1. **JAVA là gì ?**

* Là một ngôn ngữ lập trình hướng đối tượng
* Có khả năng thực thi ở nhiều loại thiết bị
* Được sử dụng rộngrãi
* Java có tính độc lập nền tảng(platform independent)
* Có câu slogan **:”write one, run anywhere”**
* Một chương trình Java có thể chạy trên các nền tảng khác nhau mà không phải biên dịch lại

1. **JDK, JRE, JVM**

JDK (Java Develoment Kit):

Là bộ công cụ phát triển Java, giúp LTV viết mã và chạy các chương trình Java.

JRE(Java Runtime Environment):

Là phần mềm cung cấp các thư viện lớp Java với các thành phần cần thiết để chạy mã Java

JVM (Java Vitual Machine):

Là máy ảo Java

1. **Thông dịch, biên dịch khác nhau như thế nào ?**
2. **JAVA là ngôn ngữ thông dịch hay biên dịch ?**

Java là 1 ngôn ngữ vừa biên dịch vừa thông dịch, vì:

* Trình BIÊN DỊCH của Java (Java compiler), nó sẽ chuyển đổi các file code thành java byte-code. => cái này chưa phải mã máy.
* Rồi sau đó Java Virtual Machine (JVM) mới thông dịch chạy java byte code thành ngôn ngữ máy.

1. **Java Platform là gì?**
2. **Các kiểu dữ liệu trong java, và giá trị mặc định khi khai báo.**

Các kiểu dữ liệu gồm

* 8 Kiểu nguyên thủy  
  byte : 0

Short: 0

Int : 0

Long : 0l

Float : 0.0f

Double: 0.0d

Boolean: false  
char :’\u0000’.

* Kiểu đối tượng:

Object : null  
String: “”

1. **Sự khác nhau giữa bộ nhớ heap và stack trong java?**
2. **Trình bày các cách khởi tạo một mảng trong JAVA ?**
3. **Phần tử của mảng có thể dùng kiểu dữ liệu nào, và có giá trị mặc định là gì ?**

Phần tử của mảng có thể dung kiểu dữ liệu:   
- Kiểu nguyên thủy :

* Kiểu String:

và tất cả các phần tử phải có chung kiểu dữ liệu

1. **OOP là gì ?**

Là phương pháp giúp LTV đưa các đối tượng ngoài thực tế vào ngôn ngữ lập trình.

Gồm có 4 tính chất:  
Encapsulation: Tính bao đóng giúp bảo mật thông tin của đối tượng chỉ đưa ra những thông tin cần thiết.

Inheritance: Tính kế thừa giúp các lớp con sử dụng phương thức và thuộc tính của lớp cha mà không cần định nghĩa lại.

Abstaction: Tính trừu tượng giúp loại bỏ những thứ không cần thiết và giữ lại những việc cần thiết đối với chương trình.

Polymosphims: Tính đa hình giúp cùng 1 hành động nhưng có nhiều thể hiện ở nhiều ngữ cảnh khác nhau.

1. **Phân biệt class và object**
2. **Constructor là gì**
3. **Cách khai báo constructor và đặc điểm constructor trong JAVA**
4. **Phân biệt constructor và method**
5. **"Tính bao đóng là gì ? Làm sao để thu được tính bao đóng trong java ?"**
6. **Tham trị, tham chiếu**
7. **Từ khóa static dùng để làm gì ?**
8. **Ràng buộc khi sử dụng static**
9. **Các loại biến trong JAVA**
10. **Trình bày các loại access modifier, và phạm vi truy cập.**
11. **Kế thừa trong JAVA là gì**
12. **Lớp con kế thừa được những tài sản nào(thuộc tính, phương thức) của lớp cha ?**
13. **Lớp Object là gì**

Lớp Object là lớp cha của tất cả các lớp ở trong java.

1. **Khái niệm đa hình**
2. **Phân biệt Overloading và Overriding**

|  |  |
| --- | --- |
| Overloading | Overriding |
| * Là cơ chế cho phép 1 lớp có nhiều phương thức cùng tên nhưng khác nhau về tham số truyền vào * Xảy ra trong cùng 1 class * Đa hình tại compile | * Là cơ chế cho phép lớp con định nghĩa lại phương thức trước đó ở lớp cha, có cùng tên cùng danh sách tham số, cùng kiểu trả về. * Access Modifier phải bằng hoặc cao hơn ở lớp cha. * Xảy ra trong quan hệ Is-a * Đa hình tại Run time |

1. **ép kiểu là gì ? các loại ép kiểu**

Ép kiểu là hình thức chuyển đổi tham chiếu đối tượng từ 1 kiểu này sang tham chiếu đối tượng kiểu khác.

