Vấn đề bài toán : Khi các tiết gia ngồi vào bàn do số lượng dĩa ăn ( tài nguyên ) có

hạn mà các triết gia khi đói muốn ăn thì phải đủ 2 chiếc dĩa tại thời điểm ( đủ các

tài nguyên) mới ăn được nếu không đủ thì sẽ phải chờ cho đến khi đủ nhưng có 1

trường hợp đó là tất cả các triết gia đều cầm 1 chiếc nĩa và chờ đợi chiếc dĩa từ

hàng xóm ( tương đương với việc chờ được chia sẻ tài nguyên tiến trình khác đang

chiếm giữ ) vậy cứ vòng tròn , ai cũng đợi mà k ai nhường lại, tất cả đều chờ vĩnh

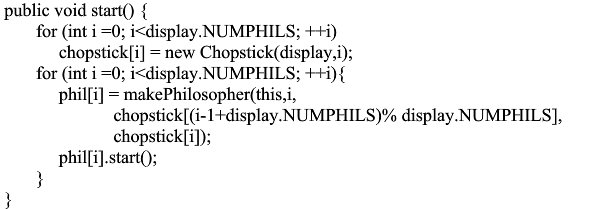
viễn thì ta rơi vào trường hợp đó là đợi vòng tròn (circular wait) dẫn đến là không

ai được ăn hay không luồng nào được thực thi (hay còn gọi là khóa chết – deallock )

giải quyết

Deallock

Tạo 5 tiến trình tương ứng với 5 triết gia

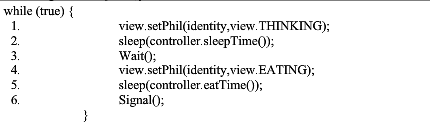


(class chopstick)

* Tạo ra 1 biến quản lý các tài nguyên dùng chung xem tài nguyên đã được sử dụng hay chưa (nếu đã được sử dụng thì là true , nếu chưa thì là false)
* Tạo 1 phương thức put() tương đương với việc triết gia đặt nĩa xuống tương đương vs chia sẻ giải phóng tài nguyên để ng khác sử dụng ( notify() gọi đến luồng trước đó )
* Tạo phương thức get() nếu tài nguyên đang được sử dụng thì triết gia phải đợi (wait()) cho đến khi tài nguyên giải phóng và triết gia sử dụng tài nguyên sẽ khóa các tiến trình khác

(class philosopher)

-đánh số ID cho từng luồng và quản lí tiến trình tiến vào



Với 1 ,2 là tiến trình đang đợi vào vùng dữ liệu , 3 kiểm tra dữ liệu vào vùng , 4,5 là thực thi và 6 là thoát khỏi và giải phóng tài nguyên

Có thể giải quyết bằng cách tạo hai hàm chia đánh số các triết gia sau đó nếu các triết gia số chẵn thì lấy bên phải còn triết gia số lẻ sẽ lấy bên trái khi đó số thứ tự khác nhau sẽ có thứ tự lấy khác nhau

