Algoritmo de Dekker

Algoritmo

```
boolean flag[2] = {false, false}
int turno = 0

Proceso Pi:
repeat
        flag[i] = true;
        while ( flag[j] )
        {
                flag[i] = false;
                while(turno!=i){ };
                flag[i] = true;
        }
        /* sección crítica*/
        turno = j;
        flag[i] = false;

end repeat
```

Algoritmo en cada proceso

Proceso P0:

```
repeat
        flag[0] = true;
         while(flag[1])
        {
                flag[0] = false;
                while(turno!=0){ };
                flag[0] = true;
        }

        /* sección crítica*/
        turno = 1;
        flag[0] = false;

end repeat
```

Proceso P1:

```
repeat
        flag[1] = true;
        while(flag[0])
        {
                flag[1] = false;
                while(turno!=1);
                flag[1] = true;
        }

        /* sección crítica*/
        turno = 0;
        flag[1] = false;

end repeat
```

Prueba de escritorio

Proceso P0	Proceso P1	Flag[0]	Flag[1]	Turno

Algoritmo de Peterson

Algoritmo

```
boolean flag[2] = {false, false}
int turno = 0

Proceso Pi:
repeat
        flag[i] = true;
        turno = j
        while(flag[j] && turno == j){ };

        /* sección crítica*/

        flag[i] = false;

end repeat
```

Algoritmo en cada proceso

Proceso P0:

```
repeat
        flag[0] = true;
        turno = 1;
        while(flag[1] && turno == 1){ };

        /* sección crítica*/

        flag[0] = false;

end repeat
```

Proceso P1:

```
repeat
        flag[1] = true;
        turno = 0;
         while(flag[0] && turno ==0){ };

        /* sección crítica*/

        flag[1] = false;

end repeat
```

Prueba de escritorio

Proceso P0	Proceso P1	Flag[0]	Flag[1]	Turno