**Anti patterns:**

1. God class – Class that does too much
2. Magnet class – Ambiguous name that attracts
3. Poorly named methods/classes - Name isn’t descriptive / describes only part of the functionality, or overly broad
4. Sawtooth code – Many random indentations due to various levels of nesting
5. Arrow code – Heavily indented in arrow shape
6. Magic numbers – Refactor to well named constant
7. Constructor over injection – Honor SRP
8. Bastard injection – Overloading constructor to provide testability
9. Negative conditionals/names
10. Coding by accident – Trying negations/casts/etc until it seems to work
11. Exceptions as control flow
12. Leaky abstractions – concerns from one layer leaking into another
13. Side Effects (calling method causes other unrelated consequences)
14. Spaghetti code – Multiple concerns intermixed in a given method/class
15. Dead code – Code that is never called.
16. “Stringly typed” – Using strings for conditionals instead of strong types
17. Hard Coded dynamic information or configuration

**Code smells:**

1. Out parameters
2. Zombie Code – commented out code
3. Long Parameter list
4. Conditional complexity – multiple conditionals
5. And/or in name – Sign that method does more than one thing
6. High cyclomatic complexity – High number of discrete paths through the code
7. Similar “shape” – Just squint your eyes
8. Abbreviated identifiers
9. Primitive Obsession – Loose variables that are logically related
10. Long Methods/Large Classes
11. Comments
12. Duplicated/similar switch statements
13. Service locator
14. Complex data heavy logic (consider DB driven method)