.jl

```
using Graphs
G = star_graph(5)
```

```
{5, 4} undirected simple Int64 graph
```

In order to extend `G` to an equilateral metric graph, we define the edge length `ℓ`

```
ℓ = pi + pi/2
```

```
4.71238898038469
```

`G` can now be represented as metric graph `Γ` by applying the function `metric_graph`

```
Γ = metric_graph(G, ℓ)
```

```
{n=5,m=4,ℓ=4.71238898038469} equilateral metric graph
```

For a small example like the star graph, vertex coordinates can be assigned that will later allow to visualize `Γ` in 3d.

```
coords = [[0,0],
          [ℓ,0],
          [-ℓ,0],
          [0,ℓ],
          [0,-ℓ]]
```

```
5-element Vector{Vector{Float64}}:
 [0.0, 0.0]
 [4.71238898038469, 0.0]
 [-4.71238898038469, 0.0]
 [0.0, 4.71238898038469]
 [0.0, -4.71238898038469]
```

The function `metric_graph` takes the optional input `vertex_coords` to specify the vertex coordinates.

```
Γ = metric_graph(G, ℓ, vertex_coords = coords)
```

```
{n=5,m=4,ℓ=4.71238898038469} equilateral metric graph
```

We may now plot `Γ` using `plot_graph_3d`

```
{n=5,m=4,ℓ=1} equilateral metric graph
```

**Note**

The previous example graph can be assembled using the constructor `metric_star_graph` `metric_star_graph(ℓ = pi + pi/2)`. Several other example graphs are implemented..

## Functions on metric graphs

A function `u` on a metric graph is represented by a vector of functions `u_e`, specifying `u` on each edge `e`.

```
u = [ x -> -3*sin(x),
    x -> sin(x),
    x -> sin(x),
    x -> sin(x)
    ]
```

```
4-element Vector{Function}:
 #1 (generic function with 1 method)
 #2 (generic function with 1 method)
 #3 (generic function with 1 method)
 #4 (generic function with 1 method)
```

If vertex coordinates are assigned to `Γ`, a function can be plotted on `Γ` with

```
4-element Vector{Function}:
 #1 (generic function with 1 method)
 #2 (generic function with 1 method)
 #3 (generic function with 1 method)
 #4 (generic function with 1 method)
```