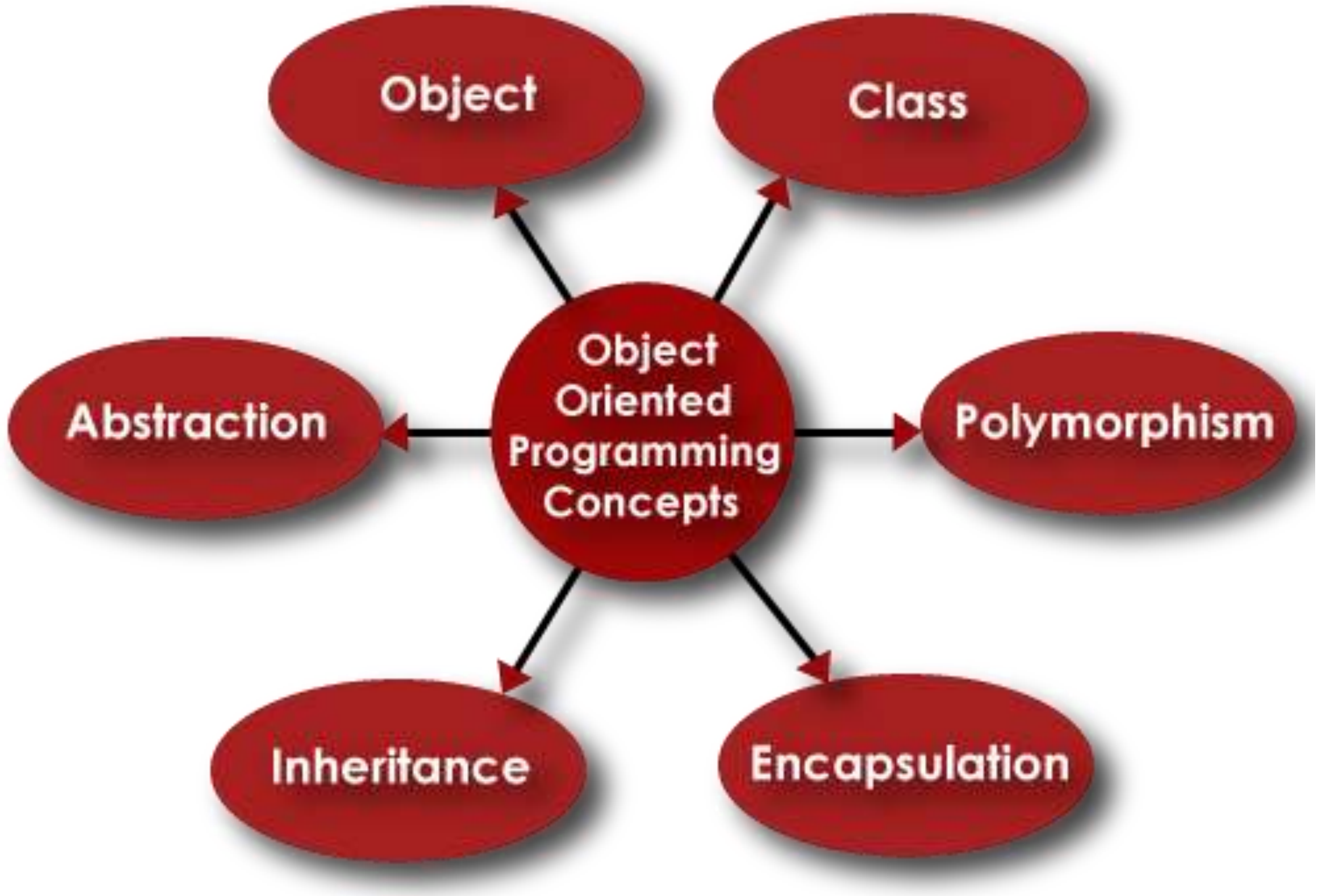
Module 3 (Part I)

OOP Concepts

- Object Oriented is a popular design approach for analyzing and designing an application
- Most of the languages like C++, Java, .net are use object oriented design concept
- Object-oriented concepts are used in the design methods such as classes, objects, polymorphism, encapsulation, inheritance, coupling and cohesion, information hiding, constructor, destructor etc..

