|  |  |  |
| --- | --- | --- |
| Stt | Câu hỏi | Câu trả lời |
| 1 | Java là gì? |  |
| 2 | JDK, JRE, JVM | * Jvm( java virtual machine) – là thành phần giúp cho chương trình java được thực thi trên mọi hệ điều hành, ngoài ra còn có chức năng dọn dẹp các bộ nhớ không sử dụng * Jre(java runtime environment) - là phần mềm chứa các phụ thuộc sẵn có như thư viện lớp java, trình tải lớp java và jvm( máy ảo java) . * Jdk ( Java Development Kit) – là phần mềm để xây dựng các ứng dụng java. |
| 3 | Thông dịch, biên dịch khác nhau như thế nào ? | Thông dịch : dịch toàn bộ file .java -> .class  Biên dịch : dịch từng dòng lệnh : .class-> mã máy |
| 4 | JAVA là ngôn ngữ thông dịch hay biên dịch ? | Java là ngôn ngữ vừa thông dịch vừa biên dịch  Java compiler sẽ dịch từ file .java sang file .class chứa bytecode  Sau đó jvm sẽ đem file .class dịch sang ngôn ngữ máy |
| 5 | Java Platform là gì? | Là một tập hợp các chương trình giúp phát triển và chạy các chương trình được viết bằng ngôn ngữ java  Nhờ java platform mà ngôn ngữ java có thể chạy trên bất kỳ hệ điều hành nào , bộ xử lý nào. |
| 6 | Các kiểu dữ liệu trong java, và giá trị mặc định khi khai báo. | * Các kiểu dữ liệu   + Nguyên thủy:   * Number :   + int = 0  + byte:=0  + short=0  +long=0  + float 0.0f  + double= 0.0d   * Char = /u0000 * Boolean = false   + đối tượng :   * String : null * Array: null * Kiểu người dùng tự định nghĩa = null |
| 7 | Sự khác nhau giữa bộ nhớ heap và stack trong java? | |  |  | | --- | --- | | Stack | Heap | | * Bộ nhớ nhỏ * Sử dụng để lưu biến cục bộ * Tự động xóa khi kết thúc hàm | * Bộ nhớ lớn, có thể thay đổi được * Lưu đối tượng(tạo bằng new) * Với java có cơ chế Garbage collection để dọn rác khi không còn ai tham chiếu tới nó. | |
| 8 | Trình bày các cách khởi tạo một mảng trong JAVA ? | Có 2 cách khởi tạo 1 mảng:  C1 – gán trực tiếp String[] arr = {1,2,3}  C2 – khởi tạo trước gán giá trị sau : String[] arr = new String() ; |
| 9 | Phần tử của mảng có thể dùng kiểu dữ liệu nào, và có giá trị mặc định là gì ? | Các kiểu dữ liệu của phần tử mảng:   * Kiểu dữ liệu số : int, double… = 0 * Kiểu dữ liệu ký tự = /n0000 * Kiểu Boolean false * Kiểu đối tượng =null * Kiểu mảng ( mảng đa chiều) arr[] |
| 10 | OOP là gì ? | OOP (Object Oriented Progaming) là phương pháp lập trình ánh xạ các đối tượng ngoài thực tế vào ngôn ngữ lập trình |
| 11 | Phân biệt class và object | |  |  | | --- | --- | | Class | Object | | * Là 1 bản thiết kế hoặc khuôn mẫu để thiết kế ra đối tượng * Là tập hợp các đối tượng có thuộc tính giống nhau * 1 class được khai báo 1 lần với từ khóa class * Lớp không được cấp phát bộ nhớ khi khơi tạo | * Là 1 thể hiện của class * Là 1 thực thể trong thế giới thực có các thuộc tính và hành vi * 1 đối tượng có thể được gọi nhiều lần thông qua từ khóa new * Đối tượng được cấp phát bộ nhớ khi gọi | |
| 12 | Constructor là gì | Contructor là phương thức đặc biệt để khởi tạo đối tượng |
| 13 | Cách khai báo constructor và đặc điểm constructor trong JAVA | * Cách khai báo contructor:   + access modiffie + tên ( giống tên class)   * Đặc điểm contructor:   + tên giống tên class  + không có kiểu dữ liệu trả về  + có thể có nhiều contructor khác tham số cùng tên  + java cung cấp 1 contructor mặc định không có tham số |
| 14 | Phân biệt constructor và method | |  |  | | --- | --- | | **contructor** | **method** | | * Dùng để khởi tạo đối tượng | * Dùng để thể hiện hành vi hành động của đối tượng | | * Được gọi thông qua từ khóa new | * Được gọi thông qua tên mothod | | * Không có kiểu dữ liệu trả về | * Có kiểu dữ liệu trả về( nếu ko có kiểu dữ liệu trả về thì dùng từ khóa void) | | * Tên trùng tên class | * Không bắt bược nhưng không nên đặt tên trùng | | * Java hỗ trợ 1 contructor mặc định không có tham số | * Java không hỗ trợ method mặc định | | * Contructor không thể đi cùng với static, final, astracb | * Có thể | | * Contructor được gọi 1 lần thông qua khởi tạo đối tượng | * Method có thể đươc gọi nhiều lần thông qua đối tượng | |
| 15 | "Tính bao đóng là gì ?  Làm sao để thu được tính bao đóng trong java ?" | * Tính bao đóng( encapsulation) : khả năng ẩn dấu đi các hành vi và thuộc tính không liên quan chỉ thể hiện các hành vi và thuộc tính liên quan * Để thu được được tính bao đóng ta dùng access modifier cho các thuộc tính, và thể hiện các tính chất thông qua getor và settor |
