**BỘ CÂU HỎI PHỎNG VẤN MODULE 2**

**ADVANCED PROGRAMMING WITH JAVA**

|  |  |
| --- | --- |
| **STT** | **Câu hỏi** |
| 1 | **Sự khác nhau giữa JDK,JRE và JVM?**  JVM (viết tắt của Java Virtual Machine) là một thiết bị trừu tượng (ảo) có thể giúp máy tính chạy các chương trình Java  JRE (là viết tắt của Java Runtime Environment) được sử dụng để cung cấp môi trường runtime. Nó là trình triển khai của JVM  JDK (là viết tắt của Java Development Kit) bao gồm JRE và các Development Tool |
| 2 | **Sự khác nhau giữa bộ nhớ heap và stack trong java?** |
| 3 | **Trình biên dịch JIT là gì?** |
| 4 | **Platform là gì?**  Bất cứ môi trường phần cứng hoặc phần mềm nào mà trong đó có một chương trình chạy, thì được hiểu như là một Platform. Với môi trường runtime riêng cho mình (JRE) và API, Java được gọi là Platform |
| 5 | **Sự khác nhau giữa Java platform và các platform khác?** |
| 6 | **Tính chất "viết một lần chạy nhiều nơi" của java là gì?** |
| 7 | **Classloader trong java là gì?** |
| 8 | **File có tên trống ".java" có hợp lệ không?**  Có, bạn có thể lưu file java với tên ".java", sau đó biên dịch bằng lệnh **javac .java** và chạy bằng lệnh **java** |
| 9 | **Các từ delete, next, main, exit và null có phải là từ khóa trong java không?**  Không |
| 10 | **Nếu không cung cấp bất kỳ đối số nào trên command line, thì mảng String của hàm main là empty hay null?**  Mảng String là empty ko phải null |
| 11 | **Chuyện gì xảy ra nếu khai báo static public void thay vì public static void?**  Chương trình được biên dịch và run đúng. |
| 12 | **Giá trị mặc định của các biến local là gì?**  Biến local không được khởi tạo các giá trị mặc định nào |
| 13 | **Khác nhau Biến tham trị và Biến tham chiếu?**  – Tham trị tức là truyền giá trị – Tham chiếu tức là chiếu đến 1 địa chỉ |
| 14 | **Bạn hiểu gì về Đối tượng?**  Đối tượng là một biểu hiện cụ thể của lớp mô tả những hành vi và trạng thái được gọi là đối tượng |
| 15 | **Định nghĩa Lớp (class)?**  Lớp là khuôn mẫu mô tả những hành vi và thuộc tính chung của một số đối tượng. |
| 16 | **Kiểu biến nào một lớp có thể chứa?**  Một lớp có thể gồm biến local, biến instance, và biến lớp. |
| 17 | **Biến local hay biến cục bộ là gì?**   * Biến cục bộ được khai báo trong các phương thức, hàm contructor hoặc trong các block. * Biến cục bộ được tạo bên trong các phương thức, contructor, block và sẽ bị phá hủy khi kết thúc các phương thức, contructor và block. * Không được sử dụng “access modifier” khi khai báo biến cục bộ. * Các biến cục bộ sẽ nằm trên vùng bộ nhớ stack của bộ nhớ. * Bạn cần khởi tạo giá trị mặc định cho biến cục bộ trước khi có thể sử dụng. |
| 18 | **Biến instance là gì?**   * Biến instance được khai báo trong một lớp(class), bên ngoài các phương thức, constructor và các block. * Biến instance được lưu trong bộ nhớ heap. * Biến instance được tạo khi một đối tượng được tạo bằng việc sử dụng từ khóa “new” và sẽ bị phá hủy khi đối tượng bị phá hủy. * Biến instance có thể được sử dụng bởi các phương thức, constructor, block, ... Nhưng nó phải được sử dụng thông qua một đối tượng cụ thể. * Bạn được phép sử dụng "access modifier" khi khai báo biến instance, mặc định là "default". * Biến instance có giá trị mặc định phụ thuộc vào kiểu dữ liệu của nó * Bên trong class mà bạn khai báo biến instance, bạn có thể gọi nó trực tiếp bằng tên khi sử dụng ở khắp nới bên trong class đó. |
| 19 | **Sự khác biệt giữa ngôn ngữ lập trình hướng đối tượng và ngôn ngữ lập trình dựa trên đối tượng là gì?**  Ngôn ngữ lập trình dựa trên đối tượng thực hiện theo tất cả các tính năng của OOPs ngoại trừ tính kế thừa. Ví dụ về các ngôn ngữ lập trình dựa trên đối tượng là JavaScript, VBScript,... |
| 20 | **Giá trị khởi tạo của biến tham chiếu đối tượng được định nghĩa là biến instance là gì?**  Trong java, các tham chiếu của đối tượng được khởi tạo là null. |
