FAQ

Module-1	
Question 1.	**What is the primary purpose of abstraction in software design?**
Answer	The purpose of abstraction is to conceal the deep intricacies of a system, enabling users or developers to concentrate on high-level operations withoutbeing burdened by the complicated internal mechanisms. The use of this approach facilitates the enhancement of system comprehensibility,maintainability, and scalability.
Question 2.	**How does 'Separation of Concerns' relate to abstraction?**
Answer	The design philosophy known as "Separation of Concerns" advocates for thedivision of a software system into discrete portions, with each component dedicated to solving a single feature or problem. The aforementioned technique has an inherent abstract nature, as it permits the independent development, optimisation, or alteration of each segment. This

	ensures that modifications made to one area do not cause disruptions in the others.
Question 3.	**Is there a distinction between the concepts of Data Abstraction and DataEncapsulation?**
Answer	Yes, a differentiation exists. The concept of data abstraction pertains to the notion of conveying fundamental characteristics while excluding extraneous contextual information. On the other hand, Data Encapsulation pertains to theprocess of consolidating both the data (variables) and the methods (functions) that manipulate the data into a cohesive entity. This entity is designed in sucha way that direct access to the data is restricted, and modifications may only be made via well specified interfaces.
Question 4.	**Why is it vital to differentiate between Interface and Implementation insoftware systems?**
Answer	The distinction between Interface and Implementation is crucial for achievingmodularity and flexibility. The interface of a component determines its mode of interaction and use, while the implementation pertains to the underlying mechanisms and processes of that

	component. By maintaining their distinctness, modifications or enhancements may be applied to the implementation without impacting the components or users engaging with the interface.
Question 5.	**Are Abstract Data Types (ADTs) language-specific or can they be used ina cross-language manner?**
Answer	Abstract Data Types (ADTs) are conceptual constructs that operate at a highdegree of abstraction and are not inherently associated with any particular programming language. A collection of operations is defined, which may beexecuted on data, without specifying the implementation details of these actions. This implies that the realisation of ADTs is contingent upon the availability of appropriate language structures in the programming languagesbeing used.