Definitions

Design: A specification of an object, manifested by some agent, intended to accomplish goals, in a particular environment, using a set of primitive components, satisfying a set of requirements, subject to some constraints.

- specification
- goals
- environment
- components
- requirements
- constraints

General Principles

There are no fixed rules to sound design – instead there are some general principles.

Here are some:

- Modularisation
- Abstraction
- Information Hiding
- Coupling & Cohesion
  - note that these also come up inside the others

Modularisation

A module should have only a single responsibility.

If a module has more than one responsibility, then changes to one may inhibit the module's ability to meet other responsibilities.

This is also known as separation of concerns.

Modules are containers for functionality or responsibilities.

Advantages of modularisation:

1. Handles system complexity by introducing well-defined boundaries/interfaces.
2. Supports software reuse.
3. Improves system maintainability.
   - if a module breaks (e.g. a fatal security flaw is found), you can replace it with a new one
4. Supports replacing modules with alternatives.
5. Supports task decomposition (teamwork).

Information Hiding

A module should expose its functionality through an interface, but hide its implementation details.

This:

- increases maintainability
- increases reuse
- supports task decomposition

Abstraction

The process of forgetting information, so that things that are different can be treated as if they were the same.

There are 2 common mechanisms:

Parameterisation

Java generics and C++ templates are examples of this.

Specification

E.g. inheritance, interface implementation

- making something more specific?

Why?

- helps to prevent duplication of code through reusability
- duplication is bad because if you need to change something you have to change it in multiple places

Coupling & Cohesion

Coupling is a measure of strength of association between modules.

Strong coupling means many connections between modules, which reduces maintainability.

Low coupling supports:

- program comprehension
- maintainability

- task decomposition

Cohesion measures the degree of connectivity between elements within a module.

Low cohesion means that a module contains many unrelated elements. This violates the principle of modularity (single responsibility) and the principle of abstraction.

[...]