| 21 | **Constructor là gì?**  Constructor giống như một phương thức được sử dụng để khởi tạo trạng thái của một đối tượng. Nó được gọi ra vào thời điểm tạo ra đối tượng. |
| 22 | **Mục đích của constructor là gì?**  Constructor mặc định cung cấp các giá trị mặc định cho các đối tượng. Trình biên dịch java tạo ra một constructor mặc định chỉ khi không có constructor nào trong lớp. |
| 23 | **Constructor trả về kiểu giá trị gì?**  Constructor trả về thể hiện của lớp hiện tại. |
| 24 | **Constructor được kế thừa không?**  Không, constructor không được kế thừa. |
| 25 | **Có thể tạo constructor final không?**  không, constructor không thể là final. |
| 26 | **Biến static là gì?**   * Biến static có thể được sử dụng để tham chiếu thuộc tính chung của tất cả đối tượng * Biến static lấy bộ nhớ chỉ một lần trong Class Area tại thời gian tải lớp đó. |
| 27 | **Phương thức static là gì?**   * Một phương thức static thuộc lớp chứ không phải đối tượng của lớp. * Một phương thức static gọi mà không cần tạo một instance của một lớp. * Phương thức static có thể truy cập biến static và có thể thay đổi giá trị của nó. |
| 28 | **Tại sao phương thức main là static?** |
| 29 | **Khối static là gì?** |
| 30 | **Chúng ta có thể thực thi một chương trình không có phương thức main() không?** |
| 31 | **Chuyện gì xảy ra khi phương thức main không có static?** |
| 32 | **Sự khác nhau giữa phương thức static và phương thức instance?** |
| 33 | **this trong java là gì?** |
| 34 | **Kế thừa là gì?** |
| 35 | **Lớp nào là lớp cha cho tất cả các lớp.** |
| 36 | **Tại sao đa kế thừa không được hỗ trợ trong java.** |
| 37 | **Composition là gì?** |
| 38 | **Sự khác nhau giữa aggregation và composition?** |
| 39 | **Tại sao java không support con trỏ?** |
| 40 | **super trong java là gì?** |
| 41 | **Có thể sử dụng cả this() và super() trong một constructor?** |
| 42 | **Object cloning là gì?** |
| 43 | **Overloading (nạp chồng) phương thức là gì?** |
| 44 | **Tại sao overloading phương thức không xảy ra khi thay đổi kiểu giá trị trả về?** |
| 45 | **Có thể overload phương thức main() không?** |
| 46 | **Ghi đè (overriding) phương thức là gì?** |
| 47 | **Có thể ghi đè phương thức static không?** |
| 48 | **Tại sao không thể ghi đè phương thức static?** |
| 49 | **Có thể ghi đè phương thức đã nạp chồng?** |
| 50 | **Có thể ghi đè biến instance không?** |
| 51 | **Sự khác nhau giữa nạp chồng và ghi đè là gì?** |
| 52 | **Biến final là gì?** |
| 53 | **Phương thức final là gì?** |
| 54 | **Lớp final là gì?** |
| 55 | **Biến final blank là gì?** |
| 56 | **Có thể khởi tạo giá trị cho biến final blank không?** |
| 57 | **Có thể khai báo phương thức main là final không?** |
| 58 | **Đa hình tại runtime là gì?** |
| 59 | **Có thể thực hiện đa hình lúc runtime với các thành viên dữ liệu không?** |
| 60 | **Sự khác nhau giữa ràng buộc tĩnh và ràng buộc động là gì?** |
| 61 | **Trừu tượng là gì?** |
| 62 | **Sự khác nhau giữa trừu tượng và đóng gói là gì?** |
| 63 | **Lớp trừu tượng là gì?** |
| 64 | **Có phương thức trừu tượng không nằm trong lớp trừu tượng không?** |
| 65 | **Có thể sử dụng cả abstract và final cho một phương thức không?** |
| 66 | **Có thể tạo thể hiện của lớp trừu tượng không?** |
| 67 | **Interface là gì?** |
| 68 | **Có thể khai báo một phương thức của interface với từ khóa static không?** |
| 69 | **Một interface có thể là final không?** |
| 70 | **Marker interface là gì?** |
| 71 | **Sự khác nhau giữa lớp abstract và interface là gì?** |
| 72 | **Có thẻ định nghĩa private hoặc protected cho các biến trong interface không?** |
| 73 | **Khi nào một tham chiếu đối tượng có thể được ép sang kiểu interface tham chiếu?** |
| 74 | **Package là gì?** |
