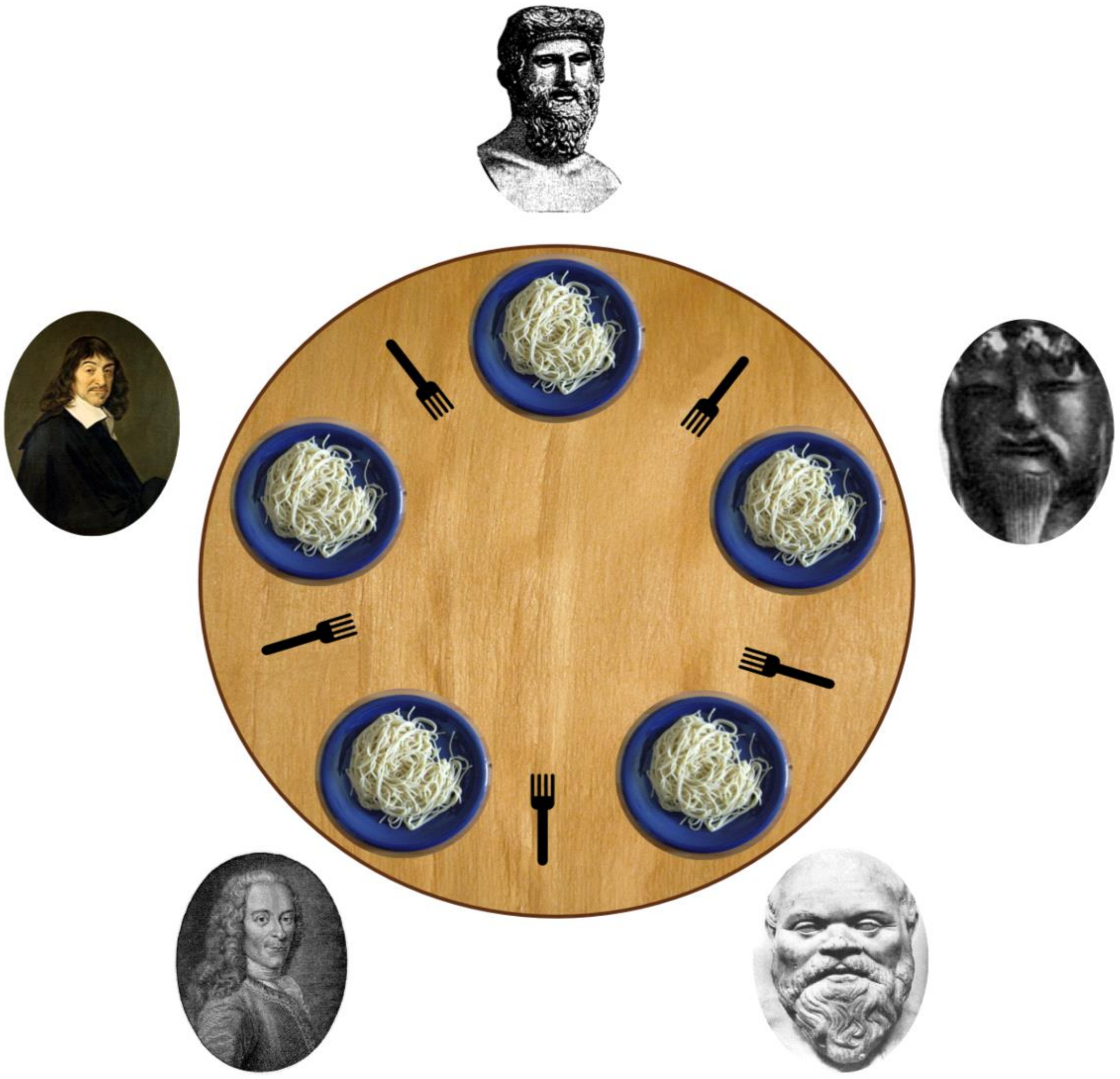

What Is The Dinig Philosophers ?

Simlpy It's One Of The Best Classic Examples Which Is Used To Describe Synchronization Issues In a Multi-Threaded Background .

In Computer Science, The Dining Philosophers Problem Is An Example Problem Often Used In Concurrent Algorithm Design To illustrate Synchronization Issues And Techniques For Resolving Them.

Our Problem ?

Five philosophers dine together at the same table. Each philosopher has their own place at the table. There is a fork between each plate. The dish served is a kind of spaghetti which has to be eaten with two forks. Each philosopher can only alternately think and eat. Moreover, a philosopher can only eat their spaghetti when they have both a left and right fork. Thus two forks will only be available when their two nearest neighbors are thinking, not eating. After an individual philosopher finishes eating, they will put down both forks. The problem is how to design a regimen (a concurrent algorithm) such that no philosopher will starve; i.e., each can forever continue to alternate between eating and thinking, assuming that philosopher can know when others may want to eat or thin.

It's called Dinig Philosophers prblem

problem solution :

1-think until the left fork is available; when it

is, check if right fork is available;

2-if available pick both of them if not available leave both

3- when both forks are held, eat for a fixed

amount of time ;

4- put the left fork down ;

5- put the right fork down ;

6- repeat from the beginning

it's pseudo code :

```
while(true)
{
think()   //Initially thinking about the whole
universal things.
pick_up_left_fork() //Readying to eat as
philosopher gets hungry eventually.
pick_up_right_fork()
eat()
put_down_right_fork()
put_down_left_fork()
think() //Back to start thinking as hungry is over.
```

Example Of Deadlock:

IF Each Philosopher takes the left
chopstick, then all The Philosopher try to
take the right Chopstick to be free that will lead to
deadlock And We Lost The Progress

Solution For Deadlock:

We solve the deadlock by make the
process of take the chopstick,
Synchronous Operation So I make
Sure, that the Philosopher Will Take
The right chopstick and left chopstick
At same time

Example Of Starvation:

We solve the deadlock Problem, but we
still have a Problem what about one of
the philosophers doesn’t have a chance to
eat So that will lead to starvation "if one
philosophers will get the first chopstick he found that
the second isn't free so he wait , and when the
second is free and he try to get it ,the first when was

been taken and so on"

Solution Of Starvation:

We make finite time for each philosopher to eat then the next philosopher eats and so on.

Example At Real Life :

shared bathroom can be taken. A shared bathroom can be used by many people but it can be used by only one person at the time. If someone is using the bathroom, others have to wait until that person to come out. Then the one of the waited people can use the bathroom