| 16 | Tham trị, tham chiếu | * Tham trị : |
| 17 | Từ khóa static dùng để làm gì ? | * Static dùng để khai báo thuộc tính hành vi của lớp * Mục đích :   + tiết kiệm vùng nhớ  + sử dụng để tạo ra các lớp tiện ích |
| 18 | Ràng buộc khi sử dụng static | * Phương thức static không gọi được non static arivable và non static method * Phương thức static chỉ có thể truy xuất tới biến static * Phương thức static chỉ có thể gọi được phương thức static khác * Phương thức static không thể đi cùng với từ khóa final, super * Có thể khởi tạo static thông qua khối khởi tạo static |
| 19 | Các loại biến trong JAVA | Có 3 loại biến :   |  |  |  | | --- | --- | --- | | **Local(cục bộ)** | **Intance( toàn cục)** | **static** | | * Phạm vi: nằm trong contructor, method, khối block | * Nằm trong class nhưng ngoài method, block | Nằm trong class nhưng nằm ngoài method, block | | * Lưu: bộ nhớ stack | * Biến nguyên thủy: lưu trong stack * Biến tham chiếu lưu trong heap | * Lưu trong bộ nhớ heap | | * Kết thúc khi ra khỏi block,method, cotructor | * Kết thúc khi gán bằng null; | * Kết thúc khi thoát chương trình | |
| 20 | Trình bày các loại access modifier, và phạm vi truy cập. | Có 4 loại access modifier :   |  |  |  |  | | --- | --- | --- | --- | | Private | Pulic | Protected | Default | | Trong class | Cả thế giới | Trong cùng 1 package và ở class con thông qua kế thừa | Nếu ko có acmdf thì mặc định default,trong cùng 1 backage. | |
| 21 | Kế thừa trong JAVA là gì ? | Kế thừa trong java là :  Tạo ra các class con để cụ thể hóa các phương thức và hành vi được định nghĩa ở lớp cha và phát triển thêm các thuộc tính và hành vi mới |
| 22 | Lớp con kế thừa được những tài sản nào(thuộc tính, phương thức) của lớp cha ? | * Lớp con kế thừa tất cả các thuộc tính và phương thức của lớp cha trừ private * Contructor không thể kế thừa * Lớp con là 1 contructor của lớp cha * Java k hỗ trợ đa kế thừa |
| 23 | Lớp Object là gì | Là lớp gốc của mọi lớp  Nếu class không kế thừa class nào thì mặc định nó luôn kế thừa class object |
| 24 | Khái niệm đa hình | Đa hình là khả năng với cùng 1 phương thức nhưng lại thể hiện khác nhau ở các đối tượng khác nhau  Thể hiện của đa hình trong java là overload và overrid |
| 25 | Phân biệt Overloading và Overriding | |  |  | | --- | --- | | **Overload** | **Overrid** | | * Cho phép định nghĩa các phương thức cùng tên nhưng khác tham số truyền vào | * Cho phép định nghĩa các phương thức cùng tên cùng tham số truyền vào | | * Trong cùng 1 class | * Xảy ra ở mối quan hệ kế thừa | | * Biểu hiện đa hình ở complist time | * Biểu hiện đa hình ở runtime | | * Ví dụ : contructor | * Ví dụ toString | |
| 26 | ép kiểu là gì ? các loại ép kiểu | Ép kiểu là : việc gán giá trị của 1 biến có kiểu dữ liệu này sang 1 biến có kiểu dữ liệu khác   * Có 2 loại ép kiểu :   + ép kiểu ngầm định : sử dụng đối với loại dữ liệu nguyên thủy: ép từ kiểu dữ liệu cao xuống kiểu dữ liệu thấp vd: int a = 5 byte b = a ;  + ép kiểu tường minh: là ép từ kiểu dữ liệu thấp lên kiểu dữ liệu cao hơn |
|  |  |  |
| 27 | Tính trừu tượng là gì ? | Là tính chất cho phép bỏ qua các hành vì và thuộc tính ko liên quan, chỉ hiện thị các tính chất liên quan đến đối tượng |
| 28 | Phân biệt abstract class và interface | Astracclass:   * Là lớp trừu tượng đến nỗi không thể tạo đối tượng * Có đầy đủ các thuộc tính và phương thức nhưng ko thể tạo đối tượng * 1 lớp astract thì ko bắt buộc phải có phương thức abstract nhưng 1 lớp có phương thức abstract thì bắt buộc phải là lớp astract * Class con kế thừa lớp abstract thì phải implement lại các phương thức abstract của lớp cha * Class Abstract kế thừa lớp abstract thì k cần phải implement lại   Interface:   * Là 1 cấu trúc tương tự lớp nhưng chỉ chứa các hằng số và astract method , sinh ra để tạo bản thiết kế chứa các method dùng chung của các class * Không có contructor, ko thể tạo đối tượng * Mặc định hằng số là pulic static final * Mặc định method là pulic abstract method * Khi 1 lớp implement 1 interface thì bắt buộc phải overid lại tất cả các phương thức của nó * 1 interface có thể kế thừa nhiều interface khác |
| 29 |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |