TopEdu

PROJECT 8 INSTRUCTIONS

Why a ruler twists so easily

Twist a ruler, then a pencil. Why such a difference? Work out the rule, then test it with your hands.

START HERE

Set up in ten minutes

This project comes with one short Python file, `08_beam_torsion.py`. It solves the project's equation with simple methods you can read line by line, checks itself against an answer that is known exactly, and ends with a list of first experiments. The research is what you do after that.

1. **Install Python 3** from python.org (on a Mac it may already be there: type `python3 --version` in Terminal).
2. **Download the starter** from the project's page, or all ten as a zip from topedu.club/topedu/research/starter-code/.
3. **Install the two libraries** in Terminal (Mac) or PowerShell (Windows):

   ```
   python3 -m pip install numpy matplotlib
   ```

   On Windows, type `py` where this guide says `python3`.
4. **Run it:**

   ```
   python3 08_beam_torsion.py --show
   ```

   The CHECK line must say PASS. `--show` opens the plots; without it they are saved in the `output` folder.

No computer of your own? Google Colab works in a browser for free: open a new notebook, upload the file with the folder icon on the left, and run `!python 08_beam_torsion.py` in a cell.

The rules that make it research

- **Predict before you run.** Write your guess, and why, in your log first. A wrong prediction you can explain is worth more than a right one you cannot.
- **Keep a log.** Date, what you changed, what you expected, what happened. Your paper is written from it.
- **Check the grid.** Run your key result again with twice the points. If the answer moves, report by how much.
- **Never edit the CHECK to make it pass.** A FAIL after your change is information: undo, confirm PASS, then change one thing at a time.
- **Own every line.** Your mentor explains; you write and run. Name everyone and everything that helped, including this starter.

The full guide, with all ten projects and five numerical ideas the starters use, is at topedu.club/topedu/research/starter-code/.

PROJECT 8 · MECHANICS AND STRUCTURES

Why a ruler twists so easily

Twist a ruler, then a pencil. Why such a difference? Work out the rule, then test it with your hands.

FILE	MATH	YOU NEED
`08_beam_torsion.py`	Geometry, Algebra II; numerical integration	Foam or wood bars and some weights

The question

Twisting a bar is governed by a surface shaped like a soap film over its cross-section. How does stiffness depend on the cross-section's shape, and how well does the simple thin-strip rule work?

Objectives

Weeks 1 to 3	**Learn to trust the model.** Run 08_beam_torsion.py until its CHECK line says PASS, and explain every line of it to your mentor. Then reproduce a known answer: A square bar's torsion constant is 0.1406 times the side to the fourth power; a long thin strip approaches 1/3. Checkpoint: you reproduce a known answer and report how close you got.
Weeks 4 and 5	**Ask your own question.** Narrow the question above to one sentence that could turn out to be wrong, and write your prediction before you run anything. The first experiments below are a place to start. Checkpoint: a one-page plan with your question and prediction.
Weeks 6 to 9	**Run it, then measure the world.** Solve for rectangles of many aspect ratios, compute the twist stiffness from each solution, then twist real bars and compare. Then step away from the screen: hang known weights from an arm attached to foam or wooden bars and measure the twist angle. Checkpoint: one chart comparing your model with reality.
Weeks 10 to 12	**Explain the gap, and share it.** Work out why your model and your measurements disagree, test at least one explanation, and write it up yourself. Say which numbers came from the starter and which from the lab. Keep the honest limit in view: only rectangles are supported, and only small twists stay in the elastic range. Checkpoint: a final paper, your code and a ten-minute talk.

The model in the starter

When you twist a bar, its stiffness comes from a "stress function" phi over the cross-section, shaped like a soap film blown up over a frame the shape of the section: -laplacian(phi) = 2 inside, phi = 0 on the edge. The torsion constant is J = 2 x (volume under the film). For an a by b rectangle (b the short side) write J = k a b^3. A square has k = 0.1406; a long thin strip approaches k = 1/3. Twist per length = torque / (G J), with G the shear modulus.

