

**DeadLock code:**

public class TestThread {

    public static Object Lock1 = new Object();

    public static Object Lock2 = new Object();

    public static Object Lock3 = new Object();

    public static void main(String args[]) {

        // Create and start 3 threads

        ThreadDemo1 T1 = new ThreadDemo1();

        ThreadDemo2 T2 = new ThreadDemo2();

        ThreadDemo3 T3 = new ThreadDemo3();

        T1.start();

        T2.start();

        T3.start();

    }

    private static class ThreadDemo1 extends Thread {

        public void run() {

            synchronized (Lock1) {

                System.out.println("Thread 1: Holding lock 1...");

                try {

                    Thread.sleep(10);

                } catch (InterruptedException e) {

                }

                System.out.println("Thread 1: Waiting for lock 2...");

                synchronized (Lock2) {

                    System.out.println("Thread 1: Holding lock 1 & 2...");

                    try {

                        Thread.sleep(10);

                    } catch (InterruptedException e) {

                    }

                    System.out.println("Thread 1: Waiting for lock 3...");

                    synchronized (Lock3) {

                        System.out.println("Thread 1: Holding lock 1, 2 & 3...");

                    }

                }

            }

        }

    }

    private static class ThreadDemo2 extends Thread {

        public void run() {

            synchronized (Lock2) {

                System.out.println("Thread 2: Holding lock 2...");

                try {

                    Thread.sleep(10);

                } catch (InterruptedException e) {

                }

                System.out.println("Thread 2: Waiting for lock 3...");

                synchronized (Lock3) {

                    System.out.println("Thread 2: Holding lock 2 & 3...");

                    try {

                        Thread.sleep(10);

                    } catch (InterruptedException e) {

                    }

                    System.out.println("Thread 2: Waiting for lock 1...");

                    synchronized (Lock1) {

                        System.out.println("Thread 2: Holding lock 1, 2 & 3...");

                    }

                }

            }

        }

    }

    private static class ThreadDemo3 extends Thread {

        public void run() {

            synchronized (Lock3) {

                System.out.println("Thread 3: Holding lock 3...");

                try {

                    Thread.sleep(10);

                } catch (InterruptedException e) {

                }

                System.out.println("Thread 3: Waiting for lock 1...");

                synchronized (Lock1) {

                    System.out.println("Thread 3: Holding lock 1 & 3...");

                    try {

                        Thread.sleep(10);

                    } catch (InterruptedException e) {

                    }

                    System.out.println("Thread 3: Waiting for lock 2...");

                    synchronized (Lock2) {

                        System.out.println("Thread 3: Holding lock 1, 2 & 3...");

                    }

                }

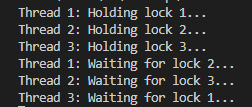
            }

        }

    }

}

Output:



**DeadLock Solution(sequence of lock in thread will be same for all threads) code:**

public class DeadLockSolution {

    public static Object Lock1 = new Object();

    public static Object Lock2 = new Object();

    public static Object Lock3 = new Object();

    public static void main(String args[]) {

        // Create and start 3 threads

        ThreadDemo1 T1 = new ThreadDemo1();

        ThreadDemo2 T2 = new ThreadDemo2();

        ThreadDemo3 T3 = new ThreadDemo3();

        T1.start();

        T2.start();

        T3.start();

    }

    private static class ThreadDemo1 extends Thread {

        public void run() {

            synchronized (Lock1) {

                System.out.println("Thread 1: Holding lock 1...");

                try {

                    Thread.sleep(10);

                } catch (InterruptedException e) {

                }

                System.out.println("Thread 1: Waiting for lock 2...");

                synchronized (Lock2) {

                    System.out.println("Thread 1: Holding lock 1 & 2...");

                    try {

                        Thread.sleep(10);

                    } catch (InterruptedException e) {

                    }

                    System.out.println("Thread 1: Waiting for lock 3...");

                    synchronized (Lock3) {

                        System.out.println("Thread 1: Holding lock 1, 2 & 3...");

                    }

                }

            }

        }

    }

    private static class ThreadDemo2 extends Thread {

        public void run() {

            synchronized (Lock1) {

                System.out.println("Thread 2: Holding lock 1...");

                try {

                    Thread.sleep(10);

                } catch (InterruptedException e) {

                }

                System.out.println("Thread 2: Waiting for lock 2...");

                synchronized (Lock2) {

                    System.out.println("Thread 2: Holding lock 1 & 2...");

                    try {

                        Thread.sleep(10);

                    } catch (InterruptedException e) {

                    }

                    System.out.println("Thread 2: Waiting for lock 3...");

                    synchronized (Lock3) {

                        System.out.println("Thread 2: Holding lock 1, 2 & 3...");

                    }

                }

            }

        }

    }

    private static class ThreadDemo3 extends Thread {

        public void run() {

            synchronized (Lock1) {

                System.out.println("Thread 3: Holding lock 1...");

                try {

                    Thread.sleep(10);

                } catch (InterruptedException e) {

                }

                System.out.println("Thread 3: Waiting for lock 2...");

                synchronized (Lock2) {

                    System.out.println("Thread 3: Holding lock 1 & 2...");

                    try {

                        Thread.sleep(10);

                    } catch (InterruptedException e) {

                    }

                    System.out.println("Thread 3: Waiting for lock 3...");

                    synchronized (Lock3) {

                        System.out.println("Thread 3: Holding lock 1, 2 & 3...");

                    }

                }

            }

        }}}

Output:

