Test sửa  
Câu hỏi java

<https://quantrimang.com/200-cau-hoi-phong-van-java-hay-thuong-gap-co-dap-an-144832>

<https://viettuts.vn/interview/list-cau-hoi-phong-van-java-core>

andorid

<https://androidcoban.com/tong-hop-cac-cau-hoi-khi-phong-van-lap-trinh-vien-java-va-android.html>

<https://vozforum.org/threads/tong-hop-cac-cau-hoi-phong-van-android-dev-cho-sinh-vien-moi-ra-truong.73551/>

<https://vietjack.com/android/alert_dialog_trong_android.jsp>

<https://thangcoder.com/category/lap-trinh-android/hoc-lap-trinh-android-can-ban/page/3>

Cẩu hỏi String

<https://viettuts.vn/interview/list-cau-hoi-phong-van-java-core-4>

Tài liệu đọc ngon

<https://viettuts.vn/java/bien-trong-java>

<https://freetuts.net/hoc-java/java-core>

<https://vietjack.com/java/bien_trong_java.jsp>

<https://freetuts.net/hoc-java/java-core>

<https://androidcoban.com/page/33>

<https://androidcoban.com/page/30>

Tài liệu đọc ngon2

<https://yellowcodebooks.com/category/lap-trinh-java/page/4/>

<https://gpcoder.com/category/java-core/oop/>

Bài tập ngon

<https://nguyenvanhieu.vn/bai-tap-java-co-loi-giai-chi-tiet/>

Lập trình hướng đối tượng: bao gồm thuộc tính và chức năng

Lập trình hướng đối tượng là mô phỏng thực thế giống lập trình, sử dụng các class để mô phỏng thuộc tính và chức năng

Vd: người chơi, bóng, và sân bóng . Một đối tượng có thuộc tính và chức năng

**4 Kiểu dữ liệu**

Nguyên thủy

1 byte = 8 bit

* Boolean : rẽ nhánh không có
* Byte (8 bit) = 1 byte
* Short (16) = 2 byte
* Int(32) = 4 byte
* Long(64) =8 byte
* Foat (32) =4 bit
* Double(64) =8 bit
* Char bảng chữ cái

Thám chiếu

* String
* arr

Kiểu dữ liệu đối tượng(tham trị)

- arr

- class

- interface

**5. Toán tử**

độ ưu tiên thì ++ và – đc yêu tiên trước nhất

* Toán tử toán học

Chia lấy phần dư 5/4 =1

++ tăng giá trị lên 1 đơn vị

--. Giảm giá trị xuống đơn vị

Việc đưa ++ hoặc – lên trước thể hiện nó làm việc công lên 1 đơn trước rồi mới tăng hoặc giảm rồi mới tăng

In xong mới cộng

* Toán tử quan hệ (trả về true hoặc false)

== kiểm tra 2 toán hạng này có bằng nhau không còn = là phép gán

!=, trả về true false ><, >=, <=;

* Toán tử loglic
* Toán tử gán

=

+=, hoặc -=

Câu hỏi

Có 3 loại biến trong java biến local biến instance và biến static

* Biến local chỉ tồn tại trong hàm hoặc contracto nếu hàm đó kết thúc thì biến cũng kết thúc
* Biến instance được khai báo trong class không tồn tại trong thân hàm là biến không chưa static
* Biến static
* Cũng giống như biến instance biến static được khai báo bên trong lớp. Nhưng lại không nằm trong bất kì một phương thức, một hàm tạo hay khối lệnh nào cả.
* Các biến static được tạo khi chương trình chạy và chỉ bị hủy khi chương trình dừng.
* Thường dùng biến static để khai báo các biến hằng.
* Được khai báo với từ khóa static.
* Biến static được khai báo trong một class với từ khóa “static”, phía bên ngoài các phương thức, constructor và block.
* Sẽ chỉ có duy nhất một bản sao của các biến static được tạo ra, dù bạn tạo bao nhiêu đối tượng từ lớp tương ứng.
* static chỉ là một biến không phụ thuộc vào một đối tượng nào cả, tức là mình có thể truy vấn trực tiếp mà không cần phải khời tạo đối tượng là tiết kiệm bộ nhớ .
* Các biến lớp cũng còn được biết đến với tên gọi là biến tĩnh (biến static) được khai báo với từ khóa static trong một lớp, nhưng nằm ngoài một phương thức, constructor hoặc một khối.

