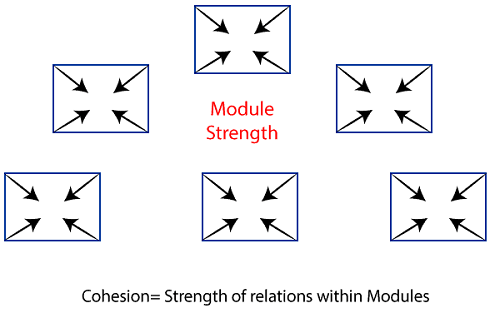
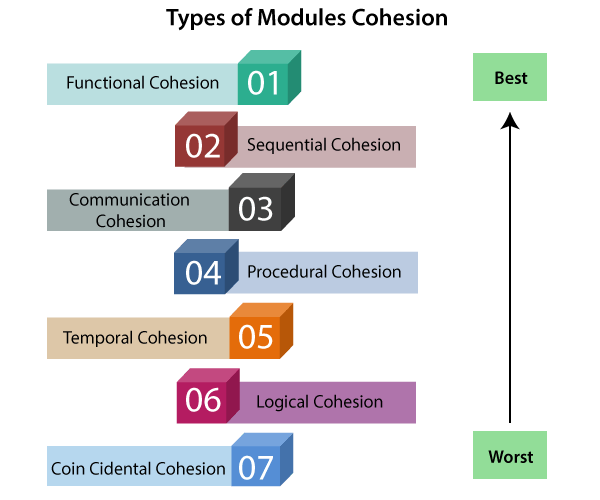
Q1. Module Cohesion

In computer programming, cohesion defines to the degree to which the elements of a module belong together. Thus, cohesion measures the strength of relationships between pieces of functionality within a given module. For example, in highly cohesive systems, functionality is strongly related.

Cohesion is an **ordinal** type of measurement and is generally described as "high cohesion" or "low cohesion."



Types of Modules Cohesion

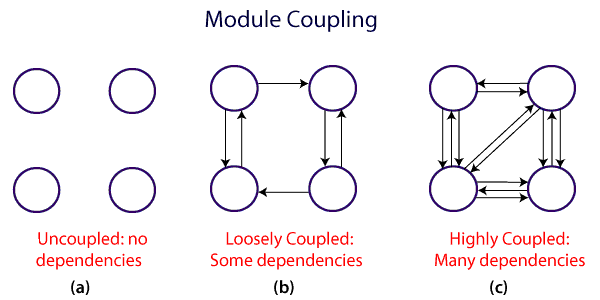


* **Functional Cohesion:** Every essential element for a single computation is contained in the component. A functional cohesion performs the task and functions. It is an ideal situation.
* **Sequential Cohesion:** An element outputs some data that becomes the input for other element, i.e., data flow between the parts. It occurs naturally in functional programming languages.
* **Communicational Cohesion:** Two elements operate on the same input data or contribute towards the same output data. Example- update record int the database and send it to the printer.
* **Procedural Cohesion:** Elements of procedural cohesion ensure the order of execution. Actions are still weakly connected and unlikely to be reusable. Ex- calculate student GPA, print student record, calculate cumulative GPA, print cumulative GPA.
* **Temporal Cohesion:** The elements are related by their timing involved. A module connected with temporal cohesion all the tasks must be executed in the same time-span. This cohesion contains the code for initializing all the parts of the system. Lots of different activities occur, all at in it time.
* **Logical Cohesion:** The elements are logically related and not functionally. Ex- A component reads inputs from tape, disk, and network. All the code for these functions is in the same component. Operations are related, but the functions are significantly different.
* **Coincidental Cohesion:** The elements are not related(unrelated). The elements have no conceptual relationship other than location in source code. It is accidental and the worst form of cohesion. Ex- print next line and reverse the characters of a string in a single component.

## Module Coupling

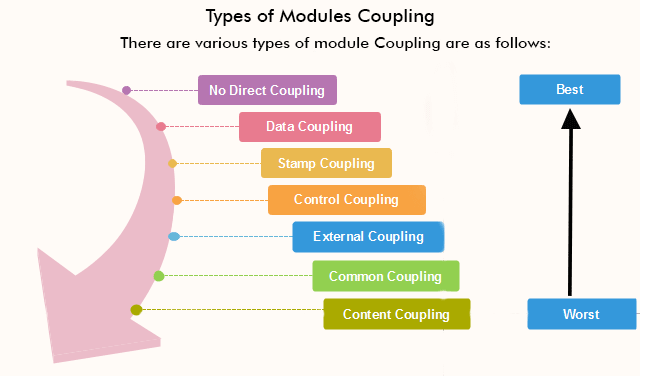
In software engineering, the coupling is the degree of interdependence between software modules. Two modules that are tightly coupled are strongly dependent on each other. However, two modules that are loosely coupled are not dependent on each other. **Uncoupled modules** have no interdependence at all within them.

**The various types of coupling techniques are shown in fig:**



A good design is the one that has low coupling. Coupling is measured by the number of relations between the modules. That is, the coupling increases as the number of calls between modules increase or the amount of shared data is large. Thus, it can be said that a design with high coupling will have more errors.

### Types of Module Coupling



1. **No Direct Coupling –** Completely uncoupled components are not systems. ν Systems are made of interacting components.
2. **Data Coupling –**  
   Data coupling simply means the coupling of data i.e. interaction between data when they are passed through parameters using or when modules share data through parameters. When data of one module is shared with other modules or passed to other modules, this condition is said to be data coupling.
3. **Stamp Coupling –** Stamp coupling simply means the sharing of composite data structure between modules. If the modules interact or communicate by sharing or passing day structure that contains more information than the information required to perform their actions, then these modules are said to be stamp coupled. Therefore, it involves tramp data. It may be necessary due to efficiency factors- this choice made by the insightful designer, not a lazy programmer.
4. **Control Coupling –**  
   Control coupling simply means to control data sharing between modules. If the modules interact or connects by sharing controlled data, then they are said to be control coupled. The controlled coupling means that one module controls the flow of data or information by other modules by them the information about what to do.
5. **External Coupling –**  
   The external coupling means the sharing of data structure or format that are imposed externally between the modules. External coupling is very important but there should be a limit also. It should be limited to a smaller number of modules with structures.
6. **Common Coupling –**  
   Common coupling simply means the sharing of common data or global data between several modules. If two modules share the information through global data items or interact by sharing common data, then they are said to be commonly coupled.
7. **Content Coupling –**  
   Content coupling simply means using of data or control information maintained in other modules by one module. This coupling is also known as pathological coupling. In these coupling, one module relies or depends upon the internal workings of another module. Therefore, if any changes have to be done in the inner working of a module then this will lead to the need for change in the dependent module.