

Effective Programming Practices for Economists

Debugging

Avoiding debugging

Janoš Gabler and Hans-Martin von Gaudecker

How editors avoid debugging

- Code highlighting shows differences between
 - variables / functions
 - built-in statements
 - string literals
 - comments
- Syntax checkers flag
 - spelling errors
 - undefined variables
 - syntax errors

How unit tests avoid debugging

- If your function is thoroughly tested:
 - It will produce correct outputs given valid inputs
 - You can skip it while debugging (at least initially)
- Careful
 - Having one test per function is usually not enough
 - Need at least make sure to cover all lines
 - Also think about difficult edge cases and test them

How error handling avoids debugging

- If your function does thorough error handling
 - It will complain when called with invalid inputs
 - You can locate easily where things go wrong
- Careful
 - You do not want to add error handling to private helper functions
 - To be useful, the error messages need to be good

How readability avoids debugging

- If you write readable modular code
 - All functions have a clear purpose, so it is easy to see whether they do what they should
 - Variable names are informative about content
 - You can easily simulate in your head what the code does
- Problem:
 - This is hard and takes practice!