| 75 | **Có cần import package import java.lang không? tại sao?** |
| 76 | **Có thể import package/lớp giống nhau hai lần không? JVM sẽ tải package hai lần khi chạy không?** |
| 77 | **Static import là gì?** |
| 78 | **Xử lý ngoại lệ (handling exception) là gì?** |
| 79 | **Sự khác biệt giữa checked exception và unchecked exception là gì?** |
| 80 | **Có phải mỗi khối try phải đi kèm với một khối catch?** |
| 81 | **Khối finally là gì?** |
| 82 | **Khối finally có thể được sử dụng mà không cần khối catch không?** |
| 83 | **Có trường hợp nào khối finally không được thực thi không?** |
| 84 | **Sự khác nhau giữa throw và throws là gì?** |
| 85 | **Có thể khai báo phương thức overriding của lớp con một ngoại lệ nếu phương thức của lớp cha không throw một ngoại lệ?** |
| 86 | **Việc tuyên truyền ngoại lệ là gì?** |
| 87 | **Có vấn đề gì không với thứ tự của các lệnh catch được viết cho FileNotFoundException và IOException?** |
| 88 | **Điểm khác nhau giữa Error và Exception?** |
| 89 | **Ý nghĩa của immutable (bất biến) trong String là gì?** |
| 90 | **Tại sao các đối tượng String trong java là immutable?** |
| 91 | **Có bao nhiêu cách để tạo ra một đối tượng String trong java?** |
| 92 | **Có bao nhiêu đối tượng String được tạo ra trong đoạn code sau?** |
| 93 | **Tại sao java sử dụng khái niệm string literal?** |
| 94 | **Có bao nhiêu đối tượng được tạo ra trong đoạn code sau?** |
| 95 | **Sự khác nhau giữa String và StringBuffer là gì?** |
| 96 | **Sự khác nhau giữa StringBuffer và StringBuilder là gì?** |
| 97 | **Làm thế nào để tạo lớp immutable trong java?** |
| 98 | **Mục đích của phương thức toString() trong java là gì?** |
| 99 | **Em biết gì về String Pool?** |
| 100 | **Nested class (lớp lồng nhau) là gì?** |
| 101 | **Có sự khác nhau giữa nested class và inner class không?** |
| 102 | **Nested interface là gì?** |
| 103 | **Có thể khai báo interface trong class không?** |
| 104 | **Có thể khai báo một class trong interface không?** |
| 105 | **Sự khác nhau giữa Array và ArrayList là gì?** |
| 106 | **Lợi thế của ArrayList so với các mảng?** |
| 107 | **Hoạt động xóa trong LinkedList là nhanh hơn trong ArrayList, tại sao?** |
| 108 | **Bạn quyết định khi nào sử dụng ArrayList và LinkedList?** |
| 109 | **Sự khác nhau giữa ArrayList và LinkedList là gì?** |
| 110 | **Sự khác nhau giữa ArrayList và Vector là gì?** |
| 111 | **Sự khác nhau giữa Iterator và ListIterator là gì?** |
| 112 | **Sự khác nhau giữa List và Set là gì?** |
| 113 | **Sự khác nhau giữa Set và Map là gì?** |
| 114 | **Sự khác nhau giữa Stack và Queue là gì?** |
| 115 | **Sự khác nhau giữa Collection và Collections là gì?** |
| 116 | **Sự khác nhau giữa Comparable và Comparator là gì?** |
| 117 | **Phương thức hashCode() là gì?** |
| 118 | **Tại sao chúng ta phải ghi đè phương thức equals()?** |
| 119 | **Làm thế nào để đồng bộ List, Set và Map?** |
| 120 | **Lợi ích của generic collection là gì?** |
| 121 | **Tại sao Generic được sử dụng trong Java?** |
| 122 | **Bạn hiểu gì về Access Modifier?** |
| 123 | **Phân biệt public, protected, private, default?** |
| 124 | **Bạn hiểu gì về Đa luồng (Multi-Thread)?** |
| 125 | **Phân biệt Multitasking, multiprocessing, multithreading** |
| 126 | **Thread có thể được tạo bằng hai cách nào? So sánh** |
| 127 | **Phân biệt run(), start()** |
| 128 | **Vòng đời Thread** |
| 129 | **Giải thích trình dọn rác (Garbage Collector) trong Java?** |
| 130 | **Các cách mà một Thread có thể đi vào trạng thái đợi (waiting state)?** |
| 131 | **Cách mà Đa luồng (Multi-Thread) diễn ra trên một máy tính với một CPU đơn?** |
| 132 | **Một Thread đã chết có thể được restart lại như thế nào?** |
| 133 | **Điểm khác nhau giữa yield và sleep?** |
| 134 | **Phương thức sleep() và wait() khác nhau ở điểm nào?** |