- The main advantage of object oriented design is that improving the software development and maintainability
- Another advantage is that faster and low cost development, and creates a high quality software
- The disadvantage of the object-oriented design is that larger program size and it is not suitable for all types of program

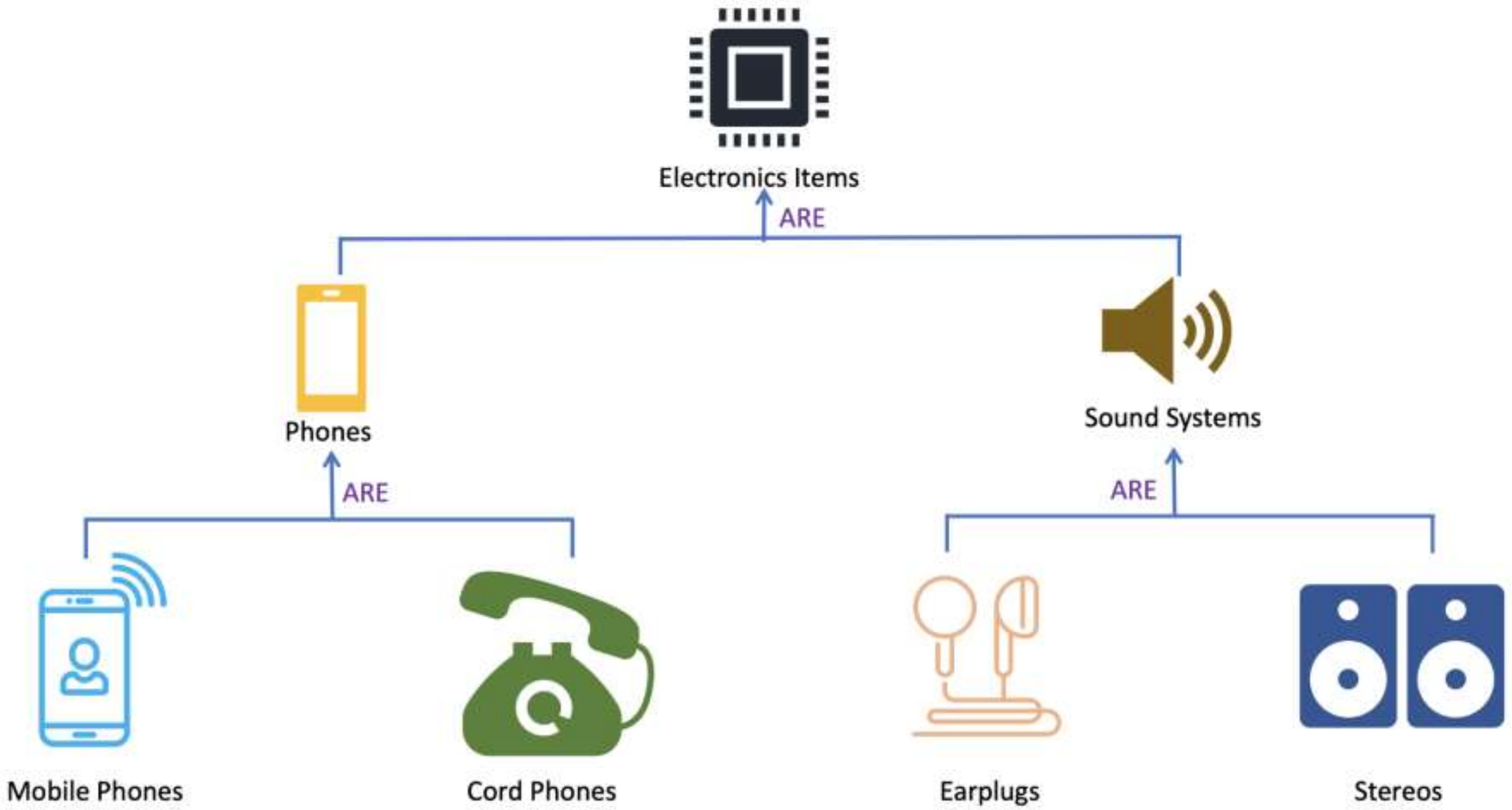
- The different terms related to object design are:

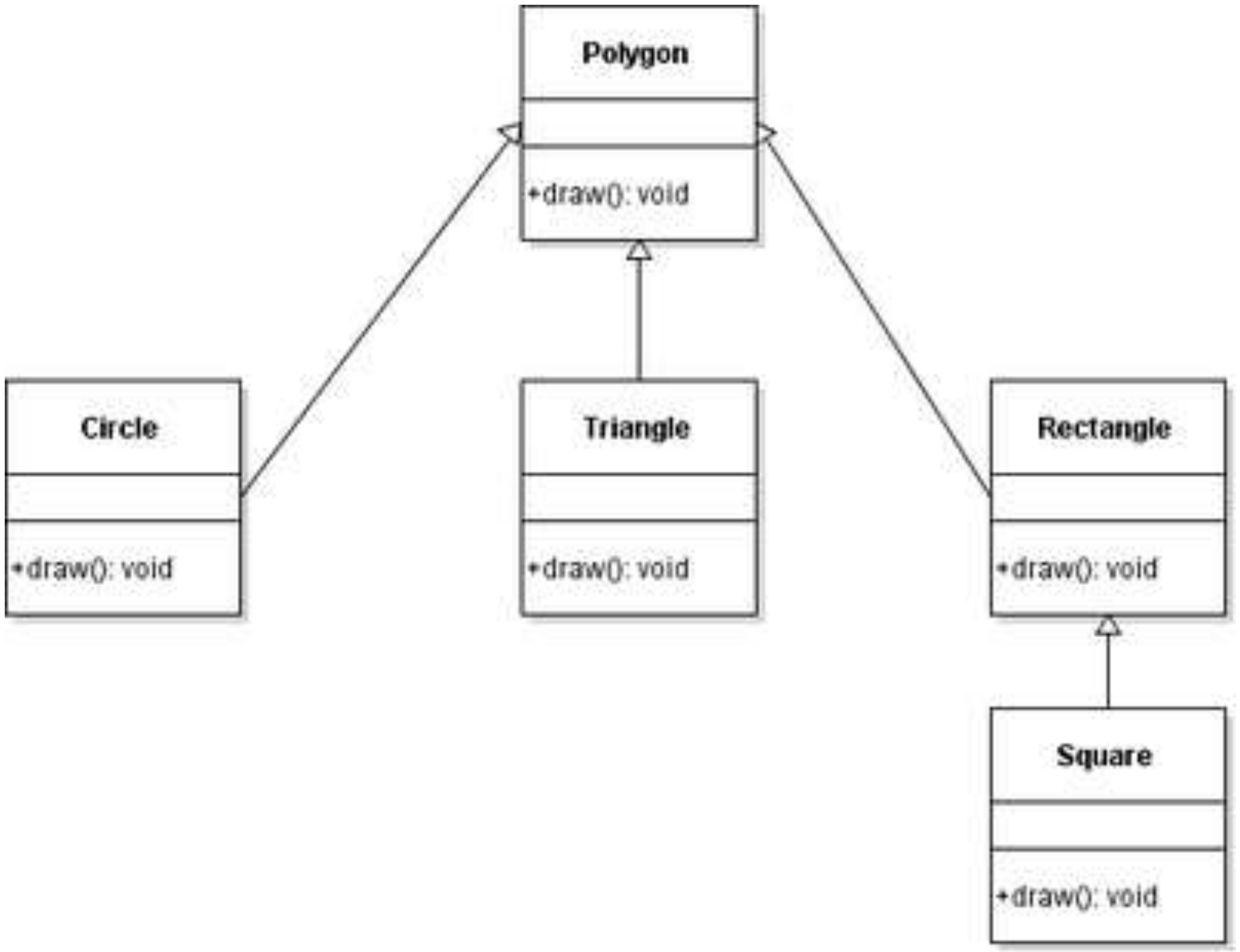


- **Class :**
  - A class is a collection of method and variables
  - It is a blueprint that defines the data and behavior of a type
  - Let’s take HumanBeing as a class
  - A class is a blueprint for any functional entity which defines its properties and its functions
  - Like HumanBeing, having body parts, performing various actions

❖ Inheritance :

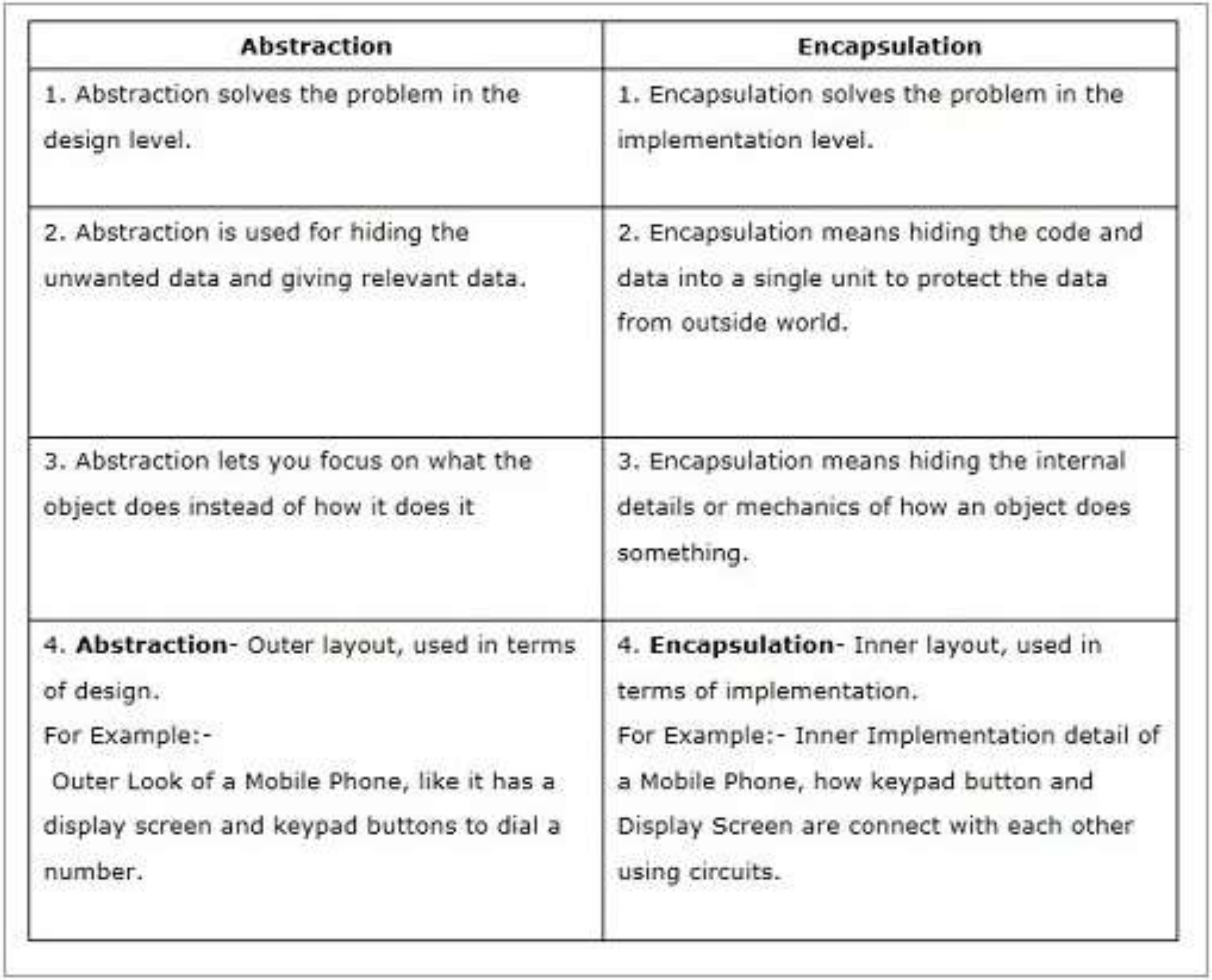
- Inheritance is a feature of object-oriented programming that allows code reusability when a class includes property of another class
- Considering HumanBeing a class, which has properties like hands, legs, eyes, mouth, etc, and functions like walk, talk, eat, see, etc.
- Man and Woman are also classes, but most of the properties and functions are included in HumanBeing
- Hence, they can inherit everything from class HumanBeing using the concept of Inheritance


Electronics Items
ARE
Phones
Sound Systems
ARE
ARE
Mobile Phones
Cord Phones
Earplugs
Stereos


Polygon
+draw(): void
Circle
+draw(): void
Triangle
+draw(): void
Rectangle
+draw(): void
Square
+draw(): void

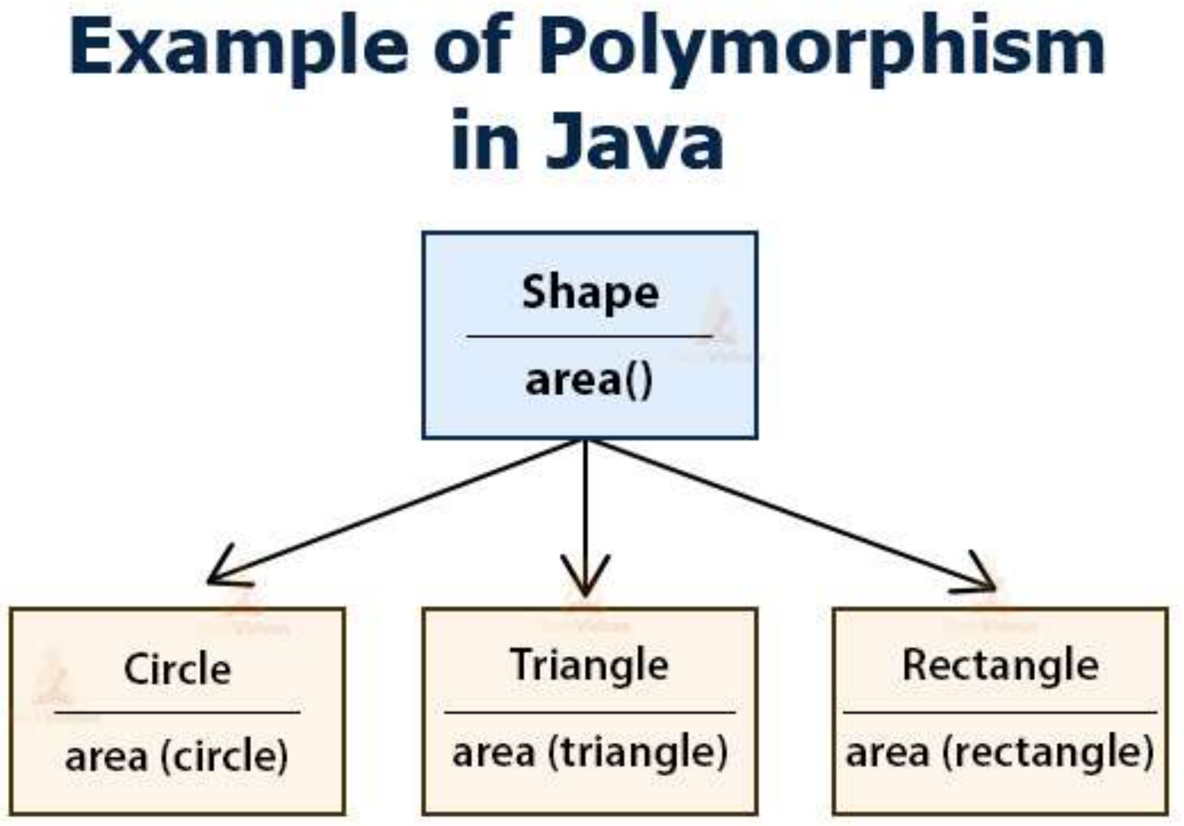
- **Objects :**
  - My name is Akhil, and I am an instance/object of class Man
- **Abstraction :**
  - Abstraction means, showcasing only the required things to the outside world while hiding the details
  - Continuing our example, HumanBeing’s can talk, walk, hear, eat, but the details of the muscles mechanism and their connections to the brain are hidden from the outside world

- **Encapsulation :**
  - Encapsulation means that we want to hide unnecessary details from the user
  - For example, when we call from our mobile phone, we select the number and press call button
  - But the entire process of calling or what happens from the moment we press or touch the call button to the moment we start having a phone conversation is hidden from us



Abstraction	Encapsulation
1. Abstraction solves the problem in the design level.	1. Encapsulation solves the problem in the implementation level.
2. Abstraction is used for hiding the unwanted data and giving relevant data.	2. Encapsulation means hiding the code and data into a single unit to protect the data from outside world.
3. Abstraction lets you focus on what the object does instead of how it does it	3. Encapsulation means hiding the internal details or mechanics of how an object does something.
4. **Abstraction**- Outer layout, used in terms of design. For Example:- Outer Look of a Mobile Phone, like it has a display screen and keypad buttons to dial a number.	4. **Encapsulation**- Inner layout, used in terms of implementation. For Example:- Inner Implementation detail of a Mobile Phone, how keypad button and Display Screen are connect with each other using circuits.

