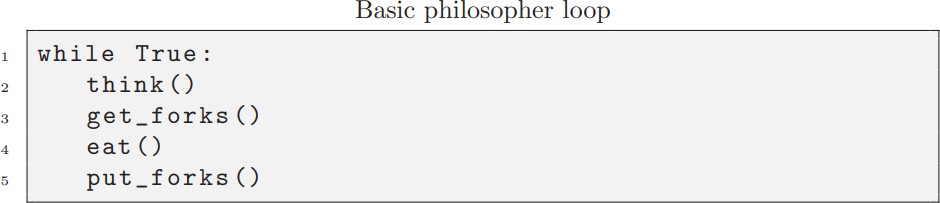
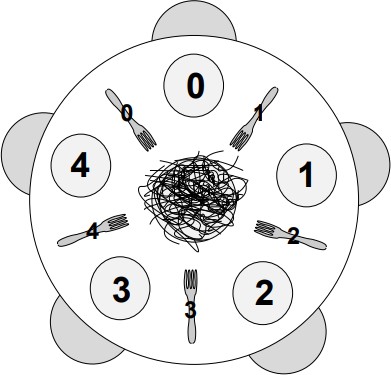
**IPC (Inter Process Communication)**

1. **Dining Philosophers Problem**

The Dining Philosophers Problem was proposed by Dijkstra in 1965, when dinosaurs ruled the earth. It appears in a number of variations, but the standard features are a table with five plates, five forks (or chopsticks) and a big bowl of spaghetti. Five philosophers, who represent interacting threads, come to the table and execute the following loop:



The forks represent resources that the threads have to hold exclusively in order to make



progress. Each philosopher alternates between thinking and eating. To eat, a philosopher must use the two forks adjacent to them. After eating, they put down the forks, and then they can start thinking again.

Assuming that the philosophers know how to think and eat, our job is to write a version

of get forks and put forks that satisfies the following constraints:

* + Only one philosopher can hold a fork at a time.
  + It must be impossible for a deadlock to occur.
  + It must be impossible for a philosopher to starve waiting for a fork.
  + It must be possible for more than one philosopher to eat at the same time.

1. Implement a synchronization problem where 26 threads each print a letter from A to Z in serial order. This ensures that the letters are printed in alphabetical order from A to Z without any disorder, despite being handled by separate threads.

Hints:

1. Use an array of threads (array size will be 26)
2. Use an array of condition variables (This will make life easier)
3. The thread that prints the letter A should give signal to a thread that will print letter B.