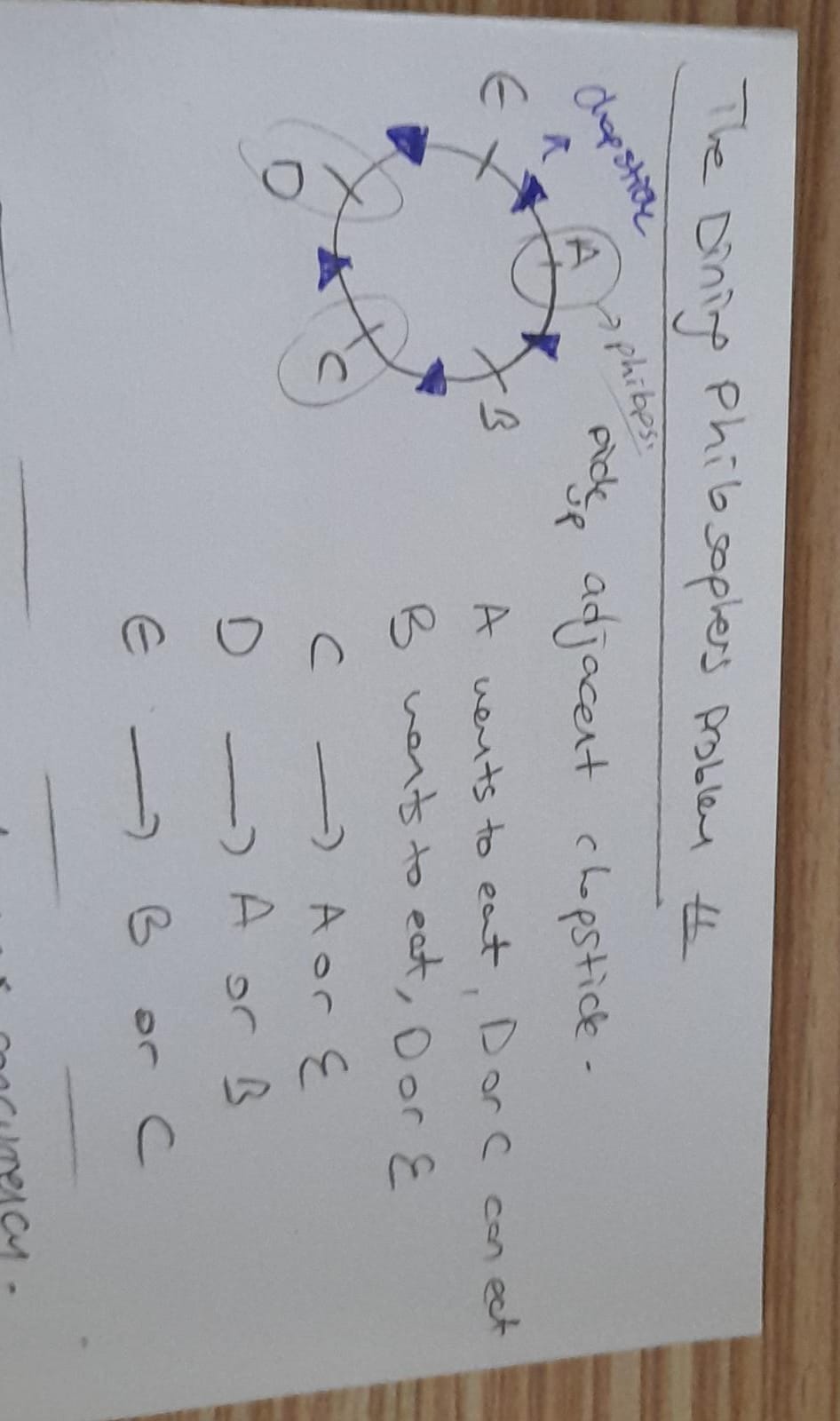
In this photo you can see that how we can decide 2 philosophers can eat at the same time.



The problem => <https://legacy.cs.indiana.edu/classes/p415-sjoh/hw/project/dining-philosophers/index.htm>

We are asked to solve the dining philosophers problem regarding following needs:

* Deadlock free solution with max concurrency (odd numbers, max 27)
* A thread = a philosopher
* Pthreads, mutex, and condition variables to synchronize
* Monitor-based or semaphores solutions
* The program 🡪 phsp & arguments 🡪 <num-phsp> <min-think> <max-think> <min-dine> <max-dine> <dst> <num>
* 1ms <= arguments <= 60 seconds
* Dst 🡪 distribution (uniform or exponential)
* Mean for exponential
  + (min-think + max-think)/2
  + (min-dine + max-dine)/2
* Thinking time will be selected randomly
* Complete after all philosophers complete dining

I looked up several sites to decide which solution to prefer.

Monitor-based solution: <https://www.geeksforgeeks.org/dining-philosophers-solution-using-monitors/>

Semaphores: <https://cs.gordon.edu/courses/cs322/lectures/transparencies/dining_phil.html>

In the solution, I solved the problem by using monitor-based solution since with semaphores solution the program can suffer from deadlock (There are several ways to avoid this issue like not allowing all philosophers to sit and eat/think at once.).

**Solution:**

At the beginning, creation of all of philosophers are done and then they started to think *randomly*. In order to eat for any philosophers, they need to check if the adjacent philosophers are eating or not (look at the picture at the beginning of the file). If the other philosophers are not eating, the philosopher who done the checking will start eating.

**Random Generation:**

For thinking and dining we needed random values, so a function with distribution features (exponential and uniform) for this purpose is added.

Uniform(max,min) =max\*random\_number

Exponential(max,min) = (-(min+max)/2)\*log(1-random\_number)

* <https://www.calculator.net/standard-deviation-calculator.html> (I used this website to see the mean and standard deviation values of the results.)

Some tests:

1. **phsp 5 50 100 5 10 exponential 15**

Philosopher 5 duration of hungry state = 0

Philosopher 2 duration of hungry state = 0

Philosopher 1 duration of hungry state = 5

Philosopher 3 duration of hungry state = 3

Philosopher 4 duration of hungry state = 3

* + Population => (5,0,3,3,0)
  + Mean => 2.2
  + Standard Deviation => 1.9390719429665

1. **phsp 5 50 100 5 10 uniform 15**

Philosopher 4 duration of hungry state = 0

Philosopher 2 duration of hungry state = 5

Philosopher 3 duration of hungry state = 2

Philosopher 5 duration of hungry state = 0

Philosopher 1 duration of hungry state = 1

* + Population => (1,5,2,0,0)
  + Mean => 1.6
  + Standard Deviation => 1.8547236990991

1. **phsp 5 50 100 5 10 exponential 45**

Philosopher 2 duration of hungry state = 0

Philosopher 4 duration of hungry state = 0

Philosopher 1 duration of hungry state = 0

Philosopher 3 duration of hungry state = 4

Philosopher 5 duration of hungry state = 17

* + Population => (0,0,4,0,17)
  + Mean => 4.2
  + Standard Deviation => 6.5848310532617

1. **phsp 5 50 100 5 10 uniform 45**

Philosopher 2 duration of hungry state = 0

Philosopher 5 duration of hungry state = 7

Philosopher 4 duration of hungry state = 0

Philosopher 3 duration of hungry state = 11

Philosopher 1 duration of hungry state = 2

* + Population => (2,0,11,0,7)
  + Mean => 4
  + Standard Deviation => 4.3358966777358