**1.Sự khác biệt giữa biến tham chiếu và biến giá trị trong Java là gì?**

* -Trong Java,các kiểu dữ liệu nguyên thủy như int,float,double,.. là biến giá trị.Điều này có nghĩa là giá trị của biến được lưu trực tiếp vào vùng nhớ khi khởi tạo giá trị hoặc khi truyền tham số vào phương thức
* -Ngược lại,các kiểu dữ liệu đối tượng là biến tham chiếu.Biến này chỉ chứa địa chỉ của vùng nhớ nơi đối tượng thực sự được lưu,khi truyền tham số vào phương thức,Java sẽ truyền địa chỉ của đối tượng

**2.Khi nào nên sử dụng Null cho đối tượng Java?**

* Null thường được sử dụng khi biến đối tượng không được khởi tạo giá trị hoặc khi muốn biểu thị rằng biến không trỏ đến bất kì đối tượng nào
* Tuy nhiên sử dụng null cần được sử lý cẩn thận để tránh lỗi NullPointerException,tránh việc sử dụng null quá nhiều bằng cách khởi tạo đối tượng và nên kiểm soát bằng cách sử dụng Optional(isEmpty,isPresent,..)

**3.Sự khác nhau giữa kiểu dữ liệu đối tượng và nguyên thủy trong Java?**

* Kiểu dữ liệu nguyên thủy bao gồm int,float,Boolean,.. Chúng được lưu trực tiếp vào bộ nhớ và chiếm không gian cố định,có giá trị khởi tạo và không null
* Kiểu dữ kiệu đối tương là các đối tượng được tạo ra từ lớp,thay vì lưu trực tiếp giá trị của chúng,biến đối tượng lưu địa chỉ tham chiếu đến vùng nhớ mà đối tượng được lưu,các đối tượng có thể thay đổi kích thước dựa trên số lượng thuộc tính chúng chứa và có thể null

**4.Giải thích cách hoạt động của phương thức equals() và hashCode() trong Java**

* Phương thức equals() dùng để so sánh sự tương đồng giữa hai đối tượng, thay vì so sánh địa chỉ tham chiếu. Mặc định, equals() trong Object so sánh địa chỉ bộ nhớ, nhưng các lớp như String hay Integer đã ghi đè (override) phương thức này để so sánh giá trị thực.
* hashCode() trả về một giá trị số nguyên đại diện cho đối tượng. Nó được sử dụng trong các cấu trúc dữ liệu như HashMap hay HashSet để xác định vị trí của đối tượng. Nếu hai đối tượng được coi là bằng nhau bởi equals(), thì giá trị hashCode() của chúng cũng phải bằng nhau.

**5.Kế thừa đơn,kế thừa đa cấp và đa kế thừa là gì?**

* -Kế thừa đơn là khi một lớp con kế thừa từ một lớp cha duy nhất
* -Kế thừa đơn là khi một lớp con kế thừa từ một lớp cha và lớp cha kế thừa từ một lớp cha khác.
* -Đa kế thừa đc hỗ trợ bởi Interface

**6.Tại sao Java không cho phép kế thừa đa hình qua lớp?**

* -Tại nếu một lớp con được kế thừa từ hai lớp con có cùng phương thức,java sẽ không biết phải sử dụng phương thức nào,để giải quyết vấn để này có thể sử dụng Interface

**7.Giải thích về super và this trong ngữ cảnh của kế thừa?**

* This được sử dụng để tham chiếu đến đối tượng hiện tại trong lớp hiện tại
* Super được sử dụng để tham chiếu đến lớp cha của đối tượng hiện tại.Nó thường được sử dụng để tam chiếu đến constructer hoặc phương thức của lớp cha

**8.Sự khác biệt giữa Abstract class và Interface trong Java?**

* Lớp trừu tượng(abstract class) là lớp có thể chứa các phương thức trừu tượng và phương thức cụ thể,lớp trừu tượng cho phép định nghĩa các hành vi cho lớp con kế thừa
* Interface chỉ chứa các phương thức trừu tượng,không thể chứa các biến instance như lớp trừu tượng
* Nên sử dụng Abstract class khi cần chia sẻ một số hành vi chung,còn giao diện khi muốn định nghĩa hành vi mà không quan tâm đến cách nó thực hiện

**9.Khi nào nên ghi đè(overrider) một phương thức trong lớp con?**

* Ghi đè phương thức là lớp con định nghĩa lại cách hoạt động phương thức đã có trong lớp cha,nó được sử dụng khi muốn thay đổi hành vi của lớp con mà không muốn thay đổi lớp cha
* Quy tắc ghi đè:phương thức ghi đè phải có cùng tên,cùng kiểu trả về và tham số với phương thức trong lớp cha.Phương thức ghi đè phải có mức độ truy cập lớn hơn hoặc bằng phương thức trong lớp cha

**10.Làm sao để kiểm tra một đối tượng có phải là thể hiện của một lớp cụ thể không?**

* Sử dụng toán tử instanceof để kiểm tra xem một đối tượng có thuộc về một lớp hay là một thể hiện của lớp cha hay không

**11.Tính đóng gói là gì tại sao nó quan trọng trong OOP?**

* Tính đóng gói là là cơ chế giới hạn truy cập trực tiếp vào một phương thức và thuộc tính của đối tượng,bảo vệ bên trong và sự thay đổi ngoài ý muốn
* Việc sử dụng private,protected,default,public giúp kiểm soát quyền truy cập,ngăn chặn người dùng bên ngoài sửa đổi các thuộc tính hoặc phương thức mà không thông qua getter và setter được định nghĩa.

**12.Cách thức tạo và sử dụng get/set cho các thuộc tính trong lớp như thế nào?**

* Getter và Setter được sử dụng để cung cấp quyền truy cập gián tiếp vào các thuộc tính của lớp,đảm bảo rằng mọi thay đổi truy cập đến thuộc tính đều đc kiểm soát
* Lợi ích là giúp bảo vệ dữ liệu và cho phép kiểm tra hoặc thay đổi giá trị khi cần thiết trước khi lưu vào thuộc tính.