**6.Toán tử trong java**

* Toán tử số học
* Toán tử bit
* Toán tử quan hệ
* Toán tử logic
* Toán tử điều kiện
* Toán tử gán

**8-15 vòng lặp**

* Câu lệnh điều khiển
  + If và switch
* Vòn lặp

**TH. Chuỗi String**

String là kiểu dữ liệu đối tượng. Kiểu đối tượng khác kiểu nguyên thủy ở chỗ nó đựng xây dụng sẵn hoặc do người người định nghĩa

* Kiểu nguyên thủy sẽ đc cấp phát một ô nhớ để lưu giá trị
* Chiều dài của chuỗi .leng
* Nối chuỗi .concat("chuôi cần nối”)
* Vị trí của chuỗi (chuyển và char) chuoi.charAt(4)
* So sách 2 chuỗi (int result = string1.compareTo(String string2);)
* Có 2 cách tạo string 2 cách gán String tenbien =””
* Cách 2 là tạo từ khóa new String str = new String(“”);

**8.1 So sánh equals với == và hashCode va compareTo**

= là phép gán

Còn == là toán tử tham chiếu

Toán tử == so sánh sự **tham chiếu** của đối tượng. nó so sánh với nhau là sự giống nhau về vùng nhớ. Nếu nó cùng 1 đối tượng thì bằng nhau còn không cùng một đối tượng thì không bằng.

dụng toán tử “==”, bộ xử lý của JAVA sẽ so sánh xem 2 biến tham chiếu này có trỏ đến cùng một đối tượng hay không

Còn equals so sánh với nhau cùng một giá trị hay nội dung chuỗi

hashCode trả về một số nguyên là giá trị của vùng nhớ giống equals

Tham khảo thêm

<https://viettuts.vn/java/phuong-thuc-equals-va-hashcode-trong-java>

**9. tính chất của oop**

- tính đống gói: tức là tao sử dụng quyền quản truy cập cập (access modifier) để khai báo các hàm các biến trong class. Hàm nào biến nào mà người dùng có thể truy cập

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phạm vi | Trong class | Trong packge | Ngoài packeg chỉ bởi lớp con(đang kế thừa lớp hiện tại) | Bên ngoài packe |
| Private | c | k | k | k |
| default | c | c | k | k |
| Protected(chuyên dung cho kế thưa) | c | c | c | k |
| Public | c | c | c | c |

* Tính kế thừa : . Tính kế thừa cho phép ta xây dựng một lớp mới dựa trên các định nghĩa của một lớp đã có. kế thừa từ Class, kế thừa từ lớp trừu tượng (*Abstract class*) và kế thừa từ Interface.

**Từ khóa super** được sử dụng trong các trường hợp sau:

Nó được sử dụng để phân biệt các thành phần có cùng tên giữa lớp cha và lớp con.

Nó được sử dụng để gọi hàm tạo của lớp cha từ lớp con.

**Từ khóa super trong java** là một biến tham chiếu được sử dụng để tham chiếu trực tiếp đến đối tượng của lớp cha gần nhất.

* Từ khóa super được sử dụng để tham chiếu trực tiếp đến biến instance của lớp cha gần nhất*. là một biến truy vấn đến biến và phương thức khởi tạo của lớp cha gần nhất*
* super() được sử dụng để gọi trực tiếp Constructor của lớp cha.
* Từ khóa super được sử dụng để gọi trực tiếp phương thức của lớp cha.

**Hàm tạo Constructor :  Hàm tạo trong Java là hàm có cùng tên với lớp và không có giá trị trả về.**

**Từ khóa this: trong java là một biến tham chiếu được sử dụng để tham chiếu tới đối tượng của lớp hiện tại.**

* Tham chiếu tới biến instance của lớp hiện tại.
* Gọi phương thức (method) của lớp hiện tại.
* Gọi hàm dựng (constructor) của lớp hiện tại.
* Trả về instance của lớp hiện tại