My top 10 list of helpful design and programming rules

Design:

1. Represent ideas (concepts) directly in code, for example, as a function, a class, or an enumeration.
2. Represent independent ideas separately in code, for example, avoid mutual dependencies among classes.
3. Use libraries, especially the standard library, rather than trying to build everything from scratch.
4. Represent independent ideas independently in code.
5. Keep information local (e.g., avoid global variables, minimize the use of pointers).
6. Aim for your code to be both elegant and efficient.
7. Dont overabstract.
8. Represent relationships among ideas directly in code, for example, through parameterization or a class hierarchy.
9. Prefer solutions that can be statically checked.

Programming:

1. Use constructors to establish invariants.
2. Use constructor/destructor pairs to simplify resource management (RAII).
3. Avoid naked `new` and `delete`.
4. Use containers and algorithms rather than built-in arrays and ad hoc code.
5. Prefer standard-library facilities to locally developed code.
6. Use exceptions, rather than error codes, to report errors that cannot be handled locally.
7. Use move semantics to avoid copying large objects.
8. Use `unique_ptr` to reference objects of polymorphic type.
9. Use `shared_ptr` to reference shared objects, that is, objects without a single owner that is responsible for their destruction.
10. Use templates to maintain static type safety (eliminate casts) and avoid unnecessary use of class hierarchies.