Programación Concurrente

Práctica 4: Semáforos

1. Dados los siguientes threads

```
thread {                    thread {
  print('A');                 print('E');
  print('B');                 print('F');
  print('C');                 print('G');
}                           }
```

utilizar semáforos para garantizar que:

   a) 'A'se muestra antes que 'F'

   b) 'F'se muestra antes que 'C'

2. Dados

```
thread {                    thread {
  print('C');                 print('A');
  print('E');                 print('R');
}                             print('O');
                            }
```

utilizar semáforos para garantizar que las únicas salidas posibles sean ACERO y ACREO.

3. Considerar los siguientes tres threads:

```
thread {              thread {              thread {
  print("R");           print("I");           print("O");
  print("OK");          print("OK");          print("OK");
}                     }                     }
```

Utilizar semáforos para garantizar que el único resultado impreso será R I O OK OK OK (asumimos que print es atómico).

4. Considere los siguientes threads que comparten dos variables y y z.

```
global int y = 0, z = 0;
```

```
thread {              thread {
  int x;                y = 1;
  x = y + z;            z = 2;
}                     }
```

   a) Cuáles son los posibles valores finales de x?

   b) Es posible utilizar semáforos para limitar que los valores posibles de x sean sólo dos?

5. Dados

```
thread                thread                thread
  while (true) {        while (true) {        while (true) {
    print('A');           print('E');           print('H');
    print('B');           print('F');           print('I');
    print('C');           print('G');         }
    print('D');         }
  }
```

Agregar semáforos para garantizar que:

   a) cantidad de F $\leq$ cantidad de A

   b) cantidad de H $\leq$ cantidad de E

   c) cantidad de C $\leq$ cantidad de G

6. Se tienen tres threads A, B, C. Se desea que la operación op_C que ejecuta C se realice sólo luego de que A haya ejecutado op_A y B haya ejecutado op_B. Como sincronizaría estos procesos utilizando semáforos?

7. Considere los siguientes dos threads:

```
thread                          thread
  while (true)                    while (true)
    print('A');                     print('B');
```

   a) Utilizar semáforos para garantizar que en todo momento la cantidad de A y B difiera al máximo en 1.

   b) Modificar la solución para que la única salida posible sea ABABABABAB...

8. Los siguientes threads cooperan para calcular el valor $N2$ que es la suma de los primero N números impares. Los procesos comparten las variables N y $N2$ inicializadas de la siguiente manera: $N = 50$ y $N2 = 0$.

```
thread {                        thread
  while (N > 0)                   while (true)
    N = N-1;                        N2 = N2 + 2*N + 1;
  print(N2);
}
```

   a) Dar una solución que utilizando semáforos garantice que se muestra el valor correcto de $N2$.