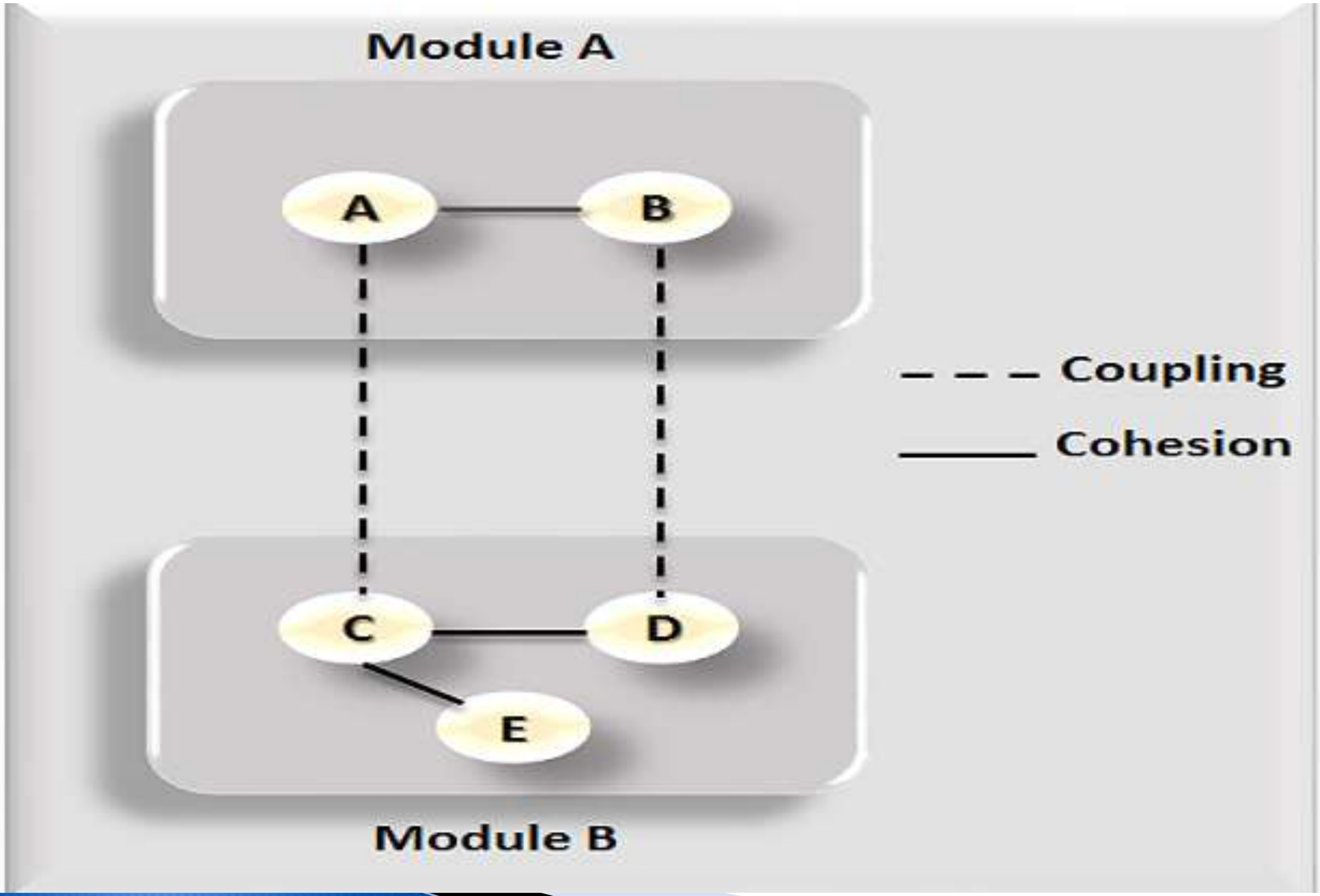
- **Polymorphism :**
  - Polymorphism is a feature of object-oriented programming languages that allows a specific routine to use variables of different types at different times



