1. JDk, JVM,JRE?

JDk (Java develop kit): bộ công cụ phát triển java, bao gồm JRE cùng các công cụ phát triển như là trình biên dịch (javac), trình biên dịch JIT(java),và các tích hợp khác để phát triển java.

JRE (Java Runtime Environment): Môi trường chạy Java, bao gồm JVM và các thư viện cần thiết để chạy ứng dụng Java.

JVM (Java Virtual Machine): Máy ảo Java, một môi trường ảo có khả năng chạy các bytecode Java. JVM chịu trách nhiệm biên dịch bytecode thành mã máy có thể thực thi trên hệ điều hành cụ thể.

(javac: biên dịch mã nguồn java sang bytecode)

1. Heap và Stack?

Heap: là nơi lưu trữ đối tượng,và biến tham chiếu đến đối tượng.Các đối tượng trong heap không được giải phóng khi phương thức kết thúc.

Stack: là nơi lưu trữ các biến nguyên thủy và được giải phóng phương thức kết thúc.

Trình biên dịch JIT ?

Trình biên dịch JIT(just-in-time-compiler): là 1 thành phần trong JVM.Có nhiệm vụ biên dịch bytecode sang ngôn ngữ máy tại thời điểm thực thi chương trình.JIT tối ưu hóa công việc bằng cách chỉ biên dịch mã của từng phương thức khi gọi đến.

1. Platform?

Là môi trường phần cứng và phần mềm mà 1 chương trình hoạt động trên đó. Trong java, thì Java Platform bao gồm JVm và các thư viện java, cho phép viết và chạy trên nhiều hệ điều hành khác nhau.

Vd: SE (java standard edition):java core

EE (java enterprise edition): java web

ME(java Micro edition): mobile, nhúng

1. Sự khác nhau giữa Java platform và các platform khác?

Java platform có đặc điểm "viết một lần chạy mọi nơi" (write once, run anywhere), có thể chạy trên nhiều hệ điều hành và kiến trúc phần cứng khác nhau. Các platform khác thường có đặc điểm riêng, chỉ chạy trên một hệ điều hành hoặc một kiến trúc phần cứng cụ thể.

1. Tính chất "viết một lần chạy nhiều nơi" của Java là gì: Tính chất "viết một lần chạy nhiều nơi" của Java cho phép một ứng dụng Java được phát triển chỉ một lần và có thể chạy trên nhiều hệ điều hành khác nhau (ví dụ: Windows, macOS, Linux) và kiến trúc phần cứng khác nhau (ví dụ: x86, ARM). Điều này là do JVM, làm cho Java trở thành một ngôn ngữ đa nền tảng.
2. Classloader trong Java là gì:

Classloader là một phần của JVM có nhiệm vụ tải các file bytecode (class files) và biên dịch chúng thành các đối tượng Java trong bộ nhớ. Classloader đảm bảo việc tải các class và phân giải các phụ thuộc giữa các class trong quá trình chạy của một ứng dụng Java.

1. File có tên trống ".java" có hợp lệ không:

Không, file có tên trống ".java" không hợp lệ. Tên file Java phải có ít nhất một ký tự và phải kết thúc bằng đuôi ".java".

1. Các từ "delete", "next", "main", "exit" và "null" có phải là từ khóa trong Java không:

Trong Java, "delete", "next", "main", "exit" không phải là từ khóa. Tuy nhiên, "null" là từ khóa trong Java và được sử dụng để đại diện cho giá trị null.

1. Nếu không cung cấp bất kỳ đối số nào trên command line, thì mảng String của hàm main là empty hay null:

Nếu không cung cấp bất kỳ đối số nào trên command line, mảng String của hàm main sẽ là một mảng rỗng (empty), không phải là null. Điều này có nghĩa là args sẽ không trỏ tới null, mà trỏ tới một mảng có độ dài bằng 0. Bạn có thể kiểm tra điều này bằng cách sử dụng args.length để xem độ dài của mảng.