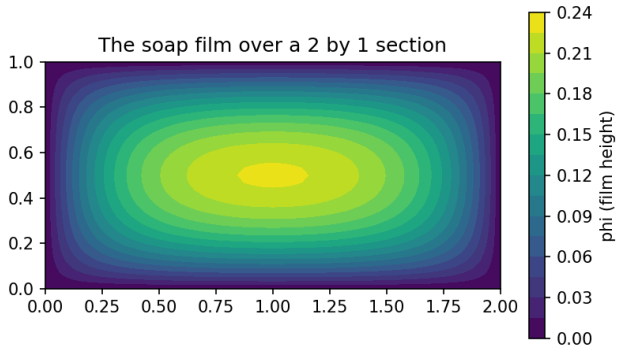
What the starter does

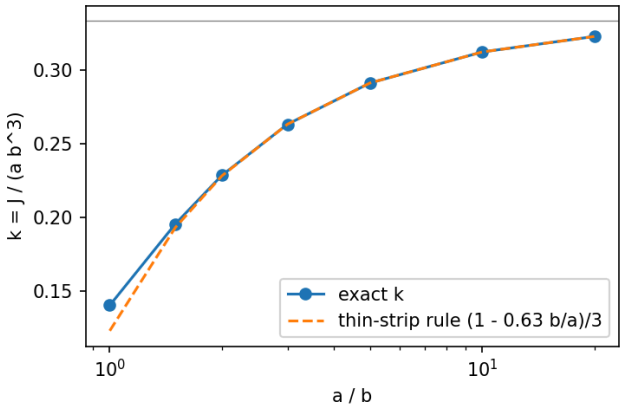
- Stage 1: run this file. It solves for phi on three rectangles and checks k against the exact series. The CHECK line should say PASS.
- Stage 2: twist real bars, measure their stiffness, and test the rule.

What you should see

Run `python3 08_beam_torsion.py`. This is the output from the run that built this guide. The numbers that depend on your computer's speed may differ; the CHECK line must say PASS.

```
1:1 bar: k = 0.1403, series 0.1406, error 0.20%
2:1 bar: k = 0.2283, series 0.2287, error 0.18%
4:1 bar: k = 0.2802, series 0.2808, error 0.20%
CHECK torsion factor against the series -> PASS (limit 1%)
```





Your first experiments

From the YOUR TURN list at the bottom of the file. Write your prediction in your log *before* you run each one.

1. A ruler is about 30 by 3 mm; a pencil about 7 by 7 mm. Predict which is stiffer in twist, and by how much, using J = k a b^3. Is the answer what your hands tell you?
2. Where does the thin-strip rule (the dashed line) go wrong by more than 5%? Read it from the plot, then check.
3. Experiment: clamp one end of a foam or wood bar, hang a weight from an arm at the other end, and measure the twist angle. Stiffness = torque x length / angle should be G J. Do bars of different shapes agree on G?
4. Honest limit: only rectangles, and only small twists. Try a hollow square by setting phi to a constant on an inner hole, and say what changes.

The data

Settings in the online lab

Setting	Range	Starts at
Long side	1 to 10 cm	2
Short side	1 to 4 cm	1

Checked on the research solver

Solver (Poisson, 2D)	1:1 beta 0.1404; 2:1 beta 0.2284; 4:1 beta 0.2804
Known answer	0.1406; 0.2287; 0.2808
Error	0.16% (limit 1.00%): passed
A first finding	beta = 0.1406 for the square, as published

The real world

Hang known weights from an arm attached to foam or wooden bars and measure the twist angle.

The honest limit

Only rectangles are supported, and only small twists stay in the elastic range.

How the full project checks itself: A square bar's torsion constant is 0.1406 times the side to the fourth power; a long thin strip approaches 1/3. The online lab runs the same project on the research solver (Poisson, 2D).

REFERENCE

When something goes wrong

You see	Try this
`No module named numpy`	Run the install line again with the same Python you use to run the file: `python3 -m pip install -r requirements.txt`
Numbers become nan or huge	A time step is past its speed limit, usually after you made the grid finer. Each solver works out its step from the grid; check you did not fix it by hand.
`did not settle`	An iterative solver needs more sweeps for a finer grid. Raise its limit, or keep the grid smaller while you explore.
A CHECK line says FAIL after your change	Good: your change moved the model away from the case with a known answer. Undo it, confirm PASS, then change one thing at a time.
No window opens	Add `--show`, or open the PNG files in the `output` folder.
A recording gives strange notes	Record in a quiet room, save as .wav (not .m4a), trim silence at the start, and set the note you expect in the file.

How the starters relate to the online lab

The online lab, for members, runs each project on the research solver (AutoSBPSAT): high-order SBP-SAT methods, the full 3D elasticity for project 9, and frames of a time series. The starters use plain second-order methods instead, so every line is short enough to read. Both are checked against the same kind of exact answers. Use the starter to understand and to explore quickly; use the lab for results you will quote, and say in your paper which one produced each number.

How these starters were checked

Every number in "What you should see" came from running the starter while this guide was built, on Python 3.9.23, numpy 1.24.4, matplotlib 3.5.2. `check_all.py` also feeds each optional data reader (the soil-station CSV, the ruler CSV and two .wav recordings) a made-up file with a known answer, and confirms the answer comes back. They have not yet been used by a student. The twelve-week plan and the time estimates in it are a design, to be revised after the first group: tell your mentor what was unclear.