❖ Cohesion and Coupling

- Cohesion and Coupling:Two OO Design Principles
- **Cohesion** refers to the degree to which elements within a module/object work together to fulfill a single, well-defined purpose. High cohesion means that elements are closely related and focused on a single purpose, while low cohesion means that elements are loosely related and serve multiple purposes.
- **Coupling** refers to the degree of interdependence between software modules. High coupling means that modules are closely connected and changes in one module may affect other modules. Low coupling means that modules are independent, and changes in one module have little impact on other modules.

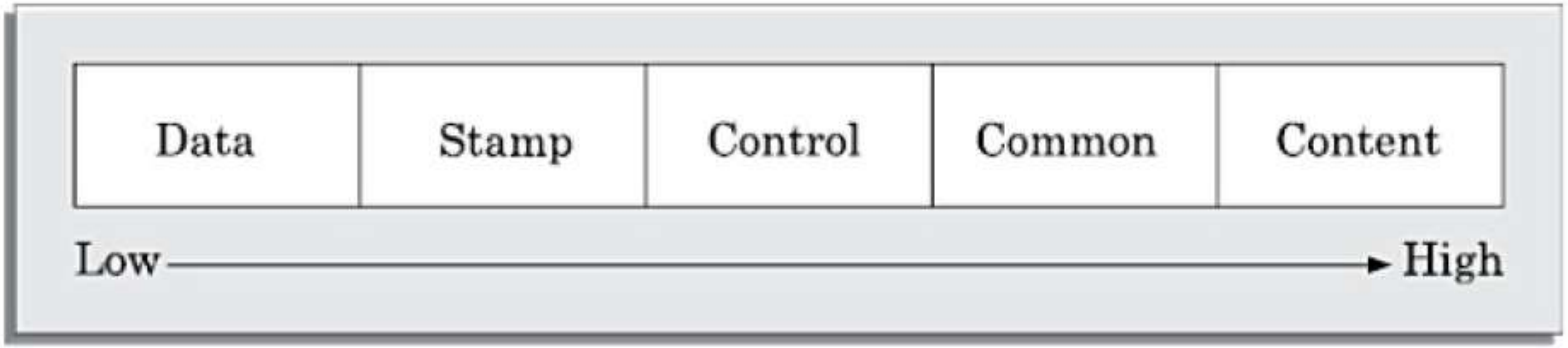
❖ Cohesion and Coupling



❖ Cohesion and Coupling

- Advantages of high cohesion
- Reduced module complexity (they are simpler, having fewer operations).
- Increased system maintainability, as logical domain changes touch fewer modules and changes in one module necessitate fewer changes in others.
- Increased module reusability, as application developers will be able to discover the component they require more readily among the module's cohesive collection of activities.

❖ Coupling



- **Data coupling:** Two modules are data coupled, if they communicate using
- an elementary data item that is passed as a parameter between the two, e.g.
- an integer, a float, a character, etc. This data item should be problem related
- and not used for control purposes.
- **Stamp coupling:** Two modules are stamp coupled, if they communicate
- using a composite data item such as a record in PASCAL or a structure in C.

❖ Cohesion

- **Control coupling:** Control coupling exists between two modules, if data
- from one module is used to direct the order of instruction execution in
- another. An example of control coupling is a flag set in one module and
- tested in another module.
- **Common coupling:** Two modules are common coupled, if they share some
- global data items.
- **Content coupling:** Content coupling exists between two modules, if they
- share code. That is, a jump from one module into the code of another module
- can occur. Modern high-level programming languages such as C do not
- support such jumps across modules.
- The degree of coupling increases from data coupling to content coupling. High coupling among modules not only makes a design solution difficult to
- understand and maintain, but it also increases development effort and also
- makes it very difficult to get these modules developed independently by
- different team members.

- **The concept of "loose coupling and high cohesiveness" underpins many of the design ideas and patterns that have been developed.**