Lab4 -- 死锁

产生死锁的4个必要条件

死锁就是两个或者多个进程，互相请求对方占有的资源。

- 互斥条件：一个资源每次只能被一个进程使用
- 请求与保持条件：一个进程因请求资源而阻塞时，对已获得的资源保持不放
- 不剥夺条件:进程已获得的资源，在末使用完之前，不能强行剥夺
- 循环等待条件:若干进程之间形成一种头尾相接的循环等待资源关系

死锁停在第几次的截图

运行后bash文件后，程序在36次死锁，此时的count值是2000000

对上述程序产生死锁的解释

bash 文件流，在系统中运行Java文件，设置了一个循环，让程序循环一百次

```
#!/bin/bash
for ((c=1;c<=100;c++))
do
        echo "$c times"
        java Deadlock
done
```

Java部分程序截图，在截图中我们知道，线程t开始时，线程B同时启动，然后b占有了b的资源，等待A释放资源；而线程A等待count的时间过后占有了a的资源，等待B释放资源。在循环一百次中，很有可能在等待时出现相互等待的情况，于是产生死锁。

```
class Deadlock implements Runnable{
     A a = new A();
     B b = new B();

     Deadlock(){
          Thread t = new Thread(this);
          int count = 2000000;

          t.start();
          while(count-->0);
          a.methodA(b);

     }
     public void run(){
          b.methodB(a);
     }
     public static void main(String args[]){
          new Deadlock();
     }

}
```

实验感想

配置死锁，其实没有什么困难，可能只是因为每个人的机子不一样导致count值不一样吧，过程也没有遇到什么困难，实在是一次值得回味的实验。