Có 2 loại ép kiểu:  
Ép kiểu tường minh:  
LTV tự động ép kiểu( nếu không sẽ lỗi compile) khi ép kiểu từ class cha sang con

Ép kiểu ngầm định:

Khi ép kiểu từ class con về class cha thì trình biên dịch sẽ tự động thực thi, LTV không cần phải thực hiện.

Dùng từ khóa instanceOf để kiểm tra Object đó có thuộc class không.

1. **Tính trừu tượng là gì ? =>**
2. **Phân biệt abstract class và interface**

|  |  |
| --- | --- |
| **Abstract Class** | **InterFace** |
| * Không thể khởi tạo đối tượng | * Không thể khởi tạo đối tượng |
| * **Class** có ít nhất 1 phương thức **abstract** thì chắc chắn **class** đó là **abstract** | * Các trường trong **interface** đều là hằng số |
| * 1 **class abstract** thì có thể có hoặc không có **phương thức abstract** * **Class abstract** **extend** từ **abstract** thì **class** **con** không cần **override** phương thức của lớp cha | * Tất cả các **phương thức** đều là **abstract** * Hỗ trợ đa kế thừa * Không tạo được **constructor** nhưng có 1 **constructor** không tham số mặc định ở trong java 8 |
| * Không đi kèm từ khóa **final** |  |
| * **Method abstract** thì không có **AM private** |  |

1. **So sánh Array và ArrayList**

|  |  |
| --- | --- |
| Array | ArrayList |
| * Kích thước cố định * Chỉ có thuộc tính length. * Có thể lưu kiểu nguyên thủy và object * Tốc độ thao tác nhanh hơn so với arraylist | * Kích thước có thể thay đổi * Có nhiều phương thức hỗ trợ. * Chỉ cho phép lưu kiểu đối tượng,đối với kiểu nguyên thủy thì sẽ chuyển về kiểu đối tượng theo cơ chế autoboxing |

1. **So sánh ArrayList và LinkedList**

|  |  |
| --- | --- |
| **ArrayList** | **LinkedList** |
| * Sử dụng mảng để lưu trữ dữ liệu * Truy xuất phần tử nhanh, thêm xóa và sửa chậm hơn so với linkedlist | * Lưu trữ dữ liệu dưới dạng danh sách liên kết * Thêm xóa sửa nhanh hơn, truy xuất chậm |

1. **So sánh List, Set và Queue ?**

|  |  |  |
| --- | --- | --- |
|  |  |  |

1. Set là gì, các lớp triển khai của Set
2. **Generic là gì**

* Là cơ chế truyền kiểu dữ liệu như là tham số
* Cho phép sử dụng cho method,class, interface với nhiều kiểu dữ liệu khác nhau.

1. **Ưu điểm và hạn chế khi dùng generic ?**

* **Ưu điểm:**
* Không cần phải ép kiểu
* Dung được cho phương thức và class
* Tái sử dụng code
* Bắt được lỗi lúc compile
* **Nhược điểm:**
* Không dung chung với từ khóa static
* K new được đối tượng
* Không tạo được mảng
* Không dung cho kiểu nguyên thủy

1. **Stack là gì, các phương thức của stack ?**

* Thực hiện các thao tác với dữ liệu thông qua phương pháp first in, last out(FILO)
* Push(),peek(),pop(),isEmpty(),

1. **Queue là gì, các class triển khai của queue ?**

* Thực hiện các thao tác với dữ liệu theo phương pháp first in, first out(FIFO).
* Linkedlist , piorityqueue, arraydeque

1. **Phương thức cơ bản của queue?**
2. So sánh Comparable và Comparator, khi nào dùng cái nào ?
3. Map là gì, các class triển khai
4. Phương thức equals() và hashcode() trong Java là gì?
5. Cây nhị phân là gì
6. Trình bày các cách duyệt cây nhị phân.
7. Ngoại lệ (Exception là gì) ?
8. Phân loại Exception
9. Phân biệt Error và Exception
10. Có bao nhiêu cách để xử lý ngoại lệ (handle)
11. Một số lưu ý khi dùng try-catch
12. Serialization là gì ? lưu ý khí sử dụng Serialization
13. SOLID là gì? Phát biểu của từng nguyên lý
14. Design pattern là gì? Chia làm bao nhiêu loại. Tổng số là bao nhiêu?