PONTIFICIA UNIVERSIDAD CATOLICA DEL PERU
FACULTAD DE CIENCIAS E INGENIERIA
Laboratorio de Sistemas Operativos

CONCURRENCIA EN PYTHON

1) (13 puntos) Se tiene el siguiente programa en Python (*alterna.py*)

alterna.py

```
1  import threading as th
2  import random as rand
3  import time
4
5  def izq_der():
6      r = rand.randint(10,20)
7      time.sleep(r/100)
8      print("------->", flush=True)
9
10 def der_izq():
11     r = rand.randint(10,20)
12     time.sleep(r/100)
13     print("<-------", flush=True)
14
15 ths = []
16 for i in range(10):
17     t = th.Thread(target=izq_der)
18     ths.append(t)
19
20 for i in range(10):
21     t = th.Thread(target=der_izq)
22     ths.append(t)
23
24 for i in range(20):
25     ths[i].start()
26
27 for i in range(20):
28     ths[i].join()
29
```

cuya salida, al ejecutarse, es la siguiente (usted puede obtener una salida diferente):

```
alejandro@abdebian:2020-1$ python3 alterna.py
------->
------->
<-------
<-------
<-------
------->
<-------
------->
------->
------->
------->
<-------
<-------
<-------
------->
<-------
------->
<-------
------->
<-------
alejandro@abdebian:2020-1$
```

Requerimientos:

- ➔ Sincronizar los hilos de forma que haya alternancia estricta entre el hilo que invoca `izq_der` y el otro hilo que invoca `der_izq`. Debe emplear *Conditions Objects* de Python.

- ➔ En general, al código base le puede añadir, pero no le puede eliminar nada. Solo puede añadir las líneas necesarias a las funciones `izq_der` y `der_izq`, para lograr la sincronización. También puede añadir variables globales.

La salida debe ser la siguiente:

```
alejandro@abdebian:2020-1$ python3 alterna.py
------->
<-------
------->
<-------
------->
<-------
------->
<-------
------->
<-------
------->
<-------
------->
<-------
------->
<-------
------->
<-------
------->
<-------
alejandro@abdebian:2020-1$
```

2) (7 puntos) Se tiene el siguiente programa en Python (*sincroniza.py*)

sincroniza.py

```
1  import threading as th
2  import random as rand
3  import time
4
5  def A():
6      while True:
7          r = rand.randint(1,50)
8          time.sleep(r/100)
9          print('A',end=' ',flush=True)
10
11 def B():
12     while True:
13         r = rand.randint(1,50)
14         time.sleep(r/100)
15         print('B',end=' ',flush=True)
16
17 def C():
18     while True:
19         r = rand.randint(1,50)
20         time.sleep(r/100)
21         print('C',end=' ',flush=True)
22
23 def D():
24     while True:
25         r = rand.randint(1,50)
26         time.sleep(r/100)
27         print('D',end=' ',flush=True)
28
29 if __name__ == "__main__":
30
31     t = []
32     h1=th.Thread(target=A)
33     t.append(h1)
34     h2=th.Thread(target=B)
35     t.append(h2)
36     h3=th.Thread(target=C)
37     t.append(h3)
38     h4=th.Thread(target=D)
39     t.append(h4)
40
41     lista = rand.sample(range(4),4)
42     for i in lista:
43         t[i].start()
44
```

Observe que el programa tiene cuatro hilos, cada uno de los cuales imprime una letra (`A`, `B`, `C` ó `D`) en un bucle sin fin. Al ejecutarlo, si usted desea pausar la salida debe presionar Ctrl+s y Ctr+q para continuar. Si usted desea interrumpirlo debe presionar Ctrl+c

Al ejecutarlo y presionar Ctr+s, se obtiene la siguiente salida

```
alejandro@abdebian:2020-1$ python3 sincroniza.py
B C D C A B C B D A B A C C D B D A C B C A D D B A C A A D C B C A C B A D A A
C B D C A B A D C D C D B C A D A B C A D B A C D D B C B B A D C A B C A C D B
C D D B A C A D B C C A D B C A C D A A B A D C A D B D D C C D A D B C A D C A
B C C D B A D C C A D B B D C C A D A D B B C D B A D A A C C D C B A A C B D D
C A D B D B C C B D A C B C D B A D D A D C D C B B B D A B C D
```

Usted puede obtener una secuencia distinta.

Requerimientos:

Sincronizar los hilos usando barreras de forma que cumplan las siguientes condiciones

- Las letras `B` y `C` (ó `C` y `B`) se deben imprimir una sola vez, siempre después de `A` y antes de la siguiente `A`.

  Las siguientes son secuencias no válidas:

  ```
  ... A B B C A ...     # Después de A, se imprime dos veces B
  ... A B C D C A ...   # Después de A, se imprime C dos veces
  ```

  Las siguientes son secuencias válidas:

  ```
  ... A B C D A ...
  ... A C B A D ...
  ... A C B D A ...
  ... A B C A D ...
  ```

- La letra `D` se deben imprimir una sola vez, siempre después de `C` y antes de la siguiente `C`.

  Las siguientes son secuencias válidas:

  ```
  ... C D A C B A D ...
  ... A B C A D C B ...
  ```

- El programa debe cumplir las dos condiciones, anteriormente señaladas, al mismo tiempo

- En general, al código base le puede añadir, pero no le puede eliminar nada. Solo puede añadir las líneas necesarias a las funciones `A`, `B`, `C` y `D`, para lograr la sincronización. También puede añadir variables globales.

La salida debe ser la siguiente:

```
alejandro@abdebian:2020-1$ python3 sincroniza.py
A B C A D B C D A C B A D C B D A C B A D C B A D C B D A B C D A B C A D C B A
D B C A D B C A D B C A D C B A D C B A D B C D A B C A D B C A D B C D A B C D
A B C A D B C A D B C A D C B A D B C A D C B A D B C A D B C A D B C A D B C A
D C B A D C
```

Lima, 22 de mayo de 2020.

Prof. Alejandro T. Bello Ruiz