```
#program to demonstrate race condition.

#Program to get letters in pairs

#remove Locks (demo race condition)

# execute the below program.


from threading import *

import random

import sys

import time

lock=Lock()

# get letters in pairs

def fn1() :

        s = 'ABCDEFGH'

        for i in range(0, len(s)) :

                lock.acquire()

                print(s[i], end=' ')

                sys.stdout.flush()

                time.sleep(int(random.random() * 3))

                print(s[i], end=' ')

                sys.stdout.flush()

                lock.release()

                time.sleep(int(random.random() * 3))


def fn2() :

        s = 'abcdefgh'

        for i in range(0, len(s)) :

                lock.acquire()

                print(s[i], end=' ')

                sys.stdout.flush()

                time.sleep(int(random.random() * 3))

                print(s[i], end=' ')
        sys.stdout.flush()
        lock.release()
        time.sleep(int(random.random() * 3))

t1=Thread(target=fn1)
t2=Thread(target=fn2)
t1.start()
t2.start()
t1.join()
t2.join()
```

sys.stdout.flush() (from geeks)

A data buffer is a region of physical memory storage used to temporarily store data while it is being moved from one place to another. The data is stored in a buffer as it is retrieved from an input device or just before it is sent to an output device or when moving data between processes within a computer. Python’s standard out is buffered. This means that it collects some data before it is written to standard out and when the buffer gets filled, then it is written on the terminal or any other output stream.

```
import time
for i in range(10):
  print(i)
  time.sleep(1)
```

When the above program is executed, the numbers from 0 to 9 are printed after every second on a new line, i.e., the output is automatically flushed out. This is because, by default end parameter of print statement is set to ‘\n’ which flushes the output.

```
import time
for i in range(10):
   print(i, end =' ')
   time.sleep(1)
```

When the above program is executed, then there is no output for the first 9 seconds, then at the 10th, all the 10 numbers from 0 to 9 appear simultaneously in a line separated by spaces. This is because the output is buffered and it is not flushed by any means.

```
import time
import sys

for i in range(10):
    print(i, end =' ')
    sys.stdout.flush()
    time.sleep(1)
```

When the above program is executed, the numbers from 0 to 9 are printed every second on the same line separated by spaces. This is because calling `sys.stdout.flush()` forces it to “flush” the buffer, meaning that it will write everything in the buffer to the terminal, even if normally it would wait before doing so. **(or)**

```
import time
for i in range(10):
    print(i, end =' ', flush = True)
    time.sleep(1)
```