1)

Cú pháp :

-Vòng lặp for:

Vòng lặp for cần 3 biểu thức gồm giá trị (initialization),điều kiện(condition),sự tăng giảm(increment/decrement)

for(initialization; condition; increment/decrement){

}

Vd for(int i=0;i<=100;i++){

System.out.print(i,+” ”);

}

-Vòng lặp while

While(condition){

}

Vd

While(i<=100){

System,out.print(i,+” ”);

i++;

}

Vòng lặp while thì chúng ta cần 1 biểu thức là là biểu thức điều kiện

Cách hoạt động:

-Vòng lặp for:

Ở vòng lặp for thì trước tiên chúng sẽ thực hiện lệnh khởi tạo trước tiên. Sau đó, nó sẽ kiểm tra xem điều kiện có thỏa mãn hay không sau đó mới thực hiện các lệnh trong vòng lặp. Cuối cùng, chúng thực hiện lệnh tăng hoặc giảm và tiếp tục một vòng lặp mới cho tới khi điều kiện trả về sai

-vòng lặp while:

vòng lặp while đầu tiên, nó sẽ kiểm tra xem điều kiện. Nếu điều kiện là đúng và chúng sẽ thực hiện vòng lặp và chỉ kết thúc khi điều kiện sai

mục đích sự dụng:

-vòng lặp for:

Vòng lặp for sẽ được sử dụng khi chúng ta đã biết trước được số lần lặp cùng với đó là khi lệnh khởi tạo và lệnh tăng/giảm đơn giản VD: xét tuần tự các thành phần trong một mảng, hay xét các số trong khoảng từ m tới n,…

-vòng lặp while:

Vòng lặp while sẽ được sử dụng khi chúng ta khó mà có thể biết được số lần lặp lại của nó hay việc khởi tạo cho vòng lặp khá phức tạp VD: tránh việc người dùng nhập sai đầu vào, kiểm tra xem hệ thống mạng có còn hoạt động hay không

2)

Vòng lặp for có thể thực đc vòng lặp ko xác định

Lí do:

Nếu tât cả 3 biểu thức đều trống thì thì sẽ tạo thành 1 vòng lặp ko xác định vì ko xác định đc số lần lặp và đk kiểm tra vòng lặp

For(; ;){

System.out.print(“unidentified”);

}

3)

Có thể trả về nhiều giá trị trong java

4)

Lớp (class) là mô tả về các đối tượng sẽ được tạo ra. Mỗi lớp có tên lớp, trong lớp có các thuộc tính như là dữ liệu của lớp và các ứng xử của lớp đó (các hàm) gọi là các phương thức

Cấu trúc tổng quát của lớp:

Tên lớp,các thuộc tính,phương thức của lớp

5)

* public: có thể truy cập ở mọi nơi trong Project.
* protected: truy cập được từ trong lớp khai báo, lớp con của lớp khai báo và các lớp cùng gói với lớp khai báo.
* default: truy cập được từ trong lớp khai báo và các lớp cùng gói với lớp khai báo.

private: chỉ có thể truy cập bên trong lớp.

Thuộc tính (Instance variable).

Thuộc tính là 1 biến được khai báo bên trong lớp nhưng ở bên ngoài một phương thức, hàm tạo hoặc 1 khối lệnh. Thuộc tính được khởi tạo khi một lớp được khởi tạo và có thể được sử dụng ở bên trong một hàm, hàm tạo hoặc trong một khối lệnh trong lớp đó.

Phương thức

Phương thức là một hàm được viết riêng cho các đối tượng của lớp, chỉ được phép gọi để tác động lên chính các đối tượng của lớp này**.**Một phương thức của lớp bao gồm 3 thành phần chính đó là phạm vi truy câp, giá trị trả về và tên của phương thức đó.

Một phương thức là một tập hợp các câu lệnh được nhóm lại với nhau để thực hiện một chức năng nào đó. Sau đây tôi sẽ giới thiệu đến các bạn cách tạo và gọi phương thức trong chương trình. Trong bài ngày hôm nay, tôi chỉ minh họa cách khai báo và gọi phương thức thông thường

6)

1. Abstract class là gì?

* Lớp trừu tượng(abstract class) đơn giản được xem như một class cha cho tất cả class có cùng bản chất.
* Mỗi lớp con chỉ có thể kế thừa trực tiếp từ một lớp cha bằng từ khóa extends.
* Không thể khởi tạo được các đối tượng thuộc lớp abstract.
* Các phương thức trong abstract class có thể là phương thức abstract hoặc không. Nếu là phương thức abstract thì chỉ được khai báo tên nhưng không viết phần nội dung.
* Lớp con kế thừa từ lớp abstract phải override các phương thức abstract trong đó.

