In software design, the goal is to create **modules** that are **easier to understand, develop, test, and maintain**. This is achieved through the principles of **cohesion**, **coupling**, and **functional independence**.

**1. Functional Independence (3 marks)**

* **Definition:**  
  Functional independence refers to designing modules so that each performs **a single, well-defined task** with **minimal interaction** with other modules.
* **Why it matters:**
  + Simplifies **testing and debugging**.
  + Enhances **reusability**.
  + Reduces the impact of changes.
* **Achieved by:**  
  Maximising **cohesion** and minimising **coupling**.

**2. Cohesion (5 marks)**

* **Definition:**  
  Cohesion is the **degree to which the elements within a module** belong together or perform a **single task**.
* **High cohesion is desirable** because it leads to:
  + **Simpler code**
  + Easier **maintenance**
  + Better **reuse** of modules
* **Types of cohesion (ordered from worst to best):**
  + **Coincidental Cohesion** – Random functions in one module (worst)
  + **Logical Cohesion** – Similar category of tasks (e.g., all I/O operations)
  + **Temporal Cohesion** – Functions executed at the same time (e.g., init tasks)
  + **Procedural Cohesion** – Related steps in a sequence
  + **Communicational Cohesion** – Functions use the same data
  + **Sequential Cohesion** – Output of one function is input to another
  + **Functional Cohesion** – All functions work toward a single task (best)

**3. Coupling (5 marks)**

* **Definition:**  
  Coupling is the **degree of interdependence between modules** — how much one module relies on another.
* **Low (loose) coupling is desirable**, because:
  + Modules can be changed or reused independently.
  + It reduces the risk of a change in one module **breaking** another.
* **Types of coupling (from highest to lowest – worst to best):**
  + **Content Coupling** – One module accesses the internal data of another (worst)
  + **Common Coupling** – Multiple modules share global data
  + **Control Coupling** – One module controls another by passing control flags
  + **Stamp Coupling** – Modules share data structures, not needed entirely
  + **Data Coupling** – Modules share only required data (best)