Code Review Checklist

Style & Appearance:

- ❑ Is the code tabbed well?
- ❑ Are curly braces used in a consistent style?
- ❑ Identifiers:
    - ❑ Are names meaningful?
    - ❑ Are variable names in camelNotation (lowercase initial)?
    - ❑ Are class names in CamelNotation (capital initial)?
    - ❑ Are any constants in ALLCAPS?
- ❑ Comments:
    - ❑ Are comments used to explain difficult segments of code?
    - ❑ Are comments provided at the top to identify author and date?
    - ❑ Are comments brief and to the point?
    - ❑ Is every comment necessary?
- ❑ Advanced:
    - ❑ Is there any redundant or duplicate code?
    - ❑ Is the code as modular as possible? Are tasks broken down into methods when possible?
    - ❑ Can any global variables be replaced (using parameters instead)?
    - ❑ Do loops have a set length and correct termination conditions?
    - ❑ Can any of the code be replaced with library functions?
    - ❑ Can any logging or debugging code be removed?
        - ❑ Example: extra System.out.println lines

Functionality:

- ❑ Does the code perform correctly?
    - ❑ Regular cases
    - ❑ Corner cases
    - ❑ Invalid cases
- ❑ Are directions to user clear and explicit?
    - ❑ When asking for input, is the type and purpose of each input clear?
    - ❑ When providing output, is the meaning of values (including units as applicable) explained

Some questions were taken from
https://blog.fogcreek.com/increase-defect-detection-with-our-code-review-checklist-example/