# Filósofos

## EQUIPO 5


HERNÁNDEZ LOZANO JUAN PABLO

MOSQUEDA GARCÍA RAÚL ISAID

RUIZ PUGA INGRID PAMELA

VELEROS VEGA LUIS ALFONSO


# Problema

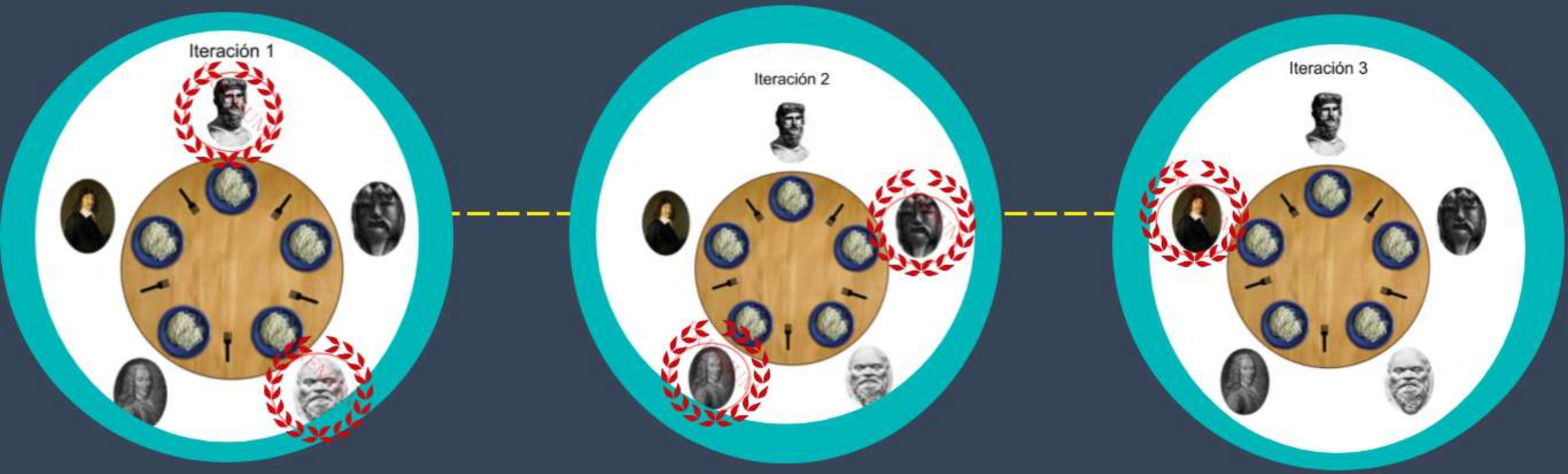
- En una cena de 5 filósofos cuyas actividades se reducen a comer y pensar, por lo que si no están comiendo están pensando. Se encuentran en una mesa redonda donde hay 5 tenedores (los tenedores están adyacentes a cada lado) para comer. El principal problema es que se desea que ningún filósofo muera de hambre dado que necesitan tomar dos tenedores adyacentes para poder comer. De esta forma solo podrán comer simultáneamente los filósofos que estén alternados (dos en total).

# Solución

Se tomará un filósofo para que coma, y el filósofo que se encuentra 2 posiciones a la derecha de este.

Cuando los filósofos terminen de comer, el filósofo a su derecha de los que ya comieron serán los próximos en comer

# Iteraciones



# Monitores

Los monitores son estructuras de datos abstractas destinadas a ser usadas sin peligro por más de un hilo de ejecución. La característica que los define es que sus métodos son ejecutados con exclusión mutua.

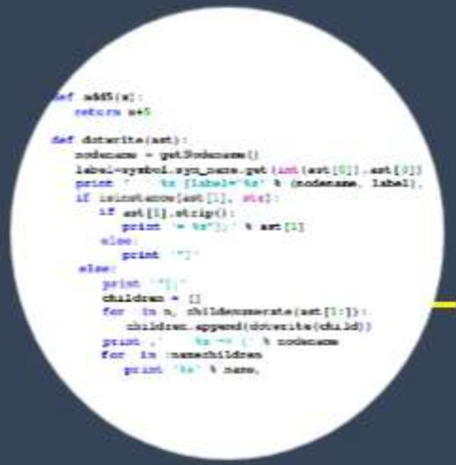
En cada momento en el tiempo, un hilo como máximo puede estar ejecutando cualquiera de sus métodos. Esta exclusión mutua simplifica el razonamiento de implementar monitores en lugar de código a ser ejecutado en paralelo.

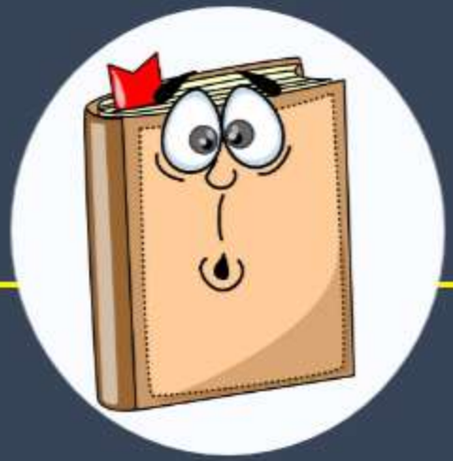
# Monitores

- Las estructuras de datos internas del monitor cuya finalidad es ser compartidas por múltiples procesos concurrentes, sólo pueden ser inicializadas, leídas y actualizadas por código propio del monitor.
- Los únicos componentes del monitor públicos (visibles desde módulos externos) son los procedimientos y funciones exportadas.
-

- El monitor garantiza el acceso mutuamente exclusivo a los procedimientos y funciones de la interfaz. Si son invocados concurrentemente por varios procesos, sólo la ejecución de un procedimiento del monitor es permitido. Los procesos no atendidos son suspendidos hasta que la ejecución del procedimiento atendido acabe.
- Dado que todo el código relativo a un recurso o a una variable compartida está incluido en el módulo del monitor, su mantenimiento es más fácil y su implementación es más eficiente.

# Componentes





- Inicialización

Código a ser ejecutado cuando el monitor es creado

- Datos privados

Procedimientos privados, que sólo pueden ser usados desde dentro del monitor y no son visibles desde fuera

- Métodos del monitor

Procedimientos que pueden ser llamados desde fuera del monitor.

- Cola de entrada

Hilos que han llamado a algún método del monitor pero no han podido adquirir permiso para ejecutarlos aún.

# Pseudo-código

```
Filosofos = [Sócrates,Pluton,Mickey,Donald,Aristóteles]
tenedor = range(len(Filosofos))
n = número_de_rondas

def comer(n, Filosofos):
    for j in range(n):
        for i in range (3):                          #Sección Crítica
            if (i==2):
                Filosofos[i+2].come(i+2)
                time(10)

            else:
                Filosofos[i].come(i)
                Filosofos[i+2].come(i+2)
                time(10) <- tiempo de comer


def come(filosofo,posicion):
 filosofo.toma_tenedor(tenedor[i])
 filosofo.toma_tenedor(tenedor[i+1])
```

# Thank You
## for listening!