1. Chuyện gì xảy ra nếu khai báo static public void thay vì public static void:

Khai báo static public void thay vì public static void trong khai báo phương thức main không có ảnh hưởng đến việc chạy của chương trình. Cả hai cách khai báo này đều hợp lệ và được hiểu là cùng một cách khai báo.

1. Giá trị mặc định của các biến local là gì:

Biến local không có giá trị mặc định trong Java. Trước khi sử dụng biến local, bạn cần gán giá trị cho nó.

1. Khác nhau giữa Biến tham trị và Biến tham chiếu:

Biến tham trị (pass-by-value): Khi truyền một biến tham trị vào một phương thức, một bản sao của giá trị biến sẽ được tạo và được sử dụng trong phương thức. Thay đổi giá trị của biến trong phương thức không ảnh hưởng đến giá trị biến gốc.

Biến tham chiếu (pass-by-reference): Khi truyền một biến tham chiếu vào một phương thức, tham chiếu đến đối tượng được truyền vào. Thay đổi trạng thái của đối tượng trong phương thức có thể ảnh hưởng đến đối tượng gốc.

1. Bạn hiểu gì về Đối tượng:

Trong Java, đối tượng là một thể hiện của một lớp. Nó được tạo từ khai báo lớp và có thể có các thuộc tính và phương thức. Đối tượng là thực thể cụ thể có thể được sử dụng để thực hiện các thao tác và lưu trữ dữ liệu.

1. Khác nhau giữa lập trình hướng đối tượng và lập trình đối tượng ?

Lập trình hướng đối tượng (OOP) và lập trình dựa trên đối tượng (PO) đều là kỹ thuật lập trình sử dụng ngôn ngữ bậc cao. Tuy nhiên, OOP tập trung vào các đối tượng và các mối quan hệ giữa chúng trong khi PO tập trung vào các thủ tục và các hàm.

1. Định nghĩa lớp ?

Lớp là khuôn mẫu định nghĩa các thuộc tính và phương thức chung của 1 nhóm đối tượng cùng loại.

1. Kiểu biến 1 lớp có thể chứa?

Biến instance: bên trong class nhưng ngoài method,block,constructor

Biến local: bên trong class, constructor, method, block

Biến static: sử dụng từ khóa static.

1. Giá trị khởi tạo của biến tham chiếu đối tượng được định nghĩa là biến instance là gì:

Khi một biến tham chiếu đối tượng (biến instance) được khai báo trong một lớp, giá trị khởi tạo của nó là "null". Điều này có nghĩa là biến chưa tham chiếu đến bất kỳ đối tượng nào và không có đối tượng cụ thể nào được liên kết với nó.

1. Constructor là gì ?

Constructor là 1 phương thức đặc biệt có tên trùng với tên class, không có KDL trả về, trong 1 class có thể có nhìu constructor, mặc định nếu không tạo constructor thì java sẽ tạo 1 constructor rỗng. Trong constructor chỉ được gọi 1 và chỉ 1 constructor khác bằng từ khóa this và phải nằm đầu tiên của phần body.

1. Mục đích của constructor?

Mục đích của constructor là khởi tạo các thuộc tính của đối tượng trong quá trình tạo đối tượng. Nó được sử dụng để cung cấp các giá trị ban đầu cho các biến instance và thiết lập trạng thái ban đầu của đối tượng.

1. Constructor có kế thừa không ?

Constructor không được thừa, lớp con phải tự định nghĩa constructor riêng. Tuy nhiên class con có thể sử dụng từ khóa super để gọi constructor cha để khởi tạo các thuộc tính được kế thừa.

1. Có thể khai báo final cho constructor ko?

Không thể khai báo cho constructor vì từ khóa final dung cho class, methd, thuộc tính.