# 2. Interface là gì?

* Interface không phải là một class, mà là bản thiết kế cho các class có cùng cách thức hoạt động.
* Không thể định nghĩa các thuộc tính, khởi tạo đối tượng mà chỉ khai báo các phương thức trong interface.
* Các phương thức chỉ được khai báo tên nhưng không được viết nội dung trong đó.
* Các phương thức của interface phải để dưới dạng public.
* Nếu lớp kế thừa từ interface thì phải override tất cả các phương thức trong đó.
* Một lớp có thể kế thừa từ nhiều interface khác nhau.

7)

B1: ta xác định các đặc điểm chung và trừu tượng

B2:  thiết kế một lớp với tất cả các thuộc tính và hành vi

B3: Xác định xem các lớp con có cần các hành vi (cài đặt của các phương thức) đặc thù của thể loại con cụ thể đó hay không?

8)

Java ko hổ trợ quan hệ đa thừa kế

Giải pháp: sử dụng interface. Một class không thể kế thừa từ nhiều class khác nhưng có thể implements nhiều interface khác nhau một lúc.

9)

Ép kiểu là việc gán giá trị của một biến có kiểu dữ liệu này sang biến khác có kiểu dữ liệu khác.

Vd

Float a=10.5;

Int b=(int)a;

ứng dụng trong oop:chuyển đổi dữ liệu giữa các giá trị trong 1 class

10)

Association:

Liên kết (Association) là mối quan hệ giữa hai lớp riêng biệt được thiết lập thông qua các Đối tượng của chúng. Liên kết có thể là một-một, một-nhiều, nhiều-một, nhiều-nhiều.  
Trong lập trình hướng đối tượng, một đối tượng giao tiếp với các đối tượng khác để sử dụng chức năng và dịch vụ được cung cấp bởi đối tượng đó

Aggregation:

Kết tập là một dạng đặc biệt của Asociation trong đó:

* Kết tập đại diện cho mối quan hệ Has-A (có một).
* Kết tập là unidirectinal association tức là mối quan hệ một chiều.

Ví dụ, khoa có thể có sinh viên nhưng ngược lại (sinh viên có thể có khoa) là không thể và do đó về bản chất là một hướng.

* Trong kết tập, cả hai thực thể có thể tồn tại riêng lẻ , điều đó có nghĩa là kết thúc một thực thể sẽ không ảnh hưởng đến thực thể khác

Compostion:

Hợp thành (Composition) là một trường hợp đặc biệt của Kết tập (Aggregation) trong đó hai thực thể phụ thuộc nhiều vào nhau.

* Hợp thành đại diện cho quan hệ part -of (là một phần).
* Trong hợp thành, cả hai thực thể đều phụ thuộc vào nhau.
* Khi có một mối quan hệ hợp thành giữa hai thực thể, đối tượng composed không thể tồn tại mà không có thực thể khác.

11)

Ý nghĩa của việc sự dụng giao diện(interface) comparable để sắp xếp 2 hay nhiều đối tượng trong 1 mảng hoặc trong 1 danh sách nào đó.

12 vì Comparator linh hoạt hơn Comparable ở chỗ, nếu phương thức compareTo() của Comparable chỉ được override lại một lần và chỉ có thể xét được 1 tiêu chí nhất định thì phương thức compare() của Comparator có thể được gọi nhiều lần và định nghĩa ngay tại vị trí cần so sánh để sắp xếp theo các tiêu chí tùy ý

13)

Ý nghĩa :của cloneable là: Phương thức clone tiết kiệm tác vụ xử lý bổ sung để tạo ra một bản sao chính xác của một đối tượng. Nếu chúng ta thực hiện nó bằng cách sử dụng từ khoá new sẽ mất rất nhiều tiến trình xử lý được thực hiện, đó là lý do tại sao chúng ta sử dụng object cloning

15)ý nghĩa: Serialization là một khái niệm giúp chúng ta có thể chuyển đổi trạng thái của một Java object thành một định dạng nào đó để Java object này có thể được lưu trữ ở đâu đó và sau đó, nó sẽ được sử dụng bởi một tiến trình khác.

Thông thường, khi sử dụng Serialization, Java object của chúng ta sẽ được chuyển đổi qua byte streams và chúng ta có thể lưu byte stream này trong bộ nhớ, trên ổ đĩa, truyền qua mạng đến một server nào đó hoặc cũng có thể lưu chúng vào database.

Và khi một tiến trình khác sử dụng một Java object đã được Serialization, nó sẽ chuyển đổi định dạng đã Serialization về trạng thái của Java object ban đầu. Nhờ vậy, tiến trình đó có thể sử dụng lại Java object của chúng ta.

Để cho một object có thể sử dụng Serialization được, chúng ta phải cho object của chúng ta hiện thực một interface với tên gọi là **java.io.Serializable**.