13. **What is encapsulation and how did you use it?**
    - Data hiding by making variables private and providing public getter and setter methods.
    - In my project I created multiple POJO/BEAN classes in order to manage test data and actual data.
        - EX: I take JSON from API response and convert to object of my POJO class all variables are private with getters and setter.

14. **What is the concept of Abstraction?**
    - In OOP, abstraction is a process of hiding the implementation details from the user, only the functionality will be provided to the user.
    - In other words, the user will have the information on what the object does instead of how it does it.
    - In Java, abstraction is achieved using Abstract classes and interfaces.
    - For example: when you log in to your bank account online, you enter your user-id and password and press the login. What happens then, how the input data sent to the server, how it gets verified are all abstracted away from you.

15. **Difference between Abstraction and Encapsulation?**
    - **Abstraction** lets you focus on **what** the object does instead of **how** it does it.
        - **Encapsulation** means hiding the internal details of how the object does something.
    - **Abstraction** is used for hiding the **unwanted** data and giving relevant data.
        - **Encapsulation** means hiding the code and data, and to protect the data from outside.
    - **Abstraction** can achieved by using Abstract class and Interfaces
        - **Encapsulation** can achieved by using "private" keyword.

16. **Difference between Abstract Class and Interface?**
    - Main difference is methods of a Java interface are implicitly abstract and cannot have implementations. A Java abstract class can have instance methods that implements a default behavior.
    - A class that is declared with abstract keyword, is known as abstract class. It can have abstract and non--abstract methods.
    - An Interface is a blueprint of a class. It is a template and it is declared with interface keyword. It can have abstract methods, default methods, static methods and public final static variables
    - When we want to use Abstract class, we use "**extend**" keyword. When we want to use Interface, we use "implement" keyword.
    - Abstract class and interface both are used to achieve abstraction Both cannot be instantiated; we cannot create an object.

17. **What is Polymorphism?**
    - Polymorphism is a very important concept in OOP because;
        - it enables to change the behavior of the applications in the run time based on the object on which the invocation happens.
        - by Polymorphism; one object can have different forms
    - Two types → **Compile Time** which is Static and **Run Time** Polymorphism which is related with child and parent class.
    - Polymorphism is implemented using the concept of Method overloading and method overriding. This can only happen when the classes are under the parent and child relationship using inheritance.

18. **What is Inheritance?**
    - Inheritance represents the **IS-A** relationship which is also known as a parent-child relationship.
    - It is the mechanism in java by which one class is allowed to inherit the features (fields and methods) of another class.
    - The idea behind inheritance in Java is that you can create new classes that are built upon existing classes.
    - When you inherit from an existing class, you can reuse methods and fields of the parent class.
    - Moreover, you can add new methods and fields in your current class also.