1. Static là gì?

Static dùng để khai báo phương thức và thuộc tính cho class, không phải của đối tượng,

Có thể truy suất thông qua tên class mà ko cần tạo đối tượng

Mục đích:

+ tạo lớp tiện ích

+ tạo phương thức và thuộc tính cho toàn bộ đối tượng

1. Biến static là gì?

Biến static là biến được chia sẻ cho toàn bộ đối tượng của lớp và tồn tại trong suốt quá trình chạy. Khai báo biến bằng từ khóa static và truy suất thông qua tên class mà ko cần khởi tạo đối tượng.

1. Phương thức static là gì:

Phương thức static là một phương thức thuộc về lớp và không phụ thuộc vào bất kỳ đối tượng cụ thể nào. Phương thức static có thể được gọi trực tiếp thông qua tên lớp mà không cần tạo đối tượng và có thể truy cập vào các biến static khác.

1. tại sao phương thức main là static:

Phương thức main được khai báo là static vì nó là phương thức khởi động của một ứng dụng Java. JVM có thể gọi phương thức main trực tiếp thông qua tên lớp mà không cần tạo đối tượng, vì vậy nó cần được khai báo là static.

1. Chuyện gì xảy ra khi phương thức main không có static:

Khi phương thức main không có từ khóa static, chương trình không thể được chạy được. JVM yêu cầu phương thức main phải là static để có thể gọi trực tiếp thông qua tên lớp mà không cần tạo đối tượng.

1. Khối static là gì:

Khối static (static block) là một khối mã được khai báo trong một lớp và được đánh dấu bằng từ khóa "static". Nó được thực thi chỉ một lần khi lớp được tải vào bộ nhớ và được sử dụng để khởi tạo các biến static hoặc thực hiện các công việc khởi tạo khác.

1. Chúng ta có thể thực thi một chương trình không có phương thức main() không:

Không, chúng ta không thể thực thi một chương trình Java nếu không có phương thức main(). Phương thức main() là điểm khởi đầu của chương trình và JVM sẽ bắt đầu thực thi chương trình từ đó.

1. Sự khác nhau giữa phương thức static và phương thức instance:

+ Phương thức static thuộc về lớp và không phụ thuộc vào đối tượng cụ thể. Nó có thể được gọi trực tiếp thông qua tên lớp mà không cần tạo đối tượng.

+ Phương thức instance phụ thuộc vào đối tượng cụ thể của lớp và phải được gọi thông qua một đối tượng. Nó có thể truy cập các biến instance và chỉ có thể được gọi sau khi tạo đối tượng.

1. "this" trong Java là gì:

"this" là một từ khóa trong Java tham chiếu đến đối tượng hiện tại. Nó được sử dụng để tham chiếu và truy cập các biến instance và phương thức của đối tượng hiện tại trong phạm vi lớp.

1. Kế thừa (inheritance)

Cho phép lớp con sử dụng các phương thức và thuộc tính của lớp cha mà không cần định nghĩa lại

1. Tại sao Java không hỗ trợ con trỏ:

Java không hỗ trợ con trỏ (pointer) vì con trỏ có thể dẫn đến các vấn đề về bảo mật và lỗi trong quản lý bộ nhớ. Con trỏ cho phép truy cập trực tiếp vào địa chỉ bộ nhớ và điều này có thể dẫn đến các lỗi như tràn bộ nhớ (memory overflow) và gây rối giữa các vùng nhớ. Java sử dụng cơ chế tham chiếu để quản lý đối tượng và không cho phép truy cập trực tiếp vào địa chỉ bộ nhớ.

1. Object cloning là gì:

Object cloning là quá trình tạo ra một bản sao của một đối tượng hiện có. Trong Java, việc sao chép đối tượng có thể được thực hiện bằng cách triển khai phương thức clone() từ giao diện Cloneable. Điều này cho phép tạo một bản sao độc lập của đối tượng mà không ảnh hưởng đến đối tượng gốc.

1. Abstract & Interface

Interface có final và static

Default method