# Tips!

**This is a "Hands-on" course. Long experience has shown that the only way to really learn the FEM is to write codes for yourself.**

**Here are some quick tips to make this process a little less painful!**

**(And don't forget that the instructors are here for you)**

# Programming

- **Start "top down", test often**
  - **Set up the framework of your program as quickly as possible, "without the guts"**
  - **Make sure you can always run your script to completion, and do so often**
  - **Generate a plot/image as quickly as possible, even if it's wrong at first**
- **Save working states. ***
- **Change one thing at a time. ***
- **Test as you go.**
  - **you WILL make errors. Work in such a way as to make them obvious**
  - **know what you want the answer to look like, before running the code**
- **Work out small, numerical tests on paper and check them against your code**

**MATLAB-specific tips**

- **The help system is very useful (e.g.** `help magic`**)**
- **Start your script with** `clear; close all; clc;`
- **While working, allocate arrays with** `NaN()` **(not** `zeros()`**)**
- **Use the debugger**
- **Many more tips from an old course on MATLAB (start at lecture 5)**
  - https://bitbucket.org/psanan/introduction-to-matlab-and-mathematica

* Version control systems (e.g. git) make these easier

# FEM

- **Remember the Procedure, before getting lost in the details**
- **Remember that more complex codes have a similar form to simple ones.**
    - **Define problem parameters**
    - **Define mesh**
    - **Define time domain**
    - **Determine numberings:**
        - **element node numbers -> point number (and type)**
        - **boundary node numbers**
        - **point number (and type) to equation number**
    - **Initialize global matrices**
    - **Element Loop (if time-independent, otherwise inside time loop)**
        - **Compute local matrices and vectors**
        - **Add to global matrices anf vectors**
    - **Time Loop**
        - **Apply boundary conditions**
        - **Solve system**

- **Break symmetries to test: use different numbers of elements in each direction**
- **Break symmetries to test: use irregular grids**